The influence of shear connection strength and stiffness on the resistance of steel-concrete composite sandwich panels to out-of-plane forces

Philip Francis

Thesis submitted to The University of Surrey for the degree of Doctor of Philosophy

March 2018
Abstract

Steel-concrete-steel (SCS) sandwich panels are an efficient means of achieving a strong and stable composite wall. Development in the 70's and 80's focussed on tunnelling, with other applications, particularly in the defence and offshore sectors, appearing later.

Renewed focus has been placed on the system in recent years due to a proliferation of proposals for new nuclear power stations in Europe. Many new nuclear projects that have been completed in recent years have been significantly delayed by problems with reinforcement congestion. SCS construction offers a potential solution to this, since reinforcement is either significantly reduced or eliminated entirely in most designs. As a result of this renewed interest, industry has sought to develop improved design rules, both for economy and easier regulatory approval.

As with any composite system, the strength of the system is derived from the ability of the materials to interface efficiently with each other where they are connected. Review of existing design guides and research showed a gap in understanding of the effects of shear connection on the overall behaviour of the system, particularly when resisting out-of-plane loads. This thesis aims to improve this understanding, leading to improved design provision and a wider range of applications for SCS panels in industry.

An extensive literature search found a large body of test results. However, the majority of these tests are for designs where shear connection is over-provisioned, meaning shear connection is not critical. The tests that were conducted with lower degrees of shear connection were found to be insufficient to draw definitive conclusions about changes in behaviour. For this reason, numerical modelling using finite element analysis was used to supplement the test data. A validation and verification exercise was performed, which showed that the model accurately predicted the behaviour seen in testing, for all of the relevant failure modes.

This thesis focusses on the three design checks that are required for panels subject to
out-of-plane loads; bending resistance, shear resistance and deflection. The effect of reduced shear connection on each of these design checks is explored in turn.

For bending resistance, design rules based on first principles cross-section equilibrium are found to accurately predict the point of failure for the majority of cases. However, the existing assumption of a smooth profile of shear connector force is found to be incorrect on the tension plate, with tensile cracking leading to discontinuities in the stud force profile. Further interpretation of this result shows that this can lead to an unconservative prediction of the failure load when a panel with a low degree of shear connection is subject to a uniformly-distributed load (UDL). A new design rule is presented for this situation.

Design equations for shear resistance are found to vary considerably between design codes and countries. As with the bending check, the test database is found to be lacking in tests with low enough degrees of shear connection to draw definitive conclusions about any changes in behaviour. A parametric FE study is presented to investigate these effects. The study focusses on varying the degree of shear connection for groups of beams loaded at different shear-span to depth ratios. Different behaviour is observed in each group, with the influence of shear connection varying, depending on which shear transfer action is dominant. The study shows that unconservative predictions are made for a number of the design models, particularly for slender beams with low degrees of shear connection. A new adjustment is presented for the Eurocode shear resistance model that removes the unconservative predictions. The models from the fib Model Code are suggested as a better alternative, again with some adjustment to account for reduced degree of shear connection.

Deflection of SCS panels is usually predicted using linear-elastic models. Debate has occurred about whether to base the stiffness used on the contribution of the steel plates only, or whether the concrete stiffness should be included. This work finds that a partial concrete contribution should be assumed. It is also found that simple bending prediction models, based on Euler-Bernoulli principles, tend to overestimate stiffness for beams with low shear-span to depth ratios. In these cases, models that include shear deformation (such as the model by Timoshenko) are found to produce more accurate predictions. Reduced shear connection is found to lead to non-linear load deflection response curves, which cannot be easily approximated with linear-elastic models. A new load-stiffness curve is proposed for simplified non-linear modelling, which could be easily implemented in most current software packages with non-linear solvers.

Finally, partial resistance factors for the bending and shear design checks are calculated,
using the procedure presented in Annex D of Eurocode 0. This method takes into account the precision and conservativeness of a particular design equation through a systematic comparison with available test data, and penalises studies that are based on limited test data. The procedure is found to be deficient when the design model includes contributions from multiple materials and large numbers of parameters. To overcome this, a novel extension to the existing procedure is proposed, termed the ‘matrix method’.

In general, it is concluded that lower degrees of shear connection are not immediately detrimental to the performance of the system. This thesis highlights the changes in behaviour that can occur, which designers should account for when calculating the resistance of panels. This thesis also presents new adjustments and design rules to allow resistance to be accurately calculated in such cases.
Acknowledgements

Firstly, I would like to thank my supervisors, Dr Juan Sagaseta and Prof. Marios Chryssanthopoulos. I appreciate the enthusiasm, knowledge and rigour that you have both brought to my work.

I would like to thank all of my colleagues at the Steel Construction Institute (SCI), for their support and patience while I pursued my study. I would like to particularly thank Dr Bassam Burgan for securing and managing the project, and supervising my work on a number of the work packages. I would also like to acknowledge the help of my colleagues that worked on the SCIENCE project; Dr Eleftherios Aggelopoulos and Sofia Meleti.

The work on this thesis was developed for and inspired by work on the SCIENCE RFCS project. I would like to thank all of the partners of the project for their discussions and reviews of my work.

Finally, I would like to acknowledge the support of my family. I would particularly like to thank Dr Robert Francis and Dr Natalie Francis for their motivating advice from their own fields. I would like to thank Antonia Pilpilidou, both for support and proof-reading of the various drafts.
Declaration of originality

This thesis and the work to which it refers are the results of my own efforts. Any ideas, data, images or text resulting from the work of others (whether published or unpublished) are fully identified as such within the work and attributed to their originator in the text, bibliography or in footnotes. This thesis has not been submitted in whole or in part for any other academic degree or professional qualification. I agree that the University has the right to submit my work to the plagiarism detection service TurnitinUK for originality checks. Whether or not drafts have been so-assessed, the University reserves the right to require an electronic version of the final document (as submitted) for assessment as above.

Signed: [Signature]

Date: 26/03/17
Contents

1 Introduction ........................................... 1
  1.1 Objectives ............................................. 2
  1.2 Methodology & thesis structure ......................... 3
  1.3 Relation to published work ............................ 5

2 Overview and history of SCS panel technology and behaviour 6
  2.1 Key features of the SCS system ............................ 7
  2.2 Failure mechanisms & design .............................. 11
  2.3 Active areas of research in SCS panel technology ........... 13
  2.4 In-plane forces .......................................... 13
    2.4.1 In-plane shear ........................................ 14
    2.4.2 In-plane compression .................................. 15
    2.4.3 In-plane tension ....................................... 16
  2.5 Out-of-plane forces ....................................... 16
  2.6 Accidental limit states ................................... 19
    2.6.1 Fire performance ...................................... 19
    2.6.2 Combined mechanical and thermal loading / pressurised steam
          release .................................................. 20
    2.6.3 Blast & impact ......................................... 21
  2.7 Other considerations ..................................... 22
    2.7.1 Construction / Execution stage ....................... 22
    2.7.2 High local stresses at interface between tie-bars and plates ... 24
    2.7.3 Fatigue ................................................ 27
## CONTENTS

4.4 Modelling using finite element analysis .................................................. 71
  4.4.1 2D vs. 3D Modelling ................................................................. 71
  4.4.2 Elements, meshing & loading ...................................................... 72
  4.4.3 Steel material model ................................................................. 78
  4.4.4 Concrete material model - Damaged Plasticity ............................... 79
4.5 Proposed model for shear connection ...................................................... 92
4.6 Choice of load-slip relationships for studs and tie-bars ......................... 94
  4.6.1 Mechanical behaviour of shear studs ........................................... 94
  4.6.2 Load-slip relationships in the literature ...................................... 97
  4.6.3 Expected differences between load-slip curves for studs used in conventional composite construction and studs used in SCS panels 101
  4.6.4 Push-test evidence for tie-bars ................................................ 102
4.7 Comparison of numerical model predictions with test results .................. 109
  4.7.1 Parametric assembly of models .................................................. 109
  4.7.2 SCIENCE SP1 ............................................................................. 109
  4.7.3 Parametric study of different load-slip curves ............................... 112
4.8 Understanding the distribution of stud forces in the span using finite element analysis ................................................................. 118
  4.8.1 Existing understanding of stud force distribution ............................ 119
  4.8.2 Demonstration case ....................................................................... 119
  4.8.3 Tension face ................................................................................ 121
  4.8.4 Compression face ........................................................................ 127
4.9 Implications of slip discontinuities on design ......................................... 128
  4.9.1 Demonstration case ....................................................................... 129
  4.9.2 Test B3 by Roberts et al. [151] ..................................................... 131
4.10 End-slip and the minimum degree of shear connection ............................ 135
4.11 Increased stud density at supports ......................................................... 139
4.12 Design procedure summary ................................................................. 143
4.13 Conclusion ......................................................................................... 144
## 5 Out-of-plane shear resistance

### 5.1 Existing Eurocode compliant model
- 5.1.1 Panels without shear reinforcement
- 5.1.2 SCS panels with shear reinforcement
- 5.1.3 Specific considerations for SCS panels affecting shear resistance

### 5.2 Analysis of experimental data
- 5.2.1 Comparisons between the resistance formulation and the test results
- 5.2.2 Comparison of strength predictions between SCS panels and RC beams

### 5.3 Shear transfer actions in members without shear reinforcement
- 5.3.1 Dowel action
- 5.3.2 Arching action

### 5.4 Investigation of the effects of shear connection on out-of-plane shear resistance using FE
- 5.4.1 Verification of the FE model
- 5.4.2 Case design

### 5.5 FE study - Panels without shear reinforcement
- 5.5.1 Group 1 - $a/d = 1$
- 5.5.2 Group 2 - $a/d = 3$
- 5.5.3 Group 3 - $a/d = 2$
- 5.5.4 Reduced stud spacing
- 5.5.5 Comparison with the Eurocode model
- 5.5.6 Conclusion

### 5.6 FE study - Panels with shear reinforcement
- 5.6.1 Group 4 - $a/d = 1$
- 5.6.2 Group 5 - $a/d = 3$
- 5.6.3 Group 6 - $a/d = 2$
- 5.6.4 Comparison with the Eurocode model
- 5.6.5 Conclusion
5.7 Alternative resistance models for members without shear reinforcement
  5.7.1 AISC N690-12
  5.7.2 JEAC-4618
  5.7.3 KEPIC-SNG
  5.7.4 fib Model Code 2010
  5.7.5 Comparison between models

5.8 Alternative resistance models for members with shear reinforcement
  5.8.1 AISC N690-12
  5.8.2 JEAC-4618
  5.8.3 KEPIC-SNG
  5.8.4 fib Model Code 2010
  5.8.5 Comparison between models

5.9 Conclusion

6 Prediction of deflection
  6.1 Methodology for comparative study
  6.2 Models for panel out-of-plane stiffness
    6.2.1 Comparisons between methods
  6.3 Results
    6.3.1 Shear deformation dominated tests
    6.3.2 Reduced shear connection stiffness
    6.3.3 Specimens with no-tie bars
    6.3.4 Conclusion
  6.4 Deflection analysis using the finite element method
    6.4.1 Comparison Study Methodology
    6.4.2 Element formulations
    6.4.3 Input of stiffness matrix values into ANSYS
    6.4.4 Translation of stiffness into an effective height and elastic modulus
  6.5 Models for out-of-plane shear stiffness
6.5.1 Comparisons of shear stiffness predictions with shear stiffness of the concrete layer ........................................ 252
6.5.2 Model by Kim and Mander ........................................ 254
6.5.3 FE comparative study - Results & Validation .................. 256

6.6 Non-linear finite element analysis ................................. 260
6.6.1 Implementation of non-linear model in ANSYS ................. 262
6.6.2 Empirical model for loss of flexural stiffness with increased moment .................................................. 262
6.6.3 Results ................................................................. 264
6.6.4 Conclusions ........................................................... 266

6.7 Overall conclusion ..................................................... 267

7 Calibration of Eurocode partial resistance factors ................. 269
7.1 History of Partial Factors ............................................. 270
7.2 Eurocode partial factor derivations in the literature ............... 272
  7.2.1 Concrete structures .................................................. 272
  7.2.2 Steel-concrete composite structures ............................. 273
  7.2.3 Steel structures ...................................................... 274
  7.2.4 Alternative structural reliability methods ....................... 275
7.3 Calibration of partial factor using Eurocode 0 Annex D - Design assisted by testing ........................................... 276
  7.3.1 Example resistance equations .................................... 276
  7.3.2 Theoretical basis .................................................... 277
  7.3.3 Allocation of uncertainty to material or resistance model .... 282
  7.3.4 Weighting factors .................................................. 284
  7.3.5 Variability associated with the basic variables ............... 286
  7.3.6 $V_{X,\text{known}}$ and $V_{X,\text{unknown}}$ .................................. 287
7.4 Application of the procedure to the resistance model for Out-of-plane shear .................................................. 288
  7.4.1 Unreinforced panels in shear .................................... 289
  7.4.2 Reinforced panels in shear ....................................... 290
## Contents

7.4.3 Discussion ................................................. 291  
7.4.4 Limitations of existing method ......................... 292  
7.5 The matrix method .......................................... 292  
  7.5.1 Theoretical basis .................................. 293  
  7.5.2 Representative structures ......................... 296  
  7.5.3 Non-linear resistance functions .................... 297  
7.6 Application of the matrix method to panels unreinforced in shear ........ 301  
7.7 Application of the matrix method to panels in bending ................. 304  
  7.7.1 Calculation of $V_\delta$ and $b$ .................. 304  
  7.7.2 Effect of variability of the basic variables ............ 306  
  7.7.3 Calculation of $\gamma_{M^*}^r$ .......................... 311  
  7.7.4 Comparison between $\gamma_{M^*}^r$ for SCS construction and the $\gamma_{M^*}^r$ for conventional composite construction ....................... 312  
7.8 Conclusions ................................................. 313  

8 Conclusion & further work .................................... 315  
  8.1 Summary ................................................. 315  
  8.2 Research significance and contributions .................. 319  
  8.3 Published work and project reports ..................... 321  
    8.3.1 Peer reviewed papers ............................. 321  
    8.3.2 SCIENCE project deliverables .................... 323  
  8.4 Further research ........................................ 323  
    8.4.1 Verification through beam testing ................ 323  
    8.4.2 Additional testing .................................. 324  
    8.4.3 Push testing of tie-bars ........................... 324  
    8.4.4 Co-existent tension and shear in tie-bars .......... 324  
    8.4.5 Application of the same techniques to in-plane shear ... 325  

Bibliography .................................................... 326  

Appendices ....................................................... 348  

xii
## CONTENTS

A  Python code for modelling an SCS beam using ABAQUS 349

B  Test database in Python code format 400

C  Python code for assembling parameters used for modelling an SCS beam using ABAQUS 560
Nomenclature

\( U_{\text{bending}} \) is the unity factor for bending

\( U_{\text{shear}} \) is the unity factor for shear

\( \alpha_{cc} \) is a factor accounting for long term load effects and stress confinement effects in flexure, usually taken as 0.85

\( \Delta T_{\text{savg}} \) is the average of the maximum surface temperature increases for the faceplates due to accident thermal conditions - equal to 0 in the absence of thermal effects

\( \eta_{i} \) is the mean value of the conversion factor taking into account volume and scale effects, effects of moisture and temperature, and any other relevant parameters

\( \gamma_{c} \) is the partial factor for concrete (generally taken as 1.5 for persistent and transient design)

\( \gamma_{s} \) is the partial factor for reinforcement (taken as 1.15 for persistent and transient design)

\( \gamma_{M,i} \) is the partial factor for variable \( i \), a material property, also accounting for model uncertainties and dimensional variations

\( \gamma_{m,i} \) is the partial factor for the variable \( i \), for a material property

\( \gamma_{Rd} \) is the partial factor associated with the uncertainty of the resistance model

\( \gamma_{V} \) is the partial factor for design shear resistance of a headed stud, taken as 1.25\[^{[29]}\]

\( \mu_{b} \) is the degree of shear connection of the bottom plate

\( \mu_{t} \) is the degree of shear connection of the top plate

\( \mu_{\text{avg}} \) is the average degree of shear connection on the top and bottom plates

\( \Phi \) the damage parameter (between 0 and 1)
\( \rho \) is the element material density

\( \rho_l \) is the longitudinal reinforcement ratio

\( \rho_v \) is the shear reinforcement ratio

\( \rho_{AISC} \) is the reinforcement ratio, given by \((t_b + t_t)/h\)

\( \sigma_t \) is the tensile stress at the point of cracking

\( \theta \) is the effective truss angle, taken as 21.8°

\( \nu_1 \) is a strength reduction factor for cracked concrete in shear = 0.6 \((1 - f_{ck}/250)\)

\( \nu_c \) is the Poisson’s ratio of concrete, taken as 0.2

\( \nu_e \) is the effective Poisson’s ratio, taken as 0.2

\( A_c \) is the area of the concrete per unit width

\( a_d \) is the design values of the geometrical data

\( A_e \) is the effective area of the equivalent element

\( A_s \) is the area of the steel per unit width

\( A_v \) is the shear area of the beam. For a unit area this is taken as \( t \)

\( A_{sw} \) is the cross-sectional area of the shear reinforcement

\( A_{swf_{ywd}} \) is the resistance of the shear reinforcement crossing the inclined shear crack between the loaded areas, as illustrated in Figure 5.10

\( b \) is the bias of the resistance prediction

\( b_{w} \) is the effective width (taken as the full width \( b \) in SCS panels)

\( b_{w} \) is the width of the panel

\( B_1 \) is the connector spacing in the longitudinal direction

\( B_2 \) is the connector spacing in the transverse direction

\( C \) is the axial rigidity of the section

\( c_2 \) is a calibration constant for determining effective flexural stiffness

\( C_{Rd,c} \) is an empirical factor; a lower bound of 0.18 is given by König and Fisher\(^{[96]}\)

\( D \) is the bending rigidity of the section
\( d \) is the effective depth of the section

\( d_s \) is the aggregate size in mm

\( E \) is modulus of elasticity of the element material

\( E_0 \) is the initial stiffness

\( E_c \) is the elastic modulus of the concrete

\( E_d \) is the design value of the effect of actions

\( E_e \) is the effective elastic modulus of the equivalent element

\( E_s \) is the elastic modulus of steel

\( E_{A_{\text{panel}}} \) is the axial stiffness of the panel

\( E_{I_{\text{eff}}} \) is the effective bending stiffness of a panel

\( E_{I_{\text{panel}}} \) is the bending stiffness of the panel

\( F \) is an axial or in-plane shear force

\( f_c \) is the cylinder strength of the concrete

\( f_u \) is the ultimate tensile strength of the stud

\( f_{ck} \) is the characteristic concrete cylinder strength

\( f_{y,b} \) is the yield strength of the bottom plate

\( f_{y,t} \) is the yield strength of the top plate

\( f_{ywk} \) is the characteristic yield strength of the shear reinforcement

\( G \) is the shear modulus

\( G_c \) is the shear modulus of concrete

\( G_s \) is the shear modulus of the steel plates

\( G_{t_{\text{Conc}}} \) is the out-of-plane shear stiffness of the concrete layer only

\( G_{t_{\text{K&M}}} \) is the effective out-of-plane shear stiffness, according to Equation 6.5.4

\( h_c \) is the thickness of the concrete compressive stress block

\( h_{sc} \) is the nominal height of the connector
\( I_c \) is the second moment of area of the concrete

\( I_e \) is the effective second moment of area of the equivalent element

\( I_s \) is the second moment of area of the steel plates

\( j \) is the number of basic variables in the resistance function

\( k \) is a curvature

\( k \) is the size effect factor = \( 1 + (200/d)^{0.5} \leq 2.0 \) (\( d \) in mm)

\( k_s \) is the shear-correction factor, typically taken as 5/6

\( k_{d,inf} \) is the characteristic fractile factor for an infinite number of tests

\( k_{d,n} \) is the characteristic fractile factor, from Annex D Table D2

\( l_0 \) is the characteristic length of the element

\( L_e \) is the effective length of the element. For 2D planar elements (as is typical in this model), this is taken equal to the square root of the area of the element

\( M \) is a moment

\( M_{ud} \) is the maximum moment in the beam

\( M_{Ed} \) is the applied moment on the element

\( M_{Rd,\mu} \) is the moment resistance of a section that fully utilises the shear connection

\( M_{Rd} \) is the moment resistance of a section, given by either Equation 4.1.1 or Equation 4.1.2

\( n \) is the modular ratio of steel compared to concrete - Taken as 13 for long term

\( n \) is the number of shear connectors between the ‘critical cross-section’ and the nearest support

\( P_{Rd} \) is the resistance of an individual shear connector

\( P_r \) is the resistance of an individual shear connector

\( R \) is the resistance function

\( R_d \) is the design value of the resistance

\( r_d \) is the design resistance, usually taken as 3.04 standard deviations below the mean resistance
\( r_k \) is the characteristic resistance

\( r_n \) is the nominal resistance, calculated from a resistance function using the nominal values of the basic variables

\( S \) is an out-of-plane shear force

\( s \) is the slip in mm

\( s \) is the spacing of the shear reinforcement

\( t \) is the effective thickness of the element

\( t_b \) is the thickness of the bottom plate

\( t_c \) is the thickness of the concrete layer

\( t_t \) is the thickness of the top plate

\( u \) is a displacement

\( v \) is the effective Poisson’s ratio of the element

\( V_\delta \) is the coefficient of variation of the error of the resistance prediction

\( V_u \) is the shear in the beam at the location of the maximum moments

\( V_{Rd,C} \) is the design shear resistance of the member without shear reinforcement

\( V_{Rd,max} \) is the design value of the maximum shear force limited by crushing of the diagonal struts.

\( V_{Rd,s} \) is the design value of the shear force sustained by the yielding shear reinforcement

\( V_{Rd} \) is the design shear resistance of the member with shear reinforcement

\( V_{rt} \) is the total coefficient of variation of the basic variables

\( V_r \) is coefficient of variation of the resistance

\( V_{X,i} \) is coefficient of variation of the variable \( i \)

\( w_1 \) is the crack displacement at point 1

\( w_{X,i} \) is the variable weighting factor, as defined in Equation 7.3.33

\( X_{k,i} \) is the characteristic value of a material property
$z$ is the effective depth of the section, normally taken as 0.9 $d$

$\Phi_E$ is the cumulative distribution function of the standard Normal distribution.

$\sigma_E$ is the standard deviation of the effect of actions

$\sigma_R$ is the standard deviation of the resistance
CHAPTER 1

Introduction

Steel-Concrete-Steel (SCS) sandwich panels constitute a robust and effective solution for buildings subject to blast and impact. Typical applications have included shear walls, blast protection, security fencing and core walls\textsuperscript{[24]}. The technology is now being investigated in Europe for use in nuclear power plants (NPP), where reductions in reinforcement requirements and faster construction time present a significant opportunity for cost savings.

Figure 1.1 shows the SCS system being used for the construction of a new nuclear power station in China.

![Figure 1.1: SCS panel technology being used for new nuclear construction, taken from a presentation by Cummins\textsuperscript{[52]}](image)

The performance of steel-concrete-steel (SCS) sandwich panels is defined by a complex relationship between materials with very different mechanical
Chapter 1: Introduction

characteristics. In order to make use of the potential mechanical advantages available through composite action, force must be transferred efficiently between the plates and the concrete core through shear connectors. The behaviour of these connectors, both individually and working in combination, is complex, with the size, spacing, and arrangement all leading to changes in behaviour.

Existing design rules, such as those found in a number of design guides\(^ {[87,97,182]}\), rarely require a detailed check on the effects of changes in the provision of shear connection on each design check. Instead, onerous rules are enacted to make the shear connection overly stiff, reducing the potential for the shear connection to effect the design. This has a detrimental effect on the economy of the panels, since the attachment of shear connectors is the most labour intensive activity associated with construction.

The aim of this thesis is to develop design guidance for SCS panel structures that properly accounts for shear connection behaviour. It is proposed that doing so will allow panel designs that require fewer shear connectors, at the expense of a more complex design. Ultimately, it is envisaged that this design guidance may be included in the Eurocodes, such that it can be employed readily by designers in Europe when designing new nuclear structures. To ensure this is possible, the design rules in this work will be developed to be compatible with the Eurocode design philosophy.

For a new design model to be acceptable for publication in this Eurocode the model must be shown to give the structure being designed an appropriate level of reliability, i.e. a very small probability of failure. Uncertainty in any behaviour is unfavourable for design, since the designer then must make conservative assumptions in order to ensure reliability targets are met, at the expense of economy.

1.1 Objectives

Investigation of existing literature (as discussed in Chapters 2 & 3) shows a gap in the understanding of the effect of shear connection detailing on behaviour of SCS panels, particularly when subject to out-of-plane loads. In most cases, designers and codes make conservative assumptions, to the detriment of economy, but there are a number of instances (as presented in this thesis) where ignorance of true behaviour can lead to unsafe design. It is clear that the SCS panel industry would benefit from a better understanding of shear connection.

SCS panels are subject to a wide range of forces, and fail through a large number of failure modes (as discussed in Chapter 2). To understand the effects of shear connection detailing on all of these areas is beyond the scope of any one thesis. This thesis therefore
focuses on out-of-plane forces, for which the best test evidence is available. However, in developing an understanding of this focused topic a number of techniques have been developed, particularly with regard to finite element modelling and reliability analysis, that have general applicability.

This thesis has the following objectives:

1. Develop modelling techniques and guidance for effective non-linear finite element analysis of SCS structures.

2. Improve understanding of the changes in panel behaviour that occur with changes in the degree of shear connection, based on test and FE evidence.

3. Develop rules to design SCS panels to resist out-of-plane forces that take into account shear connection strength and stiffness, ready for inclusion in the Eurocodes.

4. Demonstrate the procedure for developing Eurocode compatible partial factors for design models where steel and concrete are working compositely.

1.2 Methodology & thesis structure

This thesis contains the following chapters:

Chapter 2 is a background section, describing the history of SCS panel development and the key features of the system. An overview is presented of each of the design checks that must be performed by the designer. The state-of-the-art regarding each check is given, with active researchers and research groups highlighted. The models for the checks presented in existing design codes are also introduced, with differences between each highlighted where appropriate. This chapter includes a justification for focusing on out-of-plane behaviour, while also suggesting topics that require focus by other researchers.

Chapter 3 is a detailed literature review of research pertaining to resistance to out-of-plane loads, which is the focus of the remainder of the thesis. A detailed database of test results is constructed, with the intention of use in further calibration or analysis in later chapters. This review also describes a number of tests that are not included in the database. This exclusion may be as a result of incomplete recording, problems with the test arrangement, or deviation from the form of construction to which the new design rules are expected to apply e.g. extra stiffening to the face plates. These exclusions are justified on a case-by-case basis.
Design for out-of-plane forces requires the designer to check three key design criteria; **bending, shear** and **deflection**. A separate chapter is dedicated to each design check.

**Chapter 4** looks at out-of-plane bending. In many respects bending is a better understood failure mode than out-of-plane shear, but the testing that has been conducted has not been able to explore the entire design domain. Although a large number of tests have been conducted, tests on panels with low degrees of shear connection are found to be insufficient in number to draw any definitive conclusion purely based on testing. Behaviour is therefore explored using a detailed parametric finite element model, whose verification for prediction of cross-section and interfacial slip failure is described.

Using the FE model, the prevailing assumption of a smooth profile of stud force along the shear connection interface is investigated. Discontinuities in the stud force profile can mean that the critical cross-section in bending is misidentified, particularly when the degree of shear connection is relatively low. The implications of this for design are explored.

**Chapter 5** examines out-of-plane shear failure. This failure mode is found to be similar in many respects to failure of conventional reinforced concrete beams. The existing design model from Eurocode 2\(^{[30]}\) for design of RC concrete is found to produce acceptable results, with small SCS specific adjustments that are described.

Tests on panels with low degrees of shear connection are again found to be insufficient to draw definitive conclusions based only on tests. A parametric study using FE is therefore conducted to understand the effect of partial shear connection on shear resistance. Comparisons are then made between the resistance predicted by the Eurocode model and the FE results. Further comparisons are then made against design models from other countries, to understand if any of these models give better resistance predictions.

**Chapter 6** examines methods for prediction of deflection. The focus of this chapter is simplified linear elastic models, since this is particularly important in structural analysis of an SCS structure. Existing models assume effective values for the elastic modulus and second moment of area for this purpose, which either include or exclude the contribution of concrete stiffness. Comparison of the predictions against load-deflection curves from the test database (described in Chapter 3) show that deflection can be predicted using linear elastic analysis in most cases, but the models become inaccurate when shear deformation is large in proportion to the bending deformation (often the case for low shear span to depth ratios) or the degree of shear connection is low. Models are described that allow predictions of deflection to be
made with better accuracy. For designers who wish to undertake non-linear structural analysis, an empirical model is developed for reduction of stiffness with applied moment, for panels with lower degrees of shear connection.

Chapter 7 describes the process for deriving Eurocode compliant partial factors for a given resistance model. The theoretical basis of this derivation is described. Partial factors are then derived for out-of-plane bending and shear. An extension to the existing method for assessing partial factors is then proposed, with the aim of addressing limitations in the existing procedure when the contributions of different material to the resistance changes across the domain of assessment. Clarity of the procedure is improved when presented in matrix form.

Chapter 8 concludes the work. In addition to summarising the main arguments and contributions of the thesis. This chapter also describes the relationship between this thesis and a number of publications. A number of areas for further investigation are suggested.

1.3 Relation to published work

The work presented in this thesis was undertaken as part of a European research project, funded by the 'Research Fund for Coal and Steel (RFCS)'. The project title was 'Steel-Composite Sandwich Panels for Industrial, Energy and Nuclear Construction Efficiency', giving the acronym SCI\(\text{ENCE}\)^[4]. Much of the work presented herein was initiated in response to the requirements of the project. However, exploration of the effects of changes in degree of shear connection was not a central aim of SCI\(\text{ENCE}\), meaning the work to improve understanding in this area is unique to this thesis. The work presented in this thesis represents a contribution over and above the work required for the SCI\(\text{ENCE}\) project.

In addition to the project reports required for SCI\(\text{ENCE}\), a number of papers have also been prepared for submission to peer-reviewed journals. The work published in relation to this thesis is discussed in more detail in Section 8.3.
Overview and history of SCS panel technology and behaviour

Steel-concrete-steel (SCS) sandwich panels are a composite system, utilising steel plates in place of reinforcement. The SCS system potentially offers a strong and robust system for constructing walls and floors, with considerably reduced construction time over conventional reinforced concrete walls (see Section 2.1).

The history of SCS technology is difficult to trace, since a number of research groups and authors of papers have claimed to have independently conceived the system.

Technology with many of the characteristics of SCS technology appears as early as 1957\cite{38}. Japanese researchers\cite{111} later describe a similar system for construction of ice-resisting off-shore structures. The lack of further research suggests the ideas did not gain much traction.

After a gap of a number of years, Oduyemi and Wright\cite{131} describe conceiving double skin composite construction for use in submerged tube tunnels. Many pilot tests were undertaken which showed the viability of this solution across several potential uses. However, it was recognised that further research and development was required before the system could be widely utilised.

Sandwich panel technology has only had limited appeal to designers in the UK, since much of the efficiency of the system is derived from automated factory production, which is only cost effective for large orders. Recent announcements about renewal of ageing nuclear infrastructure in the UK\cite{193} have refocused researchers on SCS panel technology, since order sizes for nuclear construction will be large enough to justify investment in efficiency. Interest from the nuclear industry has led to a revival in research interest, with a great many of the studies discussed in the literature review being published between 2010 and the present day.
The key benefit of SCS panel technology over conventional reinforced concrete in nuclear applications is the limited amount of on-site reinforcement fixing required. Nuclear structures are required to resist a much more onerous set of load requirements than any other structure, which means that dense reinforcement is often required. The need for reduction of reinforcement congestion has been cited as a key lesson for developers of the latest generation of nuclear plants

2.1 Key features of the SCS system

The SCS system is characterised by steel plates sandwiching a concrete core, with some form of connector system to achieve composite action between the two.

In nearly all SCS systems composite action is achieved through the use of conventional headed shear studs. Some designs include full depth shear studs, which can also be considered as shear reinforcement (as shown in 2.1 b & c). Shear studs may be greater than half of the depth of the section to as low as a tenth of the depth.

Matsuishi and Takeshita also investigated a number of other systems for developing shear connection, such as angles. However, no more discussion of these systems is found in further work, suggesting they were quickly discounted for efficiency reasons.

SCS panels are typically characterised by their provision of reinforcement against out-of-plane shear forces. Panels can be either reinforced or unreinforced in shear, though reinforced panels are much more prevalent in recent literature (for the improved constructibility, as described below). Figure 2.1 shows a number of possible configurations of out-of-plane shear reinforcement that can be found in the literature.

![Figure 2.1](image)

Figure 2.1: Four possible configurations of shear reinforcement: (a) unreinforced, (b) long studs on a single face, (c) overlapping long studs and (d) discreetly placed stirrups or tie-bars

Early implementations of SCS technology followed a model for construction sequence
taken directly from reinforced concrete, with temporary formwork used to support the plates while the concrete cures. While this construction method may provide some efficiency improvements, it was recognised that a more efficient construction sequence could be achieved by mechanically fixing the two plates together, such that the panels have enough strength to resist the construction stage forces without formwork.

The cost of formwork is a relatively small portion of the total cost of a given system. The efficiency gains are realised through a considerable reduction in on-site construction time, which frees the area more quickly for follow on trades. Assessment by Experts in the USA in 2004 concluded that the reduction in on-site construction time might be as high as 50%, compared to constructing an equivalent wall using conventional reinforced concrete\[^{155}\]. An image from this report is shown in Figure 2.2.

<table>
<thead>
<tr>
<th>Work Structure</th>
<th>Rebar arrangement</th>
<th>Form work (assembling)</th>
<th>Placing concrete</th>
<th>Form work (removal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RC</td>
<td></td>
<td>Wooden form</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28days</td>
<td>13days</td>
<td>7days</td>
<td>4days</td>
<td>4days</td>
</tr>
<tr>
<td>SC</td>
<td></td>
<td>Steel plate</td>
<td>(welding)</td>
<td></td>
</tr>
<tr>
<td>14days</td>
<td></td>
<td>10days</td>
<td>4days</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2.2: Comparison of construction sequence for SCS panels (SC) and conventional reinforced concrete (RC), taken from report by Schlaseman\[^{155}\].

Researchers, directed by industry, have recognised the importance of this construction sequence for the efficiency of the system. Testing of samples without tie-bars after the year 2000 is rare.

A number of other systems could potentially provide mechanical fixity between the
plates. However, the key problem for constructibility is that access to the space between the plates is restricted. For small panels, less than 500mm wide, this is because it physically impossible for a person to fit in the gap. For larger panels access is physically possible, but many countries strictly regulate working in confined spaces. Welding time in such a space is likely to be restricted (if allowed at all), to avoid the build-up of poisonous gases. Any such delay is to the detriment of the efficiency of the system.

Development of methods to successfully connect the plates without internal access has been subject to considerable thought by researchers and construction professionals. The method preferred by Bi-steel is an innovative friction welding technique, where a tie-bar is placed in a rig that spins the bar at considerable speed, such that the steel at either end is melted by the frictional heat developed by contact between the plate and bar\[36\]. Sufficient weld quality has been demonstrated by testing. The solution has proven to be successful, but it is likely that patent issues will restrict its use to the UK. Figure 2.3 shows a Bi-steel panel. The characteristic collar produced by the friction welding process is clearly visible.

![Figure 2.3: Image of a ‘Bi-steel’ panel, taken from an article by Burgan\[36\]](image)

Other, more conventional, means may be employed to achieve the connection between bar and plate. One possible method that is used by researchers is to cut holes in the plate, and then weld from the outside. This method is successful, but suffers from the need for an additional fabrication process (punching holes), that inevitably leads to increased cost. Research also continues on bolted tie-bars.

A completely different system is suggested by Yan et al.\[198\]. The system utilises ‘J-Hook’ connectors, which are arranged to interlock when brought to site.
CHAPTER 2: OVERVIEW AND HISTORY OF SCS PANEL TECHNOLOGY AND BEHAVIOUR

The ‘J-Hook’ system has the key advantage of removing the confined space entirely, since all of the welding of the ‘J-Hooks’ can be performed when the plate is lying flat. However, this system is considerably less robust, since the ‘J-Hooks’ can only generate resistance when being forced apart, and therefore only work well once concreting has started. As shown in Figure 2.2, there may be a considerable portion of the construction sequence when this condition is not met i.e. when adjacent panels are being welded together, before the wet concrete is forcing the panels apart. Construction stage stability problems will likely preclude this system from ever being employed outside of a research environment.

Another radical departure from the prevailing systems is the ‘Steel-Brick’ system, owned by Cauntan Engineering\textsuperscript{[130]}. This system makes use of folded C-shaped plates, arranged back-to-back. All of the welding in this system can be performed from the outside, again removing the need for any confined space working.

Figure 2.4: Image of the J-Hook connector system, taken from paper by Yan et al.\textsuperscript{[198]}

Figure 2.5: Image of the ‘Steel-Brick’ system, taken from article appearing in NSC News\textsuperscript{[130]}

The system is particularly robust, since the webs of the steel provide significantly more
steel to resist out-of-plane shear forces than would be found in an equivalent tie-bar system. However, considerably more on-site welding is required in this system, since the maximum spacing of the ‘webs’ that determines the size of the C section is limited by the buckling resistance of the outer steel plates, which might be at most 300mm (a typical limiting ‘spacing to plate thickness ratio’ \(s/t\) for S355 steel is 30\[^4\], meaning a 10mm plate is required for a 300mm C section). This means 6 times as long a weld length is required than would be needed for a 1.8m module, which is readily achievable using the typical tie-bar systems.

It remains to be seen if the system gains any traction, though it has gained considerable research & development funding. A key issue limiting adoption of system may be that the extra welding required will cause too much of a delay in construction time and increased cost to make the system viable against other alternatives. It is possible that greater efficiency could be gained by increasing the size of a C, perhaps through the use of stiffeners welded from the outside to increase the buckling resistance of the unsupported length of plate, though again this requires more welds. More constructibility and cost study data is required in order to determine whether the conventional SCS system or the ‘Steel-Brick’ system is most efficient.

### 2.2 Failure mechanisms & design

Design of an SCS panel requires a large number of design checks. In addition to verification of the resistance to the applied forces, the shear connection must be detailed in such a way that the assumption of composite action remains valid in all situations.

Depending on the application, panels may be loaded in-plane or out-of-plane, or a combination of the two. Accidental limit states, such as fire or explosions, also introduce temperature effects, again in combination. The degree of criticality of these forces depends on whether the panel is utilised as a wall or a floor; in-plane forces tend to be more critical for walls, while out-of-plane forces are more critical for floors.

SCS panel design is beyond the scope of general purpose concrete or composite design codes, such as Eurocode 2\[^{30}\], Eurocode 4\[^{29}\] or AISC 360\[^{182}\]. Instead, designers are typically compelled to follow one of a number of design guides, which give specific guidance on checks that must be performed to ensure a robust design.

The best guidance available in the UK is the Bi-Steel manual, although this is specific to a patented system\[^{24}\]. At the time of writing, generic guidance for design of SCS
panels that covers the whole of Europe is being prepared, and is expected to be available in late 2018\cite{4}. Several other countries, such as the USA (AISC N690 Appendix N9\cite{1182}), Korea (KEPIC-SNG\cite{97}) and Japan (JEAC 4618\cite{87}) have developed design codes in recent years.

As with any design code, the exact implementation of design checks varies, due to differences of design philosophy and differences in expert opinion. However, all of the design codes contain guidance for checking the following limit states:

- In-plane shear
- In-plane compression
- In-plane tension
- Out-of-plane shear
- Out-of-plane bending
- Thermal loading (usually accidental)
- Combinations of forces and thermal effects

The forces a typical SCS wall element might be subjected to are shown in Figure 2.6.

*Figure 2.6: Design forces for a typical SCS panel element*
2.3 Active areas of research in SCS panel technology

While codes and standards have a relatively long gestation period, research on SCS panel technology has continued to be conducted and published. Emphasis on nuclear applications has generally come to the forefront, with most papers now referring almost exclusively to use in nuclear applications. This is in contrast with older papers that typically emphasised applications such as immersed tunnels and blast walls. SCS panels are a relatively niche technology, requiring specialist skills to investigate and understand test results, meaning research has tended to be concentrated in a small number of research groups.

The most prolific group is based in the USA, led by Varma. Varma’s group has published papers on nearly every aspect of SCS panel behaviour. Research in the UK was abundant, but has slowed in recent years. Key research was sponsored by Corus Bi-steel, with researchers including Wright, Oduyemi, Foundoukos and Bowerman. Research was also conducted at Southampton University, though this was mostly looking at the effect of missile impact for defence applications.

The most recent work on SCS panels in Europe was undertaken as part of the SCIENCE project. This project included a range of work packages, including testing and analysis. This thesis is a partial contribution to this project.

The vast majority of recent research on SCS panels has been presented at the ‘Structural Mechanics in Reactor Technology’ (SMiRT) conference series. Details can be found at https://www.iasmirt.org/. Japanese and Korean research in particular is mostly sourced from this conference, perhaps because full publication of the results tends to be in non-English language publications, which are not indexed in the west. It is likely that the next SMiRT conference in 2017 will include a considerable amount of SCS panel research, including research conducted as part of the SCIENCE project.

The remaining sections of this chapter present a review of literature relating to SCS panel technology, characterised by failure mechanism / design check.

2.4 In-plane forces

Three in-plane forces can be manifested in SCS panels; in-plane shear, in-plane tension and in-plane compression.
2.4.1 In-plane shear

In-plane shear forces arise from a number of sources. At the ultimate limit state lateral loads arise from wind, though this load case is rarely critical for determining the proportions of the panel. Accidental loads tend to be more significant, with seismic loading in particular generating large shear forces in the walls, on account of the relatively large mass of the structure. Seismic design criteria are particularly onerous in nuclear design, since the plant must retain the structural integrity required to perform a safe-shutdown even when subjected to the worst conceivable seismic events\textsuperscript{[83]}.

All of the design codes start with the contribution of the steel plates, with the limiting shear force being decided by the force that leads to yield. The models then diverge in their treatment of the concrete contribution; the Bi-steel manual assumes no contribution at all, while the remaining codes include varying degrees of enhancement. The codes tend to assume checking of the shear connection is not needed for in-plane forces, as the minimum stud spacings that are required to ensure plate yield occurs before plate buckling tends to lead to a shear connection that is stiff enough to ensure the steel and concrete act compositely\textsuperscript{[182]}. It is conceivable that some loading arrangements could lead to designs where the assumption of composite action in shear was invalidated, such as closely spaced loads and supports, but investigating this is outside the scope of this thesis.

Research publications on in-plane shear strength are rare, since there are very few labs in the world with the equipment available to produce in-plane shear failure, even at model scale. As stated by Varma et al.\textsuperscript{[185]}:

\begin{quote}
"Conducting pure in-plane shear tests is extremely challenging, and there are only a few laboratories in the world capable of subjecting wall panels to pure in-plane shear loading."
\end{quote}

Two additional test programmes have been undertaken, both in Japan, and are summarised by Takeda et al.\textsuperscript{[172]} and Ozaki et al.\textsuperscript{[136]}.

The lack on data on in-plane performance is a considerable gap in the knowledge of SCS panel performance, since typical designs will utilise a large percentage of the panels specified in resisting in-plane shear forces (i.e. acting as shear walls). However, those tests that have been conducted by Varma have shown that the walls have considerable strength in plane, such that in-plane shear resistance rarely determines the size of members in design. For both of these reasons an investigation of in-plane
shear forces is not included as part of this work. It is recommend that this topic could be revisited in the future if more tests became available.

2.4.2 In-plane compression

In-plane compression forces are typically generated by the self-weight of the structure and equipment. Some accidental load situations, such as a plane crashing into the roof, may also generate compressive forces in the walls.

Design methods for in-plane compression tend to follow guidance for composite columns, which are well established in design codes. Composite column design is covered by Eurocode 4 (EC4-1-1[29]). Panels can fail by cross-sectional failure (for non-slender columns) or by buckling. The rules adapted for SCS design in the SCIENCE design manual[4] adopt these rules, with small changes to the stiffness function. In all of the design codes maximum stud spacing rules are given to limit the possibility of buckling of the plates, meaning the full steel area can be utilised in compression.

SCS panels tend to be utilised in relatively low rise applications, with structures above 5 storeys being rare. SCS panels have considerable resistance to in-plane compression, given such a large area of concrete is typically available to resist the applied compressive stress. For these reasons, in-plane compression resistance is rarely found to be a limiting design criterion in design.

Verifying axial compression resistance requires a number of failure modes to be considered, including global buckling of the compression element, local buckling of the plates (with or without shear connector pull-out), yielding of the plates, failure in the shear connection and crushing of the concrete.

Tests of SC elements under axial loading are available in the literature from Wright et al.[194], Takeuchi et al.[173], Usami et al.[181], Choi et al.[43] and Choi and Han[42].

The main parameters varied in the tests are: spacing of the connectors on both faces; steel area; length of the connectors; and concrete strength. It can be seen that most tests failed by local buckling of the steel plates or connection failure at the interface, rather than global buckling of specimens. Such failure modes indicate the criticality of correctly detailing the shear connection.
2.4.3 In-plane tension

In-plane tension is very rare in SCS panel design. Design cases that can be postulated include loads generated by heavy attachments, which may hang from higher floors. However, given the relative obscurity of this design situation, little effort has been expended in developing design models. Most design codes rely on the effective area of the plates acting alone.

2.5 Out-of-plane forces

Out-of-plane shear and bending occur as a result of forces normal to the face of the panels. In wall construction, accidental loading by explosion is most prevalent. For floors, the weight of equipment is the key generator of out of plane forces. As with more conventional construction, wind and occupant loading may generate out-of-plane forces on walls and floors respectively, but these effects are usually insignificant. Out-of-plane loads are generally small compared to in-plane loads, meaning they are rarely the critical for determining the size and proportions of the panel. However, failure by out-of-plane shear or bending encompasses several sub-failure modes, the failure point of each being affected by the detailing. It is notable that the current design guide, in common with the design guides from other countries, devotes considerably more detail to resistance against out-of-plane loads compared to resistance against in-plane effects.

Figure 2.7 shows a number of the sub-failure mechanisms that can occur when an SCS panel is subjected to out-of-plane forces.
Figure 2.7 highlights five failure modes. For designs detailed with a stiff shear connection bending failure is usually manifested through **yielding of tension plate**, at the point of maximum moment. Designers generally consider this the most desirable bending failure mode, since: (1) steel yield is a ductile failure mode; (2) a material undergoing yield is maximising its potential strength per tonne; (3) the mechanical model for bending failure (plastic cross-section analysis) is well established and accurate. This model is discussed in more detail in Section 4.1.

**Compression plate buckling** occurs when the spacing of studs on the compression plate is large enough to facilitate a buckled shape before compression yielding. The extent to which compression plate buckling is a true failure mode is debatable, as in most instances the concrete core is still able to take most or all of the compression force needed to maintain cross-section equilibrium. Despite this, all the design guides give rules that limit the spacing of studs and tie-bars on the compression plate, with the express intent of removing the possibility of compression plate buckling.

**Concrete crushing** is observed in some tests (for example, see tests by Koukkari and Fülöp, described in Section 3.2.13), but the circumstances required to produce it are unusual in an efficient design; if the compression plate and tension plate are of equal thickness and strength (as is typical), and there is no possibility of compression plate buckling, then plastic analysis principles dictate that tension plate yield will occur before simultaneous failure of the compression plate and the concrete. Tension plate failure is observed in the vast majority of tests that fail by bending. Those tests that show concrete crushing failure tend to have stud spacings that allow compression plate buckling, which would not be allowed by any of the design guides.

It should be noted that concrete crushing is more prevalent in half-SC construction,
where there is no compression plate, but this construction method is outside of the scope of this thesis. Further detail on half-SC construction can be found in the SCIENCE Design Guide[4].

All three of the failure modes above can only manifest themselves when there is sufficient shear connection to mobilise the full strength of the materials. If this is not the case, the shear connection will fail before the materials reach yield.

Shear connection failure is considered a brittle failure mode, due to the possibility of ‘un-zipping’ of the shear connection (discussed in more detail in Section 4.2). As such, most of the codes seek to avoid any form of interfacial shear failure. The way in which this is achieved varies considerably in each of the design manuals. A key argument of this thesis is that these rules do not ensure full shear connection in some circumstances, which can lead to unsafe design. This is discussed in more detail in Section 4.8.

The final out-of-plane failure mode that must be considered is out-of-plane shear failure. Out-of-plane shear failure manifests through large concrete cracks, appearing suddenly, that result in a catastrophic loss of resistance. Out-of-plane shear is the most brittle of all of the out-of-plane failure modes.

The mechanics of beam shear failure are not well understood, even for conventional reinforced concrete. Debate has arisen in conventional RC construction about the extent to which design parameters, such as the percentage of reinforcement, affect the overall resistance, with research continuing to be active[62]. Given the majority of the models for out-of-plane shear failure for SCS panels are imported from conventional RC construction, it is inevitable that the models in each of the SCS design codes are also very different, particularly in their treatment of arching action. Existing models tend to be conservative, especially for loads applied close to the support. Out-of-plane shear failure is the focus of Chapter 5.

Out-of-plane forces are more easily produced in laboratory conditions than in-plane forces, meaning considerably more researchers are able to investigate panel resistance to out-of-plane forces.

Tests in Europe and the USA tend to be in simply supported single span configurations, sometimes with a single point load, but often with two point loads. Tests in Japan and Korea have tended to include continuous configurations, such that hogging resistance is also critical. It is not immediately clear why these tests tend to be arranged this way, but it is likely to be due to the greater influence of seismic forces on design.

Out-of-plane failure is the focus of this thesis, meaning a more detailed literature review is warranted; this can be found in Chapter 3.
2.6 Accidental limit states

In addition to the normal limit states, a number of other limit states must be checked. These checks are particularly prevalent in nuclear applications, which due the nature of the fuel and generating process are capable of creating load cases outside of the normal conditions that a structure might be subject to.

2.6.1 Fire performance

Thermal actions manifest themselves in a number of accidental scenarios. In addition to fire (a concern for all buildings), structures in nuclear applications may be subject to a number of thermal load scenarios specific to the use of radioactive fuel.

With regard to fire, the structure must be capable of maintaining at least its stability for a period that allows the fire to be extinguished by fire-fighters or its fuel source is spent. Effective fire design also ensures ‘compartmentation’, which means that the fire is contained within a compartment at its point of ignition, to prevent spread into other areas. ‘Compartmentation’ is key to all fire design, including nuclear applications. Safety Guide NS-G-1 from the International Atomic Energy Agency (2004) states:

“Building structures need to be suitably fire resistant. The fire stability rating of the structural elements of a building that are located within a fire compartment or that form the compartment boundaries should not be less than the fire resistance rating of the fire compartment itself.”

Although direct comparison is difficult, an SCS panel might be expected to be affected by fire more than an equivalent reinforced concrete wall. This can be attributed to the following features:

1. The steel in the SCS system is directly exposed to the fire. The steel temperature can be expected to follow the temperature of the fire closely, since steel has relatively high conductivity and low specific heat, compared to concrete. In conventional reinforced concrete structures the reinforcement is embedded within the concrete, which provides insulation from the fire temperature. Temperatures may be halved within 30mm of the exposed surface\[^{26}\]. Strength is proportional to temperature (although the relationship is not linear), meaning hotter structures fail faster.

2. Much of the strength of the system is achieved through the shear connection,
which is again exposed to the hottest temperatures of the fire. Shear connectors are equally affected by high temperatures.

3. The temperature differential across the section can lead to thermal expansion on the exposed surface of the panel. This difference in expansion can lead to out-of-plane moments.

Each of the design guides has recommendations on design for fire performance. The rules generally guide the designer towards analysis by first principles, where thermal time-history is used with a mechanical analysis to calculate a time-resistance history for the structure. This calculation is considerably more complex than the analysis for ambient conditions, since parameters like the plastic neutral axis position will move as the temperature increases. The design guide[^4] has considerably more involved rules, but these are mostly simplifications of the first principle analysis.

Generally, non-combustible materials are used wherever possible in nuclear power plants. Furthermore, the number of possible sources of ignition is kept to an absolute minimum. The design and construction of each plant system should, as far as it is practicable, ensure that its failure does not cause a fire.

SC structures are somewhat more vulnerable to fire than RC structures, since the steel is exposed directly to the fire. Varying temperatures across the SC cross-section can also cause differential expansion, which may cause high additional stresses that can lead to premature buckling of the plates.

Fire performance has been studied by Moon et al.[^119], Kim et al.[^94] and Lu et al.[^110]. Despite the presence of the steel on the outside of the structure, which is therefore directly exposed to the fire, the fire performance of SCS panels is generally found to be adequate for typical fire resistance requirements.

### 2.6.2 Combined mechanical and thermal loading / pressurised steam release

Both nuclear reactors and nuclear waste management facilities are capable of generating accidental scenarios that are considerably more onerous than those allowed for in typical structures. Containment of radioactive material within a confinement structure requires that the structure remain airtight, which means the large deformations of cracks often allowed in fire design may not be appropriate in nuclear design.

A design scenario unique to nuclear design is a loss of coolant accident (LOCA), which can occur in spent fuel storage when flow of coolant around the spent fuel is restricted,
leading to boiling of the cooling water. This scenario was critical in the Fukushima Nuclear Accident\textsuperscript{[11]}. The LOCA scenario is characterised by a rapid rise in temperature, up to around 170°C. This temperature rise is accompanied by a large pressure, due to the thermal expansion of the coolant during its phase change from liquid to gas. Throughout this accident, the structure must remain airtight, else the radioactive coolant will be released into the atmosphere. It may be many hours before flow of coolant can be restarted. The LOCA scenario was the focus of a work package in the SCIENCE project\textsuperscript{[121]}. Rules are also included in a number of US design standards, though specific rules for SCS panels are not included in AISC N690 Appendix N9\textsuperscript{[182]} at the present time.

Use in nuclear applications can lead to a number of unusual load cases. One of the most critical is a pressurised steam release, where rupture of a coolant carrying pipe leads to a combination of high pressure (a mechanical load) and temperature. Resistance to this particular load case can often be limiting criteria for sizing of the element, and has therefore begun to be studied in more detail as emphasis on nuclear applications has increased.

Varma’s research group has conducted a number of studies on combined mechanical and thermal loading. In a study by Booth and Varma\textsuperscript{[22]} two identical 22ft. beam type specimens were tested under four point bending combined with different thermal loading in each case. It was found that the behaviour and flexural stiffness can be effectively modelled using mechanical analysis, but temperature induced deformations were more difficult to predict accurately. Additional numerical work was also presented\textsuperscript{[187]}.

A collaborative project in Japan has led to a large amount of testing, presented at SMiRT 23. A series of papers discusses these results\textsuperscript{[80,95,124,133,134]}.

2.6.3 Blast & impact

Projectile impact is a load case that can occur in a number of SCS panel applications. Much of the research that has been conducted in this area is aimed towards defence applications, such as terrorism resistance, but projectiles may be generated in other accidental scenarios, such as high-pressure stream releases or pipe ruptures.

SCS panels have shown good performance against blast and impact. The system is robust, but also shows high levels of ductility, which allows post-failure mechanisms to develop, such as catenary action. Large impact loads, including aircraft collision, may be assessed using static resistance formulations, perhaps with improved material
properties to account for high strain-rate strength improvement.

A thesis by Sohel\cite{170} describes in detail many of the aspects of design for impact resistance. A series of drop weight impact tests were performed. Additional publications also describe these tests\cite{108,171}. The tests showed significant damage, with concentrated areas of high deformation and plate tearing. Integrity of the back plate was maintained.

Mizuno et al.\cite{117} performed a series of tests that modelled the conditions of aircraft impact. The results suggest that an SCS panel will perform better than an equivalent RC panel, on account of increased ductility. The failure modes presented suggest the SCS panel failed via a ductile bending mode, which maintained integrity due to the tensile membrane action in the steel. This is opposed to the concrete structure, which suffered a brittle ‘punch-through’ failure.

Remennikov et al.\cite{149} performed a falling mass test, on a non-composite panel. The paper suggests that tensile membrane action allows for significant system resistance to be achieved even after extensive cracking of the concrete core.

Bruhl et al.\cite{34} describe a large literature review of both SCS panel tests and impact tests on conventional reinforced concrete structures. The authors find that SCS panels are effective against missile impact.

2.7 Other considerations

2.7.1 Construction / Execution stage

As explained in Section 2.1, much of the efficiency of the SCS system is derived from its reduced construction time. To achieve this, the panel must be capable of remaining stable in its un-concreted state, and must also resist the loads applied to it during the concrete placing and curing phases. The loads generated in a typical concrete pour can be significant, and in many cases determine the size of the plates and the layout of the connecting tie-bars\cite{179}.

Each of the design guides devotes considerable attention to the construction stage checks. Some of the key checks at the construction stage are described below.

Firstly, the panel must be checked for stability during lifting and placement in an un-concreted state. The most efficient construction sequence involves off-site fabrication of the un-concreted structure, which must then be lifted into place using a crane. The SCIENCE Design Guide\cite{4} includes an explicit method for calculating a
stiffness prior to concreting, while the other design guides include more general statements about the need for stability to be ensured throughout the construction process.

Once the panel is in place, the system must be capable of resisting the loads generated by the concrete. In the first instance these loads are hydrostatic, since concrete that is not cured flows as a liquid. The final design pressure is higher however, as the design pressure must also account for the dynamic force exerted by the concrete falling from height. This problem has been widely studied, since the pressures are the same as those exerted on the shuttering used to construct conventional reinforced concrete structures. CIRIA Report 108\cite{47} is widely referred to for this purpose, and is recommended by the SCIENCE design guide\cite{4}.

Once the pressures have been determined, the designer must ensure that the pressures are capable of being resisted. There are two modes of failure that must be checked; excessive stress in the tie-bars and weld/bar interface, and excessive deflection of either faceplate.

The interaction between the stresses generated at the construction stage and the performance at the ultimate limit state is not clear in many cases. It can be expected that locked in stresses in the tie-bars may affect their load carrying capacity when loaded to their yield or ultimate stresses, as can occur when panels are subject to out-of-plane shear.

No specific testing of unconcreted properties of SCS panels has been discussed in the literature. It is therefore not possible to say with certainty what the limits of construction stage stability may be, particular with regard to maximum pour heights etc. However, it is not reasonable to suggest that there is no evidence of inherent constructibility, since those tests that have been performed by researchers have not attributed failure to construction reasons. Connection tests, such as those performed by Müller et al.\cite{120}, have shown adequate strength and stability for a 1m pour height, in a wall configuration.

Work by Zhang et al.\cite{199} explores the importance of faceplate ‘waveiness [sic]’ on the buckling resistance of the plates when loaded in the fully composite condition. The publication gives faceplate deflection limits that result in a limited effect on plate buckling resistance. These limits are incorporated into AISC N690\cite{10}. However, no guidance is given as to how the faceplate deflections can be calculated, or to mitigation strategies in cases where the deflections might be significant.

Section 2 of the Korean code (KEPIC-SNG\cite{97}) concerns the construction stage. A
specific model is given for the fresh concrete pressures, but no guidance is given on limiting values of deflection or resistance. The code specifies that all steel must be kept below the yield strength of the material, though the designer is left to determine the failure modes to which this check is carried out. The Japanese code [87] makes no reference to the construction stage, other than compelling the designer to select a plate size; as given in Section 3.1(1): “the integrity of steel plate as formwork and concrete filling performance during the placement of concrete shall be considered [when determining the plate size].”

The only significant publication to provide comprehensive guidance for the constriction state was performed as part of work package 8 of the SCIENCE project. The project has developed a number of reports, each of which provides guidance that appears in the final design guide [4].

In the absence of testing, many of the checks required have been investigated and verified against evidence from finite element models. In many cases, these checks are likely to be very conservative. A key example of this analysis, which was determining the plate size in the initial reference design, is described below in Section 2.7.2.

### 2.7.2 High local stresses at interface between tie-bars and plates

One of the key problems identified during the design of an example building (performed as part of the SCIENCE project [179]) was the build-up of high local stresses at the weld between the tie-bars and the plate, as a result of the fresh concrete pressures. Figure 2.8 shows an image of the Von-Misses stress at the tie-bar weld, as a result of fresh concrete pressure.
Figure 2.8: Image of high local stresses at the weld between a tie-bar and the plate, as a result of concrete pressure, taken from report by Francis and Aggelopoulos [65].

Figure 2.9 shows the stress profile across the bar, at the surface of the plate, for a typical combination of plate thickness, bar diameter and concrete pressure (which is mostly a function of pour height). It can be seen that the peak stress from the elastic model, of around 400N/mm$^2$, is much larger than the average stress across the bar, which is around 100N/mm$^2$. 
Introducing plastic material properties into the model results in the second stress profile shown in Figure 2.9. It can be seen that the stress profile is cropped at the designated yield strength of the material (in this case 235N/mm²). There is no change in the overall stress profile, except that the region close to the weld undergoes plastic straining.

Development of plasticity in this situation is of some concern, since these areas are then prone to crack initiation, particularly under dynamic loads. The extent to which these stresses remain ‘locked-in’ is arguable, since the concrete will absorb water and shrink during curing, which will likely relieve the worst hydrostatic pressures.

After considerable discussion, the SCIENCE consortium agreed that some plasticity should be allowed to develop during the construction stage. The discussion regarding this conclusion is described in Francis and Aggelopoulos [65]. This conclusion was mostly reached due to the limited plastic strains that occur in reasonable designs. The report gives two alternative calculation models.

Although this conclusion was supported by a group of experts, it is important to note that this conclusion is not supported by testing. It is suggested that testing should be conducted as part of a future research program, as discussed in the conclusion.
2.7.3 Fatigue

Metal fatigue failure occurs when a structure is subject to cyclic loading over a long period of time, causing brittle behaviour due to work hardening. Many of the suggested applications for which SCS panels may be suitable are subject to such load cases, meaning fatigue is often a consideration.

Fatigue performance is a key issue for SCS panels utilised in tunnel systems, since they are subject to considerable cyclic loading by waves. Since Bi-steel was originally conceived for use in immersed tube tunnels, much of the fatigue testing that has been conducted is in relation to this system.

The first occurrence of cyclic fatigue testing in the literature is described by Roberts and Dogan [150]. The same work is also described many years later by Dogan and Roberts [56]. 8 full beam tests were conducted, along with a single static reference test. In addition, 6 cyclic push tests were also conducted. The authors conclude that existing methods for assessing fatigue performance, as defined in Eurocode 3, predict the degradation in performance after a large number of load cycles well. No specific adjustments are needed to account for the studs being used in the SCS system.

In later years, Foundoukos et al. [64] also conducted fatigue testing of both beams and tie-bars, through push-testing. The authors in this case conclude that the fatigue life is less than that predicted by BS 5400 [25]. The authors suggest that this occurs due to the coexistent stresses in the connection from both tension and shear. An adjustment to the modelling approach is suggested. It is still found that fatigue performance is more than acceptable for most typical applications.

The research of fatigue performance of SCS panels that has been conducted has shown that they perform adequately in most typical applications. For applications in the nuclear sector fatigue is unlikely to be a sizing design criteria.

2.7.4 Beyond design basis events

It is an accepted reality in nuclear design that not all events can be predicted, and therefore mitigated against. The most recent serious nuclear accident occurred at the Fukushima Daiichi Nuclear Power Station. The plant survived the 9.0 magnitude Tōhoku earthquake, which was the 4th most powerful earthquake since recording
began in 1900\cite{40}. A large earthquake was anticipated by the designers, meaning its effects could be mitigated. However, the designers had not anticipated fully the effect of the accompanying Tsunami, which disabled vital equipment for maintaining the function of the reactors and spent fuel storage ponds, ultimately leading to partial meltdown\cite{69}.

If it is accepted that these unanticipated events may occur, the designer must aim to make the structure as robust as possible, even if the eventual outcome is catastrophic failure. This is typically achieved by maximising ductility; ductile structures are to distribute loads without breaking, with development of plasticity allowing energy from the event to be dissipated\cite{57}.

Design for ductility means maximising the potential for ductile failure modes, such as steel yielding, to occur before brittle failure modes, such as shear. The extent to which this can achieved is debatable, as there are some situations, such as a short spans with high loads, where shear failure is inevitable.

2.8 Shear connection

SCS panels are a composite system, meaning any resistance relies on the two materials (concrete and steel in this case) in the system working together adequately. Composite action is typically developed in the SCS system using shear studs and tie-bars (as discussed in Section 2.1), the spacing and density of which can significantly affect the performance when subject to the design forces.

Full shear connection occurs when the steel plates yield before failure of any shear connectors occur. Conversely, partial shear connection occurs when the shear connectors fail before the steel reaches yield.

Full composite action is typically considered as desirable; steel yield is a ductile, and well understood, failure mode. However, affixing of each shear connector requires welding, which adds to the total cost of the system. In extreme circumstances, it is possible that the density of the shear connectors is such that there is not enough room to allow the welding to occur. In this case, full shear connection cannot be achieved.

Partial shear connection is not necessarily a problem in design, but the changes in stress distribution and additional deformations that occur should be accounted for. However, much of the existing design guidance is sparing regarding shear connection, with guidance not even available for understanding when the shear connection is not sufficient.
Shear connection stiffness was recognised as a key source of uncertainty in the early design guidance. The following text is presented in Section 5.4.5 of the Bi-steel manual[24]:

"Research indicates that the shear flexibility of Bi-Steel is relatively high. For example, a 10/200/10 Bi-Steel beam spanning 4m in a three point bend test will deflect approximately twice as far as elastic bending theory would predict. This additional flexibility is the subject of ongoing research, but is due to a combination of:

1. Slip between the steel plates and concrete.
2. Compression of the diagonal compression struts.
3. Stretching of the bars in Tension (see Figure 2.4.10)."

The effects of changes in the provision of shear connection remains relatively unexplored in recent publications. Most recent research has eliminated the need to consider the effects of reduced shear connection by specifying a dense shear connector layout. Older tests, such as those conducted by Oduyemi and Wright[131] tend to have reduced shear connection, but this is often coupled with a lack of tie bars, which makes them less applicable to understanding the modern SCS system.

Gallocher et al.[71] consider the effects of interface slip on the capacity of SCS column structures. An analytical method is suggested that gives a reduced Euler buckling load, as used in the resistance calculations. However the method is purely based on first principles, and has not been compared to any tests results.

A study into the effects of reduced shear connection has been carried out by Zhang et al.[199]. In this study, the concept of a ‘development length’ is discussed, which is described by the authors as the length the beam containing enough shear connectors that the yield strength of the steel plate can be fully developed. The authors find that beams with large plate thicknesses have development lengths that generally exceed typical spans, which means either the number of studs within the length available must be increased or the plate thickness reduced. The authors argue that the number of studs required makes these designs uneconomic. The conclusion states that panels should be designed such that a development length of three times the wall thickness is allowed, but no guidance is given as to what can be done if the building geometry makes this impossible.

Zou et al.[200] has conducted a study on partial strength. An advanced FE model has been developed and two key failure mechanisms, but no analytical expressions are
given to account for these failure modes in analysis or design.

### 2.9 Justification for focus on out-of-plane behaviour

As the previous sections highlight, there is an incomplete understanding of many areas of SCS panel performance. Further research is needed in virtually all areas.

Impact is of key concern to researchers; pursuing research that has no bearing on practical design is less valuable than research that will affect design or detailing. Impact is difficult to judge, but a measure to which further research may impact SCS design is the extent to which each of the design criteria is critical to the design of a real structure.

SCS technology has not been utilised in a nuclear structure to date, but indicative studies have begun. Design of a mock SCS structure, described by Tuscher [179], shows that resistance to pressures generated by fresh concrete pressure to be the criterion that determines the plate size for most of the structure. This would suggest that research focussed on design for the construction stage would be of most practical use to practitioners.

However, analysis of the report suggests that fresh concrete pressure is sizing for this structure only because it is relatively small, compared with other typical nuclear structures. The walls for the entire building are only required to be 400mm thick, which is far less than the 900mm walls included in the project test program. The DUS building is given as 22m high. The primary containment for the Westinghouse AP1000 reactor design is 82.3m [84].

The final size of the plate given (8mm) is only determined as such by poor weld performance for tie-bars joined to thin plates (discussed further in Section 2.7.2). This criterion is a reflection of practical reality, meaning no amount of research will allow thinner plates.

Panel economy is also a function of the provision of tie-bars. In the design of the DUS, the tie-bar spacing is taken on a grid of 400mm, which is conservative; tests with spacings of 600mm have shown adequate performance [98]. Had the bar spacing grid been further optimised, it is likely that design for out-of-plane forces would have become more critical.

The second measure of suitability for research is the size of the research problem. In the sense of impact, the mock design suggests in-plane shear research is more relevant to design than out-of-plane shear and bending research. However, there is limited
scope for improving knowledge of in-plane shear performance. As described in Section 2.4, in-plane shear strength mostly relies on yield of the steel plates, which is realised even when the spacing between tie-bars is large. As such, there is limited scope for researchers to improve the design rules. This is not the case for out-of-plane failure; strength is much more dependent on the detailing of the shear connection, with changes in loading scenario and shear connector layouts leading to considerable changes in behaviour.

Based on a review of the existing literature, out-of-plane resistance is the area in which the largest uncertainty exists in SCS panel design. Many tests have been conducted in this area (as described further in Chapter 3), but clarity only exists where the shear connection is very stiff. No consensus exists for assessing the degree of shear connection, and the effect of shear connection of out-of-plane shear response is usually assumed to be uncorrelated. In some cases, it can be argued that the rules are un-conservative (see Section 4.8.1). Although the impact of better understanding of the out-of-plane behaviour may not be the most significant route to achieving design economy, improved understanding will ensure that in the cases where out-of-plane loads are critical, the structure does not fail prematurely.
Chapter 3

Out-of-plane shear and bending: A database of tests

Chapter 2 has presented an overview of the key areas of ongoing research in SCS panel technology. From these areas, it has been found that resistance to out-of-plane loads is the area most worthy of further research; this is justified in Section 2.9. The remainder of this thesis deals only with this particular topic.

Considerable research effort has been devoted to researching the effects of out-of-plane loads on SCS panel systems, by a number of researchers and groups. The literature available is reviewed in detail in Section 3.2.

The large body of test evidence available allows considerable understanding of behaviour to be developed, and provides physical confirmation of behaviours that can be used for calibration of both analytical and numerical models. However, the number of parameters that can be varied in an SCS panel design that affect the out-of-plane behaviour means that gaps do exist; these are discussed in Section 3.3.

3.1 Presentation of test results

As work developed on the remaining chapters in this thesis it became clear that assembly of a database of tests, with all of the associated parameters required to fully describe them, would facilitate an enhanced level of rigorous study of the various panel failure modes. Through the use of a number of different programming techniques it became possible to automate the application of new and existing design rules to each of the tests in the database, allowing changes to be made in a time-scale not possible if manual design were employed.

In the first instance, data about each test is stored in an excel spreadsheet. When
needed, this data can then be extracted into other forms, using bespoke sub-routines programmed using the macro language 'Visual Basic for Applications’[190]. For use in ABAQUS (as discussed in Chapter 4), the data was stored a python file[54]. For use in ANSYS (as discussed in Chapter 6), the data was transformed into an APDL form[116].

The parameters required to fully describe a test of an SCS panel subject to an out-of-plane load include:

- Panel height and width
- Plate thickness
- Plate material properties
- Concrete material properties
- Shear connector height and diameter
- Shear connector material properties
- Tie-bar dimensions and material properties
- Stud and tie-bar positions
- Span and load positions

Given the vast number of parameters, it is not possible to reproduce the entire description of the tests in a succinct way in this thesis. Instead, a number of metrics and ratios are presented, such that a simple comparison is possible between tests, without knowing the full details.

The following parameters / ratios are shown:

- **Total section height** $h$ - As in concrete beams, both the shear resistance and the bending resistance of the beam tends to increase with height. Panels less than 400mm in height can be considered model scale tests.

- **$a/d$ ratio** - The $a/d$ ratio describes the ratio of the critical shear span ($a$) to the effective depth of the section ($d$). The $a/d$ ratio is typically presented in concrete beam tests, as it is strong indicator of the susceptibility of the beam to fail in shear rather than bending; the higher the $a/d$ ratio, the more likely bending failure is to occur. Beams with an $a/d$ ratio of less than 2 also tend to be significantly affected by arching action; this is discussed in more detail in Section 5.3.2.
• **Percentage of flexural reinforcement** $\rho_{flexural}$ - Taken as the ratio of the tension plate thickness to the depth of the section. Typical tests have values between 2% and 4%.

• **Percentage of shear reinforcement** $\rho_{shear}$ - Taken as the area of shear reinforcement per metre squared. In the majority of cases this is provided by tie-bars.

• **Shear connection percentage** $\mu$ - The shear connection percentage is calculated in accordance with the rules defined in Section 4.12. Shear connection percentages greater than 100% are possible, though in these cases the plate will fail before the shear connection.

• **Bending unity factor** $U_{shear}$ - The unity factor for bending resistance, calculated in accordance with the rules described in Section 4.12.

• **Shear unity factor** $U_{shear}$ - The unity factor for shear resistance, calculated in accordance with the rules described in Section 5.1.

• $R_{Test}/R_{Design}$ - Ratio of test resistance to design model prediction, indicating the conservativeness of the design rules.

For each test series, a diagram of a typical test is shown. These diagrams are indicative only, with the main aim being to show the load arrangement and shear connector layout.

### 3.2 Database of tests

#### 3.2.1 Casillas García de León et al. (1957)

The earliest known occurrence of a plate being utilised in a composite system as tension reinforcement is found in work by Casillas García de León et al.\(^{[38]}\). This work describes the use of plates as a substitute to reinforcing bar in conventional applications, but particularly focussing on floor slabs. The work explores the use of friction enhancing measures, such as welded wire fabric, but concludes that sufficient shear transfer is only available when embedded studs are utilised. A number of configurations of studs are tested, encompassing a number of different degrees of partial and full shear connection.

The author concludes the work with the development of design rules. Even in this early work, considerable discussion is devoted to the need for sufficient studs to
realise the bending capacity. Rules are given for the maximum stud spacing, though the model scale of the tests conducted mean that these rules are less onerous than would be required for current designs. The author records both load-deflection and load-slip responses for each of the tests he conducts, meaning this work is helpful for calibrating numerical models. These tests are not included in the test database, as they only have a single plate, on the tension side of the beam.

The work by Casillas García de León et al. appears to have been mostly for academic interest, since no further research or demonstration projects can be found that utilise the system for a number of years. However, it does appear that the work provided inspiration to Japanese researchers, who propose the use of a double skin steel-concrete composite system for use in offshore applications that are subject to considerable loads from ice. These papers are the first to discuss a true sandwich panel system.

3.2.2 Matsuishi and Takeshita (1977)

The first Japanese work that discusses the SCS system is by Matsuishi and Takeshita[111]. For this work, the authors conducted 18 tests on a number of different configurations of panels, utilising a number of different systems to provide shear connection, including angles and embedded tie-bars. Most notable is the ST-series, which shows the use of headed studs, in a similar manner to the modern configurations. An image of this system is shown in Figure 3.1.

![Image of ST-Type system](image)

**Figure 3.1: Typical test from the ST-Type series, tested by Matsuishi and Takeshita[111]**

While the ultimate loads of tests have been recorded, no data is given for the material properties or a number of the design parameters, including connector spacing and plate thickness. For these reasons, these tests cannot be included in the test database.
3.2.3 Nojiri et al. (1986)

A second series of Japanese tests is described by Nojiri et al.\cite{129}. This work is described by the authors as an extension to the work by Matsuishi and Takeshita, with the focus of the development again being ice-resisting off-shore structures.

As in the previous paper, the test group includes a number of shear connector types that are not used in later tests by other researchers. A number of the designs utilise plates spanning the entire cross-section, which can offer considerable enhancement to the resistance, but hinder the pouring and compaction of concrete. The sample also includes three conventional RC beams, used as reference cases, that are not relevant to this investigation. Tests #3, #9–#13, #16–#19, #22, #23 and #25–#27 are excluded from the test database due to unusual reinforcement or shear connection arrangements. Of the 27 cases described in the paper, this leaves 11 cases to be included in the database.

Sufficient data is given for the model parameters (material properties, connector spacing etc.) to fully model all of the tests.

The remaining tests are divided into two loading arrangements; five point continuous beams (#1 to #13), and three point simply supported beams (#14 to #27). In all cases the shear span to depth ratios of the tests are relatively small (less than 2), which leads to shear being critical in all of the cases. The crack patterns presented in the paper show evidence for considerable direct arching action, as seen in Figure 3.2.

![Figure 3.2: Crack pattern for test #6, taken from Nojiri et al.\cite{129}](image)

An unusual feature of this tests series when compared to other tests in the literature is the use of closing plates at either end of the sample (as shown in Figure 3.2). Since these closing plates prevent slip of the concrete against the steel, they may be modelled as shear connectors. It is the presence of these end plates that prevents bending resistance limiting the capacity of the design, despite the lack of shear connectors in many of the tests (or the complete lack of connectors in test #1).

Since no other test series in the database includes closing plates, no rules have been developed for their design. The closing plates will be considerably more stiff than the shear connectors, which means the closing plates would likely buckle outwards.
before the shear connectors reach their full bending capacity. There is no suggestion that this type of failure has occurred, which suggests that the plates are of sufficient thickness to resist the longitudinal forces applied. Since end slip is prevented, 100% shear connection must be developed in this instance.

For the purposes of the back-analysis presented in Table 3.1 the end plate is modelled as a shear connector of sufficient resistance to develop 100% shear connection. This supports the assumptions made by the authors in their back analysis, which only allows for plastic failure of the cross-section in bending (as per Equation 4.12.2).

The tests included in this series are shown in Table 3.1.

Table 3.1: Summary of tests by performed by Nojiri et al.\textsuperscript{[129]}

<table>
<thead>
<tr>
<th>Test</th>
<th>$a$</th>
<th>$a/d$</th>
<th>$\rho_{\text{flexural}}$</th>
<th>$\rho_{\text{shear}}$</th>
<th>$\gamma_{\text{bottom}}$</th>
<th>$\gamma_{\text{top}}$</th>
<th>$U_{\text{bending}}$</th>
<th>$U_{\text{shear}}$</th>
<th>$R_{\text{test}} / R_{\text{Design}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>312</td>
<td>0.96</td>
<td>19.2</td>
<td>-</td>
<td>35.5</td>
<td>35.5</td>
<td>0.28</td>
<td>1.00</td>
<td>4.11</td>
</tr>
<tr>
<td>#2</td>
<td>312</td>
<td>1.20</td>
<td>19.2</td>
<td>-</td>
<td>70.9</td>
<td>70.9</td>
<td>0.17</td>
<td>1.00</td>
<td>3.27</td>
</tr>
<tr>
<td>#3</td>
<td>312</td>
<td>1.44</td>
<td>19.2</td>
<td>-</td>
<td>0.0</td>
<td>0.0</td>
<td>1.00</td>
<td>0.60</td>
<td>4.60</td>
</tr>
<tr>
<td>#4</td>
<td>312</td>
<td>1.39</td>
<td>3.70</td>
<td>-</td>
<td>18.3</td>
<td>18.3</td>
<td>0.42</td>
<td>1.00</td>
<td>3.42</td>
</tr>
<tr>
<td>#5</td>
<td>312</td>
<td>1.44</td>
<td>19.2</td>
<td>-</td>
<td>70.9</td>
<td>35.5</td>
<td>0.21</td>
<td>1.00</td>
<td>2.80</td>
</tr>
<tr>
<td>#6</td>
<td>312</td>
<td>1.44</td>
<td>19.2</td>
<td>-</td>
<td>0.0</td>
<td>0.0</td>
<td>1.00</td>
<td>0.60</td>
<td>4.60</td>
</tr>
<tr>
<td>#7</td>
<td>324</td>
<td>1.39</td>
<td>3.70</td>
<td>-</td>
<td>18.3</td>
<td>18.3</td>
<td>0.42</td>
<td>1.00</td>
<td>3.42</td>
</tr>
<tr>
<td>#8</td>
<td>312</td>
<td>1.44</td>
<td>19.2</td>
<td>-</td>
<td>70.9</td>
<td>35.5</td>
<td>0.21</td>
<td>1.00</td>
<td>2.80</td>
</tr>
<tr>
<td>#9</td>
<td>312</td>
<td>1.44</td>
<td>19.2</td>
<td>-</td>
<td>0.0</td>
<td>0.0</td>
<td>1.00</td>
<td>0.60</td>
<td>4.60</td>
</tr>
<tr>
<td>#10</td>
<td>312</td>
<td>1.44</td>
<td>19.2</td>
<td>-</td>
<td>70.9</td>
<td>35.5</td>
<td>0.21</td>
<td>1.00</td>
<td>2.80</td>
</tr>
<tr>
<td>#11</td>
<td>312</td>
<td>1.44</td>
<td>19.2</td>
<td>-</td>
<td>0.0</td>
<td>0.0</td>
<td>1.00</td>
<td>0.60</td>
<td>4.60</td>
</tr>
<tr>
<td>#12</td>
<td>312</td>
<td>1.44</td>
<td>19.2</td>
<td>-</td>
<td>70.9</td>
<td>35.5</td>
<td>0.21</td>
<td>1.00</td>
<td>2.80</td>
</tr>
<tr>
<td>#13</td>
<td>312</td>
<td>1.44</td>
<td>19.2</td>
<td>-</td>
<td>0.0</td>
<td>0.0</td>
<td>1.00</td>
<td>0.60</td>
<td>4.60</td>
</tr>
<tr>
<td>#14</td>
<td>384</td>
<td>1.17</td>
<td>3.13</td>
<td>-</td>
<td>341.7</td>
<td>341.7</td>
<td>0.20</td>
<td>1.00</td>
<td>5.34</td>
</tr>
<tr>
<td>#15</td>
<td>384</td>
<td>1.17</td>
<td>3.13</td>
<td>-</td>
<td>156.6</td>
<td>156.6</td>
<td>0.16</td>
<td>1.00</td>
<td>5.04</td>
</tr>
<tr>
<td>#16</td>
<td>384</td>
<td>1.17</td>
<td>3.13</td>
<td>-</td>
<td>156.6</td>
<td>156.6</td>
<td>0.16</td>
<td>1.00</td>
<td>5.04</td>
</tr>
<tr>
<td>#17</td>
<td>378</td>
<td>1.19</td>
<td>2.38</td>
<td>-</td>
<td>96.8</td>
<td>96.8</td>
<td>0.19</td>
<td>1.00</td>
<td>7.51</td>
</tr>
<tr>
<td>#18</td>
<td>378</td>
<td>1.19</td>
<td>2.38</td>
<td>-</td>
<td>96.8</td>
<td>96.8</td>
<td>0.19</td>
<td>1.00</td>
<td>8.93</td>
</tr>
<tr>
<td>#19</td>
<td>378</td>
<td>1.19</td>
<td>2.38</td>
<td>-</td>
<td>96.8</td>
<td>96.8</td>
<td>0.19</td>
<td>1.00</td>
<td>8.93</td>
</tr>
<tr>
<td>#20</td>
<td>378</td>
<td>1.19</td>
<td>2.38</td>
<td>0.51</td>
<td>135.3</td>
<td>135.3</td>
<td>1.00</td>
<td>0.77</td>
<td>1.38</td>
</tr>
<tr>
<td>#21</td>
<td>378</td>
<td>1.19</td>
<td>2.38</td>
<td>1.05</td>
<td>175.3</td>
<td>175.3</td>
<td>1.00</td>
<td>0.49</td>
<td>1.68</td>
</tr>
<tr>
<td>#22</td>
<td>378</td>
<td>1.19</td>
<td>2.38</td>
<td>1.05</td>
<td>175.3</td>
<td>175.3</td>
<td>1.00</td>
<td>0.49</td>
<td>1.68</td>
</tr>
<tr>
<td>#23</td>
<td>378</td>
<td>1.19</td>
<td>2.38</td>
<td>1.05</td>
<td>175.3</td>
<td>175.3</td>
<td>1.00</td>
<td>0.49</td>
<td>1.68</td>
</tr>
<tr>
<td>#24</td>
<td>378</td>
<td>1.19</td>
<td>2.38</td>
<td>1.57</td>
<td>369.5</td>
<td>369.5</td>
<td>1.00</td>
<td>0.34</td>
<td>1.66</td>
</tr>
</tbody>
</table>
Chapter 3: Out-of-plane shear and bending: A database of tests

<table>
<thead>
<tr>
<th>Test</th>
<th>$a$</th>
<th>$a/d$</th>
<th>$P/\text{crand}$</th>
<th>$P_{\text{shear}}$</th>
<th>$\Delta_{\text{top}}$</th>
<th>$\Delta_{\text{bending}}$</th>
<th>$R_{\text{Test}} / R_{\text{Design}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>#25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#26</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#27</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.2.4 Oduyemi and Wright (1989)

The first presentation of the SCS system in western literature is found in Oduyemi and Wright\cite{131}. Text from the papers at the time suggests the system was developed independently of the Japanese researchers. Burgan\cite{35} gives the date of conception as 1985, therefore pre-dating the work by Nojiri et al..

The independence of the two groups is reflected in the different ways with which shear connection and out-of-plane shear resistance is achieved. Oduyemi and Wright focuses on the use of long studs to provide shear resistance, while Nojiri et al. focus on a number of different systems, including various solutions utilising plates.

Long headed studs are potentially an efficient solution to providing both out-of-plane shear resistance and shear connection to the attached plate. Even at the time of publication the use of shear studs was well established in conventional composite construction, including efficient techniques for stud welding using specialised welding guns. This efficiency meant that long-studs persisted as the dominant connector system through much of the early testing. Back analysis of the tests by the authors shows that existing models for predicting out-of-plane shear resistance of beams with shear reinforcement hold well. However, by 1999 the use of long-studs is supplanted by the use of full-depth tie-bars, welded at both ends, which led to greater efficiency at the construction stage (this is discussed further in Section 2.1). It is notable that lack of formwork is mentioned as a positive attribute in this paper, despite the lack of tie-bars. Perhaps this paper refers to the lack of need for shuttering, even if props are still required.

The paper describes a large parametric study of model scale specimens, utilising 18 different designs. All of the tests in this paper are subject to out-of-plane forces. Four additional full scale prototype tests are mentioned as part of a private internal report,
which unfortunately cannot be found. The designs are split into six test series, designed to investigate a number of design parameters. These are described by the authors as below:

- Series A and E: Effects of overlapping shear studs
- Series B: Effects of steel skin thickness
- Series C: Effects of spacing of top shear connector
- Series D: Effects of the amount of bottom shear connection
- Series F: Effects of low concrete strength

Series D is a particularly interesting series, since it is one of the few available in the literature that includes failure by interfacial slip. Cases like this are particularly useful for calibrating the modelling of shear connection; see Section 4.7.

A typical test from this series is shown in Figure 3.3.

![Figure 3.3: Typical test from tests by Oduyemi and Wright](image)

The tests included in this series are shown in Table 3.2.

### Table 3.2: Summary of tests by performed by Oduyemi and Wright

<table>
<thead>
<tr>
<th>Test</th>
<th>$a$</th>
<th>$a/d$</th>
<th>$\rho_{flexural}$</th>
<th>$\rho_{shear}$</th>
<th>$\gamma_{bottom}$</th>
<th>$\gamma_{top}$</th>
<th>$U_{bending}$</th>
<th>$U_{shear}$</th>
<th>$R_{Test} / R_{Design}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>150</td>
<td>3.67</td>
<td>2.00</td>
<td>-</td>
<td>112.3</td>
<td>155.5</td>
<td>1.00</td>
<td>0.82</td>
<td>0.95</td>
</tr>
<tr>
<td>A2</td>
<td>150</td>
<td>3.67</td>
<td>2.00</td>
<td>-</td>
<td>112.3</td>
<td>155.5</td>
<td>1.00</td>
<td>0.82</td>
<td>1.19</td>
</tr>
<tr>
<td>B1</td>
<td>150</td>
<td>3.67</td>
<td>1.33</td>
<td>-</td>
<td>186.6</td>
<td>93.6</td>
<td>1.00</td>
<td>0.61</td>
<td>1.30</td>
</tr>
<tr>
<td>B2</td>
<td>150</td>
<td>3.67</td>
<td>2.00</td>
<td>-</td>
<td>112.3</td>
<td>93.6</td>
<td>1.00</td>
<td>0.88</td>
<td>1.19</td>
</tr>
<tr>
<td>B3</td>
<td>150</td>
<td>3.67</td>
<td>2.67</td>
<td>-</td>
<td>134.4</td>
<td>74.7</td>
<td>0.90</td>
<td>1.00</td>
<td>1.31</td>
</tr>
<tr>
<td>B4</td>
<td>150</td>
<td>3.67</td>
<td>4.00</td>
<td>-</td>
<td>95.8</td>
<td>74.7</td>
<td>0.58</td>
<td>1.00</td>
<td>1.71</td>
</tr>
</tbody>
</table>

39
Once the tests have been described, the remainder of the paper is devoted to mechanical interpretation of the results. Several other papers from the same authors use the same data.

The paper describes the use of plastic analysis for calculating cross-section resistance, with steel force limited by the total resistance of the studs at the interface. This model is shown to work well for cases with high degree of shear connection, and as such is still used for calculation of flexural resistance in all of the design guidance available.

The bending model is presented in detail in Chapter 4.

For tests with lower degrees of shear connection plastic analysis tends to overestimate resistance, even when the plate force is limited to the capacity of the studs. Oduyemi and Wright suggest that the lack of resistance is due to studs on the tension side being less effective, due to the presence of tensile cracks. A curve fitting exercise is used to reduce resistances, which concludes that 50% stud utilisation on the tension plate and 90% stud utilisation on the compression plate give an acceptable degree of conservatism. However, further analysis suggests that this lack of correlation with the tests is as a result of misidentification of the critical cross-section in bending. The results of these tests are well predicted by models for bending resistance that utilise the correct critical cross-section without a reduction factor on stud-strength. This is
discussed further in Section 4.9.2.

The same authors subsequently released two other papers, based on the same data set, but with updated interpretation. Wright et al.\cite{194} describes the same data set, with the addition of a series of column tests. Three additional tests were also conducted on beams subject to combined axial and out-of-plane forces. This type of arrangement is outside of the scope of this thesis, so is not included in the test database. The 2nd paper in the series, also written by Wright et al.\cite{195}, focusses on the design model. The paper reiterates the use of plastic analysis, but further detail is given regarding the assumptions of concrete contribution. The same strength reduction factor of 50% for studs on the tension plate is given.

3.2.5 Roberts et al. (1996)

After a gap of seven years with little activity, a European funded development project, described by Burgan\cite{35}, appears to have generated a number of publications. One of the test series included in this project is described by Roberts et al.\cite{151}. In many respects this testing is similar to the testing by Oduyemi and Wright\cite{131}, being tested at the same laboratory and employing the same scale specimens. As with the previous work, the specimens were designed to fail via different failure modes, depending on the provision of shear connection and the thickness of the plates.

The paper includes 9 tests. In each case the beam is subject to four point loads, in order to approximate a UDL. This test series is the only series in the database that includes this arrangement. A typical test from this series is shown in Figure 3.4.

![Figure 3.4: Typical test from tests by Roberts et al.\cite{151} (Test B1 shown)](image)

The tests included in this series are shown in Table 3.3.
Table 3.3: Summary of tests by performed by Roberts et al.\textsuperscript{[151]}

<table>
<thead>
<tr>
<th>Test</th>
<th>$\kappa$</th>
<th>$a/d$</th>
<th>$P/\kappa_{cr,nal}$</th>
<th>$\rho_{shear}$</th>
<th>$\gamma_{bottom}$</th>
<th>$\gamma_{top}$</th>
<th>$U_{bending}$</th>
<th>$U_{shear}$</th>
<th>$R_{Test} / R_{Design}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1</td>
<td>166</td>
<td>1.20</td>
<td>4.84</td>
<td>0.92</td>
<td>34.8</td>
<td>18.8</td>
<td>1.00</td>
<td>0.38</td>
<td>1.78</td>
</tr>
<tr>
<td>B2</td>
<td>162</td>
<td>1.23</td>
<td>5.00</td>
<td>0.92</td>
<td>21.5</td>
<td>5.6</td>
<td>1.00</td>
<td>0.35</td>
<td>2.11</td>
</tr>
<tr>
<td>B3</td>
<td>162</td>
<td>1.23</td>
<td>4.98</td>
<td>0.92</td>
<td>10.8</td>
<td>11.2</td>
<td>1.00</td>
<td>0.19</td>
<td>1.84</td>
</tr>
<tr>
<td>B4</td>
<td>166</td>
<td>1.20</td>
<td>4.85</td>
<td>0.92</td>
<td>34.8</td>
<td>11.7</td>
<td>1.00</td>
<td>0.37</td>
<td>2.05</td>
</tr>
<tr>
<td>B5</td>
<td>162</td>
<td>1.23</td>
<td>5.00</td>
<td>0.92</td>
<td>10.7</td>
<td>11.2</td>
<td>1.00</td>
<td>0.18</td>
<td>2.71</td>
</tr>
<tr>
<td>B6</td>
<td>162</td>
<td>1.23</td>
<td>5.00</td>
<td>0.92</td>
<td>10.7</td>
<td>11.4</td>
<td>1.00</td>
<td>0.18</td>
<td>2.98</td>
</tr>
<tr>
<td>B9</td>
<td>166</td>
<td>2.80</td>
<td>4.84</td>
<td>0.92</td>
<td>25.0</td>
<td>13.5</td>
<td>1.00</td>
<td>0.16</td>
<td>1.95</td>
</tr>
<tr>
<td>B10</td>
<td>162</td>
<td>6.94</td>
<td>4.97</td>
<td>0.92</td>
<td>75.0</td>
<td>10.7</td>
<td>1.00</td>
<td>0.17</td>
<td>1.13</td>
</tr>
<tr>
<td>B11</td>
<td>162</td>
<td>6.95</td>
<td>4.93</td>
<td>0.92</td>
<td>37.8</td>
<td>24.9</td>
<td>1.00</td>
<td>0.09</td>
<td>1.35</td>
</tr>
</tbody>
</table>

Of particular interest in this paper is the diligent recording of both load-deflection curves and crack patterns. Both recordings in combination allow additional interpretation of the tests, particularly with regard to the effect of reduced degree of shear connection. Most importantly, test B3 shows evidence of bending failure associated with reduced degree of shear connection at a critical cross-section not occurring at the point of maximum moment. This test, and the implication on design for UDL’s, is described in Section 4.9.2.

3.2.6 Mckinley and Boswell (2002)

In parallel with Roberts et al.\textsuperscript{[151]}, a number of tests were conducted at City University. An overview of the test series is given by Mckinley and Boswell. However, this paper does not comprehensively record much of the required data, including the material properties of the bars and studs or the stud layout. Instead, this data can be gleaned from the thesis of the lead author.

The testing described by Mckinley\textsuperscript{[112]} includes 16 tests. The majority of the tests were conducted using either 8mm or 10mm plate. Concrete strengths are relatively high, with recorded strength generally in excess of 50N/mm\textsuperscript{2}.

The tests include two series. The first test series describes two tests constructed using long studs, in a similar manner to those tested by Oduyemi and Wright\textsuperscript{[131]}. The second series is more notable, as it marks the first test series conducted using tie-bar
reinforcement. The significance of tie-bars to SCS panel performance is discussed further in Section 2.1.

Although tie-bars are added to the system to take construction stage and out-of-plane shear forces, their large size means they provide significant interfacial shear resistance. As a result, the tests by Mckinley and Boswell tend to have degrees of shear connection that considerably exceed the requirements for full shear connection. Given the consistency of the plate thickness in the series, it is therefore not surprising that all of the tests in this series fail at similar load-levels. The spacing of the tie-bars, which is the key variable in parametric study, has little effect on the behaviour observed, since bending resistance at full shear connection is largely unaffected by small changes in bar provision.

A typical test from this series is shown in Figure 3.5.

![Figure 3.5](image)

**Figure 3.5:** Typical test from tests by Mckinley\[112\] (Test City1 shown)

The tests included in this series are shown in Table 3.4.

<table>
<thead>
<tr>
<th>Test</th>
<th>h</th>
<th>a/d</th>
<th>f/(f_{\text{crand}})</th>
<th>f/shear</th>
<th>f(\text{Bottom})</th>
<th>f(\text{Top})</th>
<th>U(\text{Bending})</th>
<th>U(\text{Shear})</th>
<th>R(\text{Test} / \text{R\text{Design}})</th>
</tr>
</thead>
<tbody>
<tr>
<td>City1</td>
<td>220</td>
<td>7.73</td>
<td>4.55</td>
<td>1.23</td>
<td>195.4</td>
<td>195.4</td>
<td>1.00</td>
<td>0.19</td>
<td>1.63</td>
</tr>
<tr>
<td>City2</td>
<td>220</td>
<td>7.73</td>
<td>4.55</td>
<td>1.23</td>
<td>195.4</td>
<td>195.4</td>
<td>1.00</td>
<td>0.19</td>
<td>1.59</td>
</tr>
<tr>
<td>Stud2</td>
<td>220</td>
<td>7.73</td>
<td>4.55</td>
<td>-</td>
<td>193.1</td>
<td>175.6</td>
<td>0.61</td>
<td>1.00</td>
<td>1.96</td>
</tr>
<tr>
<td>Stud2b</td>
<td>220</td>
<td>7.73</td>
<td>4.55</td>
<td>-</td>
<td>193.1</td>
<td>175.6</td>
<td>0.63</td>
<td>1.00</td>
<td>1.99</td>
</tr>
<tr>
<td>City3</td>
<td>224</td>
<td>5.80</td>
<td>5.36</td>
<td>1.23</td>
<td>175.3</td>
<td>208.2</td>
<td>1.00</td>
<td>0.25</td>
<td>1.38</td>
</tr>
<tr>
<td>City4a</td>
<td>220</td>
<td>5.91</td>
<td>4.55</td>
<td>1.23</td>
<td>222.2</td>
<td>266.8</td>
<td>1.00</td>
<td>0.22</td>
<td>1.67</td>
</tr>
<tr>
<td>City4b</td>
<td>220</td>
<td>5.91</td>
<td>4.55</td>
<td>1.23</td>
<td>210.7</td>
<td>250.2</td>
<td>1.00</td>
<td>0.23</td>
<td>1.56</td>
</tr>
<tr>
<td>City4c</td>
<td>220</td>
<td>5.91</td>
<td>4.55</td>
<td>1.23</td>
<td>210.7</td>
<td>250.2</td>
<td>1.00</td>
<td>0.22</td>
<td>1.53</td>
</tr>
<tr>
<td>City4d</td>
<td>220</td>
<td>5.91</td>
<td>4.55</td>
<td>1.23</td>
<td>181.4</td>
<td>224.2</td>
<td>1.00</td>
<td>0.25</td>
<td>1.74</td>
</tr>
<tr>
<td>City5</td>
<td>216</td>
<td>6.02</td>
<td>3.70</td>
<td>1.23</td>
<td>262.5</td>
<td>278.9</td>
<td>1.00</td>
<td>0.20</td>
<td>1.47</td>
</tr>
</tbody>
</table>
A unique feature of these tests over a number of the other tests in the database are the high degrees of shear connection, which considerably exceeds the 100% required to ensure yield of the plates. As a result, the bending failures that are observed are purely plastic, and not influenced by slip.

As the $R_{Test} / R_{Design}$ column shows, the plastic bending model predicts conservative resistances. Examination of the load-deflection curves presented in the thesis suggest that the results are considerably influenced by strain hardening of the plate material beyond the first yield, which the high degree of shear connection allows to be developed. The plastic bending model is typically a good prediction of the point of onset of non-linear load-deflection behaviour.

### 3.2.7 Takeuchi et al. (1999)

In parallel with testing in the UK, work was also progressing in Japan. A test series is described by Takeuchi et al.\cite{174}. This work appears to have little to do with the previous testing by Japanese researchers\cite{111,129}, given the lack of citations to previous work and the differences between the previous test arrangements in the tests conducted as part of this work.

Much of the detail of the samples is difficult to obtain, as the paper is both written in Japanese and relatively short (2 pages long). A number of the tests have been excluded from the database, as the parameters of the tests could not be obtained from the paper with sufficient confidence.

The tests include two series of continuously arranged specimens, with the first series utilising four point loads to produce four peak moments (two sagging and two
hoggings) of equal magnitude, with two points of inflexion. The second series uses two point loads to produce two peak moments (one sagging and one hoggings), with a single point of inflexion.

A typical test from the first series is shown in Figure 3.6.

![Figure 3.6: Typical test from tests by Takeuchi et al.](image)

A typical test from the second series is shown in Figure 3.7.

![Figure 3.7: Typical test from tests by Takeuchi et al.](image)

The tests included in this series are shown in Table 3.5.

<table>
<thead>
<tr>
<th>Test</th>
<th>$\zeta$</th>
<th>$\alpha/d$</th>
<th>$\rho_{flexural}$</th>
<th>$\rho_{shear}$</th>
<th>$U_{bending}$</th>
<th>$U_{shear}$</th>
<th>$R_{Test}/R_{Design}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>225</td>
<td>2.00</td>
<td>2.00</td>
<td>-</td>
<td>9.9</td>
<td>9.9</td>
<td>1.00</td>
</tr>
<tr>
<td>2</td>
<td>450</td>
<td>1.00</td>
<td>1.00</td>
<td>-</td>
<td>9.9</td>
<td>9.9</td>
<td>0.91</td>
</tr>
<tr>
<td>3</td>
<td>450</td>
<td>1.00</td>
<td>1.00</td>
<td>-</td>
<td>54.2</td>
<td>54.2</td>
<td>0.24</td>
</tr>
<tr>
<td>4</td>
<td>450</td>
<td>2.00</td>
<td>1.00</td>
<td>-</td>
<td>39.5</td>
<td>39.5</td>
<td>0.71</td>
</tr>
<tr>
<td>5</td>
<td>600</td>
<td>2.00</td>
<td>2.00</td>
<td>-</td>
<td>14.1</td>
<td>14.1</td>
<td>1.00</td>
</tr>
</tbody>
</table>
The number of tests in the study is such that a wide variety of failure modes are observed in the sample. The $a/d$ ratio is either 1 or 2, both of which are relatively small compared to the tests in the database performed by other researchers. Beams with $a/d$ ratios of this magnitude normally fail via out-of-plane shear, but in this series the bottom plate thickness is also relatively thin, which leads to early onset of bending failure. Of particular interest in this sense are tests #8, #9 and #10, which are identical except for the provision of shear reinforcement. As shown in the paper, the provision of shear reinforcement has little effect on stiffness, but does affect the point of failure; #8 shows a clear shear failure, while #9 and #10 are bending failures.

### 3.2.8 Shanmugam and Kumar (2005)

After another short break, renewed interest in SCS technology from the nuclear sector led to a new wave of testing being performed in the late 2000s. The test series performed by Shanmugam and Kumar\textsuperscript{[158]} is unique, in that it...
considers a 2-way spanning slab, as opposed to the beam test configuration used by other researchers. The load was applied in each case through a single jack at the centre. Load-deflection curves in each of the cases recorded show that failure was ductile, which would suggest that punching shear failure was not critical, as might be expected. This is supported by the images, which show considerable evidence of compression plate buckling.

Analysis of 2-way spanning slabs of this type is not straightforward. While it is possible to establish the points of failure using yield-line analysis, it is not clear how the interfacial shear generated at the cross-section are distributed into the shear connectors. The load-slip curve shown in Figure 6 of the paper suggests that slip in the top (compression) plate is greater than the bottom plate, which is the opposite of what is observed in typical simply supported tests. This suggests that membrane effects in the top plate may be present, for which there is currently no model.

SCS panels are not currently utilised for two-way spanning applications\textsuperscript{[179]}. Development of a design model has not been attempted by any other research group, and does not appear in any of the current design codes. For this reason this test series is considered outside the scope of this work, and is not included in the database.

3.2.9 Foundoukos et al. (2008)

Further testing was performed at Imperial College London by Foundoukos et al.\textsuperscript{[64]}. The same test series was also described in an earlier paper by the same authors\textsuperscript{[197]}. The test series is also described in detail in the thesis of Foundoukos\textsuperscript{[63]}.

The test series presented consists entirely of beams without shear studs. Tie-bars are used to provide all of the shear connection, as well as providing resistance to out-of-plane shear. Such a design is probably unique to the Bi-steel system, due to its utilisation of friction welding to efficiently connect the bars to the plates. Use of any other method, such as bolting or conventional welding, would likely lead to uneconomic designs at the dense spacings used.

A typical test from this series is shown in Figure 3.8.
Figure 3.8: Typical test from tests by Foundoukos et al.\textsuperscript{[64]} (Test BS1 shown)

The tests included in this series are shown in Table 3.6.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|c|c|c|}
\hline
Test & $\rho$ & $\rho_{\text{flexural}}$ & $\rho_{\text{shear}}$ & $\gamma_{\text{bottom}}$ & $\gamma_{\text{top}}$ & $U_{\text{bending}}$ & $U_{\text{shear}}$ & $R_{\text{Test}} / R_{\text{Design}}$
\hline
BS1 & 412 & 1.50 & 1.23 & 186.7 & 186.7 & 1.00 & 0.31 & 1.30
BS2 & 416 & 1.90 & 1.23 & 147.7 & 147.7 & 1.00 & 0.42 & 1.17
BS3 & 418 & 2.80 & 1.23 & 189.8 & 89.2 & 1.00 & 0.34 & 1.07
BS4 & 220 & 3.60 & 1.23 & 147.5 & 89.2 & 1.00 & 0.43 & 1.00
BS5 & 224 & 5.29 & 1.23 & 89.7 & 89.9 & 1.00 & 0.63 & 0.87
BS6 & 224 & 2.84 & 0.82 & 186.7 & 89.2 & 1.00 & 0.36 & 0.99
BS7 & 218 & 3.63 & 0.82 & 146.2 & 89.9 & 1.00 & 0.52 & 0.79
BS8 & 220 & 3.63 & 0.82 & 89.2 & 89.2 & 1.00 & 0.28 & 1.19
BS9 & 224 & 5.32 & 0.82 & 89.2 & 89.2 & 1.00 & 0.36 & 0.99
BS10 & 218 & 2.80 & 0.61 & 189.8 & 88.4 & 1.00 & 0.23 & 1.46
BS11 & 220 & 3.63 & 0.61 & 146.2 & 89.2 & 1.00 & 0.29 & 1.34
BS12 & 224 & 3.33 & 0.61 & 89.0 & 89.2 & 1.00 & 0.39 & 1.04
BS13 & 320 & 3.21 & 1.23 & 118.2 & 119.3 & 1.00 & 0.59 & 1.09
BS14 & 324 & 3.67 & 1.23 & 102.2 & 102.6 & 1.00 & 0.68 & 0.98
BS15 & 320 & 3.13 & 0.82 & 97.2 & 94.8 & 1.00 & 0.57 & 0.89
BS16 & 324 & 3.63 & 0.82 & 82.6 & 82.1 & 1.00 & 0.57 & 0.84
BS17 & 320 & 3.18 & 0.82 & 119.6 & 119.2 & 1.00 & 0.44 & 0.93
BS18 & 324 & 3.67 & 0.82 & 102.1 & 102.1 & 1.00 & 0.52 & 0.75
\hline
\end{tabular}
\end{table}

The authors report a range of failure modes, including cross-sectional bending failure and out-of-plane shear failure. This is despite the calculation models predicting...
bending failure in every case.

A number of the tests failed due to premature failure at the base of the stud, which can be linked to relatively thin thickness of the plate compared to the diameter of the tie-bar. It is notable that the tests with the thinnest plates (BS1, BS4, BS7, BS10) did not fail at the welds, but this can be attributed to the tie-bars not being heavily utilised at the point of bottom plate tension failure. This test series highlights the importance of adhering to the detailing rules given in Section 4.12.

The lack of additional shear connectors mean that many of the tests, including BS6, BS9, BS12, BS15 and BS16 have less than 100% shear connection on the bottom flange. In all of these cases apart from BS6, bar shear is reported as the failure mode, which is consistent with the prediction of failure for a beam with reduced shear connection. Examination of the load-slip curves presented in the thesis of Foundoukos[63] shows the onset of excessive slip in these cases.

Foundoukos et al.[64] reports tests BS5 and BS6 as shear failure. However, further analysis suggests these tests actually fail in flexure. The image presented in the paper shows cracks that would be characteristic of shear failure (reproduced in Figure 3.9).

![Concrete shear failure](image)

**Figure 3.9:** Crack pattern for test BS6, taken from Foundoukos et al.[64]

However, shear failure is characterised by a sudden drop in load carrying capacity at the point of crack formation. Examination of the load-deflection curves in the thesis shows no evidence of such a drop in capacity. BS5 in particular shows a smooth plateau at the ultimate capacity, which is characteristic of failure at the interface i.e. excessive slip. Slips in excess of 3mm are recorded for both cases. It is likely that the tests were ended prematurely in this respect, and that the generally accepted failure slip of 6mm would have been reached had the tests been allowed to continue.
3.2.10 Hong et al. (2010)

Korean testing began in the late 2000s, in order to provide data for their own design code. One such series included in this work is by Hong et al. [82].

The testing by Hong et al. [82] include three different loading arrangements. Two of the loading arrangements are continuous, in a similar configuration to the tests performed by Takeuchi et al. discussed earlier. The final group is tested in simply supported four point bending.

A typical test from this series is shown in Figure 3.10.

![Figure 3.10: Typical test from tests by Hong et al. [82] (Test NR-0R-3S400-4ST shown)](image)

A number of the tests are excluded from the sample due to the presence of I-beam ribs attached to the bottom flange. Such designs cannot be analysed using the existing models, as the enhanced shear connection provided by the friction of the beam with the surrounding concrete cannot be accounted for. It is clear from the load-deflection curves that the presence of these ribs does provide enhancement to the panel resistance. The tests included in this series are shown in Table 3.7.

**Table 3.7: Summary of tests by performed by Hong et al. [82]**

<table>
<thead>
<tr>
<th>Test</th>
<th>$h$</th>
<th>$a/d$</th>
<th>$\rho_{flexural}$</th>
<th>$\rho_{shear}$</th>
<th>$\gamma_{bottom}$</th>
<th>$\gamma_{top}$</th>
<th>$U_{bending}$</th>
<th>$U_{shear}$</th>
<th>$R_{Test}/R_{Design}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>NR-0R-3S400-4ST</td>
<td>500</td>
<td>3.20</td>
<td>1.80</td>
<td>0.19</td>
<td>70.3</td>
<td>70.3</td>
<td>0.60</td>
<td>1.00</td>
<td>1.81</td>
</tr>
<tr>
<td>NRC-0R-4S400-4ST</td>
<td>500</td>
<td>3.20</td>
<td>1.80</td>
<td>0.25</td>
<td>76.4</td>
<td>76.4</td>
<td>0.74</td>
<td>1.00</td>
<td>1.59</td>
</tr>
<tr>
<td>NR-0R-3S200-4ST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NRT-0R-3S400-4ST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 3.7: Summary of tests by performed by Hong et al.\(^\text{[82]}\)

<table>
<thead>
<tr>
<th>Test</th>
<th>(\varepsilon)</th>
<th>(a/d)</th>
<th>(\rho/\varepsilon_{\text{crand}})</th>
<th>(\rho_{\text{shear}})</th>
<th>(\varepsilon_{\text{top}})</th>
<th>(\varepsilon_{\text{bottom}})</th>
<th>(U_{\text{bending}})</th>
<th>(U_{\text{shear}})</th>
<th>(R_{\text{Test}} / R_{\text{Design}})</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-4R-2S400-4ST</td>
<td>500</td>
<td>3.60</td>
<td>1.20</td>
<td>0.13</td>
<td>163.7</td>
<td>163.7</td>
<td>1.00</td>
<td>0.97</td>
<td>1.39</td>
</tr>
<tr>
<td>S-4R-2S400-4ST</td>
<td>500</td>
<td>3.60</td>
<td>1.20</td>
<td>0.13</td>
<td>163.7</td>
<td>163.7</td>
<td>1.00</td>
<td>0.98</td>
<td>1.14</td>
</tr>
<tr>
<td>S-4R-2S600-4ST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S-4R-2S800-4ST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S-0R-2S400-4ST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S-4R-0S-4ST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S-4R-2S400-0ST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The tests are designed with extremely heavy shear connections. In most of the tests, this means that full shear connection is achieved. In the two tests that have two inflection points the spans are particularly short, meaning shear failure occurs before any prospect of excessive slip. No recording of slip is given in the paper.

### 3.2.11 Varma et al. (2011)

The first American research to appear in the literature concerned with out-of-plane load effects is described by Varma et al.\(^\text{[184]}\). Further details, and a number of additional cases, can be found in the thesis of the second author, Sener\(^\text{[156]}\).

The paper describes eight beam tests, five without shear reinforcement and three including shear reinforcement. These tests are supplemented with an additional group, described by Sener\(^\text{[156]}\). All of the tests are simply supported, with either three or four point loading. The tests have relatively small \(a/d\) ratios, meaning most of the tests fail in shear. A typical test from this series is shown in Figure 3.11.
CHAPTER 3: OUT-OF-PLANE SHEAR AND BENDING: A DATABASE OF TESTS

The tests included in this series are shown in Table 3.8.

<table>
<thead>
<tr>
<th>Test</th>
<th>( \gamma )</th>
<th>( \alpha / d )</th>
<th>( \rho_{\text{flexural}} ) %</th>
<th>( \rho_{\text{shear}} ) %</th>
<th>( \Theta_{\text{bottom}} ) %</th>
<th>( \Theta_{\text{top}} ) %</th>
<th>( U_{\text{bending}} )</th>
<th>( U_{\text{shear}} )</th>
<th>( R_{\text{Test}} / R_{\text{Design}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP1-1</td>
<td>457</td>
<td>3.18</td>
<td>-</td>
<td>112.6</td>
<td>112.6</td>
<td>0.58</td>
<td>1.00</td>
<td>1.29</td>
<td></td>
</tr>
<tr>
<td>SP1-2</td>
<td>457</td>
<td>3.18</td>
<td>-</td>
<td>56.3</td>
<td>56.3</td>
<td>1.00</td>
<td>0.91</td>
<td>0.89</td>
<td></td>
</tr>
<tr>
<td>SP1-3</td>
<td>457</td>
<td>3.18</td>
<td>2.08</td>
<td>75.9</td>
<td>75.9</td>
<td>0.58</td>
<td>1.00</td>
<td>1.25</td>
<td></td>
</tr>
<tr>
<td>SP1-4</td>
<td>457</td>
<td>2.48</td>
<td>1.40</td>
<td>78.8</td>
<td>78.8</td>
<td>0.58</td>
<td>1.00</td>
<td>1.43</td>
<td></td>
</tr>
<tr>
<td>SP1-5</td>
<td>914</td>
<td>3.50</td>
<td>1.39</td>
<td>81.0</td>
<td>81.0</td>
<td>0.71</td>
<td>1.00</td>
<td>1.01</td>
<td></td>
</tr>
<tr>
<td>SP2a-1</td>
<td>914</td>
<td>3.50</td>
<td>2.09</td>
<td>59.0</td>
<td>59.0</td>
<td>1.00</td>
<td>0.49</td>
<td>2.09</td>
<td></td>
</tr>
<tr>
<td>SP2a-2</td>
<td>914</td>
<td>2.50</td>
<td>2.09</td>
<td>42.5</td>
<td>42.5</td>
<td>1.00</td>
<td>0.54</td>
<td>2.15</td>
<td></td>
</tr>
<tr>
<td>SP2a-3</td>
<td>914</td>
<td>3.50</td>
<td>2.09</td>
<td>89.5</td>
<td>89.5</td>
<td>1.00</td>
<td>0.55</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>SP2a-4</td>
<td>914</td>
<td>5.50</td>
<td>2.09</td>
<td>59.0</td>
<td>59.0</td>
<td>1.00</td>
<td>0.63</td>
<td>1.50</td>
<td></td>
</tr>
<tr>
<td>SP2c-1</td>
<td>762</td>
<td>2.00</td>
<td>2.51</td>
<td>78.2</td>
<td>78.2</td>
<td>1.00</td>
<td>0.63</td>
<td>1.42</td>
<td></td>
</tr>
<tr>
<td>SP2c-2</td>
<td>762</td>
<td>2.00</td>
<td>2.51</td>
<td>78.2</td>
<td>78.2</td>
<td>1.00</td>
<td>0.63</td>
<td>1.34</td>
<td></td>
</tr>
<tr>
<td>SP2c-3</td>
<td>762</td>
<td>3.00</td>
<td>2.51</td>
<td>117.4</td>
<td>117.4</td>
<td>1.00</td>
<td>0.59</td>
<td>1.31</td>
<td></td>
</tr>
<tr>
<td>SP2c-4</td>
<td>762</td>
<td>3.00</td>
<td>2.51</td>
<td>117.4</td>
<td>117.4</td>
<td>1.00</td>
<td>0.59</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>SP2c-5</td>
<td>1219</td>
<td>2.00</td>
<td>1.04</td>
<td>70.1</td>
<td>70.1</td>
<td>1.00</td>
<td>0.30</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>SP2c-6</td>
<td>1219</td>
<td>2.00</td>
<td>1.04</td>
<td>70.1</td>
<td>70.1</td>
<td>1.00</td>
<td>0.30</td>
<td>0.00</td>
<td></td>
</tr>
</tbody>
</table>

All of the tests in the SP-1 group fail by shear, except SP1-2, which was deliberately
designed with a low degree of shear connection. The load-deflection curve presented for this case is characteristic of such a failure, making it useful for calibration studies (see Section 4.7). Unfortunately, a measurement of end-slip was not presented anywhere in the paper or the thesis.

The remaining SP-2 series show a variety of failure modes. Test SP2-a2 shows a low degree of shear connection, which is again observed in a slow plateauing of the load-deflection curve. SP2-a3 is shows a perfect representation of the load-deflection curve for a panel with a heavy shear connection failing in cross-sectional bending, with an elastic response up to the point of yield of the tension plate, at which a plateau is formed. The only strength enhancement from this point is stain-hardening of the tension steel.

3.2.12 Chu et al. (2013)

Chu et al.\[46\] describes work by a research group in China. The work includes eight tests, loaded in either three or four point bending. Two of the tests in the series are subject to additional tensile forces, but these are excluded from the database. A stiff shear connection is provided, with studs at roughly 150mm centres. The key parameter that is varied between the cases is the shear span length, with the larger span tests tending to fail in bending and the shorter span tests failing in shear.

Shear reinforcement in this series is provided through the use of channel sections. These channel sections have a larger effective area than would be provided by tie-bars, meaning the resistance provided is substantial. However, the channels are placed at 1.2m centres, which is also much larger than the typical spacings used for tie-bars. Since the ratio of spacing to height is much larger than would typically be allowed, this test series provides a good indicator of whether design rules for shear are still appropriate for such a system.

No attempt is made to account for longitudinal shear provided by the angle sections attached to the plates. Given that the sections are small, and that they are aligned with the direction of shear transfer, any enhancement can be expected to be minimal.

A typical test from this series is shown in Figure 3.12.
3.2.13 Koukkari and Fülöp (2013)

Out-of-plane load testing was performed as part of the SCIENCE project[4]. The testing was performed in Finland, and is described in the technical report by Koukkari and Fülöp[98].

The test series covers both bending and shear failure. Tests SP1 and SP2 were deliberately designed to fail in bending. An unusual feature of these tests is that the bottom tension plate is thicker than the top plate. This proves to be important to the behaviour, since an arrangement of this type is susceptible to compression plate buckling before the full resistance of the beam is realised. The layout of studs and tie-bars in these tests results in a relatively high degree of shear connection.
The shear tests (SP3-SP6) were tested at both ends, in an effort to produce more data without the cost of building additional specimens. However, it was realised after the testing was complete that the tests on end-1 had not reached failure in shear. The tests on these ends are therefore ignored in the database.

A typical test from this series is shown in Figure 3.13.

![Figure 3.13: Typical test from tests by Koukkari and Fülöp [98] (Test SP1 shown)](image)

The tests included in this series are shown in Table 3.10.

<table>
<thead>
<tr>
<th>Test</th>
<th>$L$</th>
<th>$a/d$</th>
<th>$\rho_{flexural}$</th>
<th>$\rho_{shear}$</th>
<th>$\gamma_{bottom}$</th>
<th>$\gamma_{top}$</th>
<th>$U_{bending}$</th>
<th>$U_{shear}$</th>
<th>$R_{test} / R_{design}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP1</td>
<td>800</td>
<td>4.50</td>
<td>1.88</td>
<td>0.35</td>
<td>194.4</td>
<td>236.7</td>
<td>1.00</td>
<td>0.53</td>
<td>0.93</td>
</tr>
<tr>
<td>SP2</td>
<td>800</td>
<td>4.50</td>
<td>1.88</td>
<td>0.35</td>
<td>193.5</td>
<td>285.9</td>
<td>1.00</td>
<td>0.53</td>
<td>1.04</td>
</tr>
<tr>
<td>SP3_E1</td>
<td>800</td>
<td>3.75</td>
<td>1.88</td>
<td>0.11</td>
<td>180.9</td>
<td>180.9</td>
<td>0.74</td>
<td>1.00</td>
<td>1.39</td>
</tr>
<tr>
<td>SP3_E2</td>
<td>800</td>
<td>3.75</td>
<td>1.88</td>
<td>0.08</td>
<td>176.7</td>
<td>176.7</td>
<td>0.87</td>
<td>1.00</td>
<td>1.07</td>
</tr>
<tr>
<td>SP4_E1</td>
<td>800</td>
<td>3.75</td>
<td>1.88</td>
<td>0.08</td>
<td>177.5</td>
<td>177.5</td>
<td>0.61</td>
<td>1.00</td>
<td>0.96</td>
</tr>
<tr>
<td>SP5_E1</td>
<td>800</td>
<td>3.75</td>
<td>1.50</td>
<td>0.06</td>
<td>219.6</td>
<td>219.6</td>
<td>0.85</td>
<td>1.00</td>
<td>0.99</td>
</tr>
<tr>
<td>SP6_E1</td>
<td>800</td>
<td>3.75</td>
<td>1.50</td>
<td>0.11</td>
<td>219.6</td>
<td>219.6</td>
<td>0.85</td>
<td>1.00</td>
<td>0.99</td>
</tr>
<tr>
<td>SP6_E2</td>
<td>800</td>
<td>3.75</td>
<td>1.50</td>
<td>0.11</td>
<td>219.6</td>
<td>219.6</td>
<td>0.85</td>
<td>1.00</td>
<td>0.99</td>
</tr>
</tbody>
</table>

The tests in this series all fail as expected. End-slip was measured in this series. However, the degree of shear connection in the designs is such that very little slip would be expected in any failure mode. Slips as recorded tended to be small (less than 0.5mm), meaning the values recorded were subject to considerable measurement
precision error. Of all the tests in the series, only SP3 demonstrated enough slip to be useful for model calibration purposes. This is described in Section 4.7.

Another unusual feature of this test series is that SP2 is constructed using stainless steel, rather than conventional carbon steel. Stainless steel does not exhibit an elastic-plastic stress-strain response, instead showing a rounded hardening phase. Such behaviour means that the response of stainless steel to load can be quite different, particularly with respect to buckling. Guidance for stainless steel design is covered in separate publications\[18\]. In accordance with established procedures, an equivalent elastic-plastic stress-strain curve can be used, with the 'yield strength' taken as the stress at which 0.2% permanent strain remains in the material i.e. the 0.2% 'proof stress'. When this procedure is followed, an appropriate resistance for test SP2 can be calculated. In this respect, this test demonstrates that stainless steel may be utilised for SCS panel construction with only minimal adjustment.

3.2.14 Tan et al. (2015)

Tan et al.\[175\] includes a number of tests of beams subject to a point load close the support. Such a case is particularly onerous case for rules regarding degree of shear connection, since the 'bending span' over which enough shear studs must be mobilised tends to be short. These tests are therefore a good indicator of whether any rules that are developed are appropriate. This comparison is aided by the fact that the tests in this series are constructed using only tie-bars, and are therefore less susceptible to shear failure before excessive slip occurs.

A typical test from this series is shown in Figure 3.14.

![Figure 3.14: Typical test from tests by Tan et al.\[175\] (Test SC1 north shown)](image)

The tests included in this series are shown in Table 3.11.
Table 3.11: Summary of tests by performed by Tan et al.\cite{175}

<table>
<thead>
<tr>
<th>Test</th>
<th>$\gamma$</th>
<th>$a/d$</th>
<th>$P/\gamma\text{crutal}$</th>
<th>$P_{\text{shear}}$</th>
<th>$\U_{\text{flexural}}$</th>
<th>$\U_{\text{shear}}$</th>
<th>$R_{\text{Test}} / R_{\text{Design}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC1 north</td>
<td>406</td>
<td>2.50</td>
<td>1.17</td>
<td>0.13</td>
<td>26.9</td>
<td>26.9</td>
<td>1.00</td>
</tr>
<tr>
<td>SC1 south</td>
<td>406</td>
<td>2.50</td>
<td>1.17</td>
<td>0.13</td>
<td>26.9</td>
<td>26.9</td>
<td>1.00</td>
</tr>
<tr>
<td>SC2 south</td>
<td>406</td>
<td>2.50</td>
<td>1.17</td>
<td>0.13</td>
<td>28.2</td>
<td>28.2</td>
<td>1.00</td>
</tr>
<tr>
<td>SC3 north</td>
<td>406</td>
<td>2.50</td>
<td>1.17</td>
<td>0.13</td>
<td>28.2</td>
<td>28.2</td>
<td>1.00</td>
</tr>
<tr>
<td>SC3 south</td>
<td>406</td>
<td>2.50</td>
<td>1.17</td>
<td>0.13</td>
<td>28.2</td>
<td>28.2</td>
<td>1.00</td>
</tr>
<tr>
<td>SC4 north</td>
<td>406</td>
<td>2.50</td>
<td>1.17</td>
<td>0.13</td>
<td>26.9</td>
<td>26.9</td>
<td>1.00</td>
</tr>
<tr>
<td>SC4 south</td>
<td>406</td>
<td>2.50</td>
<td>1.17</td>
<td>0.13</td>
<td>21.5</td>
<td>21.5</td>
<td>1.00</td>
</tr>
<tr>
<td>SC5 south</td>
<td>406</td>
<td>1.50</td>
<td>1.17</td>
<td>0.13</td>
<td>16.1</td>
<td>16.1</td>
<td>1.00</td>
</tr>
<tr>
<td>SC5 north</td>
<td>406</td>
<td>1.50</td>
<td>1.17</td>
<td>0.13</td>
<td>13.4</td>
<td>13.4</td>
<td>1.00</td>
</tr>
<tr>
<td>SC6</td>
<td>406</td>
<td>5.06</td>
<td>1.17</td>
<td>0.13</td>
<td>56.4</td>
<td>56.4</td>
<td>1.00</td>
</tr>
</tbody>
</table>

3.2.15 Leng and Song (2016)

Research on SCS panel construction is still active with a number of research groups. A recent paper to be published on the subject has been prepared by Leng and Song\cite{105}. This test series includes 9 tests. The tests have relatively low thickness compared to others conducted by other authors, meaning these tests are more susceptible to shear failure. The degree of shear connection is also relatively high, as a result of the close spacing of both studs and tie-bars.

A typical test from this series is shown in Figure 3.15.

![Figure 3.15: Typical test from tests by Leng and Song\cite{105} (Test JZ2.5-1 shown)](image)

The tests included in this series are shown in Table 3.12.
All of the tests in the series are predicted to fail by shear. This is consistent with the crack patterns shown in the paper, which are all characteristic of shear failure.

### 3.3 Conclusion

As discussed in detail in Section 3.2, a considerable number of tests have been conducted to explore the resistance of SCS panels to out-of-plane forces. The tests cover a large number of different parameters and potential designs. Figure 3.16 shows a plot of the tests included in the database, characterised by panel depth (which indicates scale) and shear span to effective depth \( (a/d) \) ratios (which is a key indicator of behaviour; see Section 5.3.2). It can be seen that the database covers a wide variety of designs.
Significant deviation does not occur between SCS panels and conventional reinforced concrete structures. Bending failure is generally controlled by the tension steel, while shear failure is generally dominated by concrete cracking. Existing models for reinforced concrete structures are successfully employed for both shear and bending resistance, though modifications are required.

While the database is large, it does contain a number of gaps. Tests with low degree of shear connection are rare, with those tests that are available suggesting that existing interpretations of the structural behaviour may be lacking. In areas where the database is lacking, finite element modelling may be used to fill the gaps. This is discussed further in Section 4.4.
Bending Resistance & Minimum Degree of Shear Connection

Detailed review of a number of tests (as described in Chapter 3) and existing design methodologies has highlighted a gap in understanding of the behaviour of SCS panels when low levels of shear connection are allowed for between the steel plates and the concrete. Changes in the behaviour are particularly acute in the out-of-plane limit states, as discussed in Section 2.5.

In order to improve understanding, each of the out-of-plane limit states is investigated in turn. The focus of this chapter is the bending resistance.

The investigation focuses first on the existing design models for bending, which are found in a number of design guides. Weaknesses in the design models with respect to the effects of reduced shear connection are found. This investigation finds that the test evidence that is available from the test database (described in Chapter 3) is not sufficient to fully clarify the behaviour, so a detailed finite element model is developed, capable of accurately reproducing the existing tests, covering a wide range of degrees of shear connection, and can be extrapolated to designs that have not been tested.

Application of the finite element model to a number of the tests shows that existing theories about the distribution of forces between individual shear connectors (such as the theory by Newmark et al.\cite{126} often used in beam design) are not correct, particularly on the tension plate. This lack of precision is found to effect design of panels subject to uniformly distributed loads (UDL), with unconservative resistances being predicted by the analytical model when compared to the results from FE and testing. Improvements to the existing model are proposed, taking into account the effect of the stud force distribution changes, which give more accurate results in the cases analysed.
4.1 Existing model

In all of the design manuals for SCS panels throughout the world, the bending resistance of an SC panel is found from a first principles plastic analysis of the cross-section\[^{[24]}\]. However, to justify the use of plastic analysis additional checks are required on the stiffness of the shear connection and the buckling resistance of the compression plate. It is these rules that vary between standards, and where uncertainty remains.

The expressions and terminology presented below is consistent with the expressions and terminology found in the SCIENCE SCS Design Manual\[^{[4]}\]. This standard implements the principles of Eurocode 4\[^{[29]}\].

The plastic bending resistance of the cross-section is given by taking moments of stress blocks about an arbitrary point. Assuming this point is the bottom of the section and tensile stresses are positive, the sagging resistance is given by:

\[
M_{sag} = \left\{ +f_{y,b}t_b \left( \frac{t_b}{2} \right) - f_{y,t}t_t \left( h - \frac{t_b}{2} \right) - \alpha_{cc}f_c h_c \left( h - t_t - \frac{h_c}{2} \right) \right\} b_w \tag{4.1.1}
\]

Where:

- \(f_{y,b}\) is the yield strength of the bottom plate
- \(f_{y,t}\) is the yield strength of the top plate
- \(t_b\) is the thickness of the bottom plate
- \(t_t\) is the thickness of the top plate
- \(f_c\) is the cylinder strength of the concrete
- \(\alpha_{cc}\) is a factor accounting for long term load effects and stress confinement effects in flexure, usually taken as 0.85
- \(h_c\) is the thickness of the concrete compressive stress block
- \(b_w\) is the width of the panel

Figure 4.1 shows the assumed stress blocks associated with this model:
The hogging resistance is given by:

\[ M_{\text{hog}} = \left\{ -f_y b t_b \left( \frac{t_b}{2} \right) + f_y t_l \left( h - \frac{t_b}{2} \right) + \alpha_{cc} f_c h_c \left( t_b + \frac{h_c}{2} \right) \right\} b_w \]  

(4.1.2)

Assuming the concrete has no tensile strength, the concrete is effective up to the position of the plastic neutral axis. By considering equilibrium of the stress blocks, the following expression is developed for \( h_c \):

\[ h_c = \frac{(f_y b t_b - f_y t_l)}{\alpha_{cc} f_c} \]  

(4.1.3)

The analysis is more complicated when partial shear connection is considered. Partial shear connection occurs when there are insufficient studs to carry the forces required to yield the plate. In these cases the force mobilised in the plastic analysis is limited by the force capable of being transmitted through the studs. The sagging resistance (given in Equation 4.1.1) then becomes:

\[ M_{\text{sag}} = \left\{ +f_y b t_b \left( \frac{t_b}{2} \right) \mu_b - f_y t_l \left( h - \frac{t_b}{2} \right) \mu_l - \alpha_{cc} f_c h_c \left( h - t_l - \frac{h_c}{2} \right) \right\} b_w \]  

(4.1.4)

Where:

\[ \mu_b \quad \text{is the degree of shear connection of the bottom plate} \]

\[ \mu_l \quad \text{is the degree of shear connection of the top plate} \]

The hogging resistance is given by:
\[ M_{bog} = \left\{ -f_{y,b} t_b \left( \frac{t_b}{2} \right) \mu_b + f_{y,t_t} \left( h - \frac{t_b}{2} \right) \mu_t + \alpha_{cc} f_c h_c \left( \frac{t_b + h_c}{2} \right) \right\} b_w \] (4.1.5)

And:

\[ h_c = \frac{f_{y,b} t_b \mu_b - f_{y,t_t} t_t \mu_t}{\alpha_{cc} f_c} \] (4.1.6)

The degree of shear connection in the plate is calculated as follows:

\[ \mu_b = \min \left\{ \frac{f_{y,b} t_b}{f_{y,b} t_b}, n P_{Rd} \right\} \] (4.1.7)

\[ \mu_t = \min \left\{ \frac{f_{y,t_t} t_t}{f_{y,t_t} t_t}, n P_{Rd} \right\} \] (4.1.8)

Where:

- \( P_{Rd} \) is the resistance of an individual shear connector
- \( n \) is the number of shear connectors between the ‘critical cross-section’ and the nearest support

Shear connectors resistance is calculated according to Equations 6.18 and 6.19 of Eurocode 4\(^{[29]}\), as presented below in Equation 4.1.9. The same expression is used to calculate the resistance of tie-bars, which are assumed to contribute to the longitudinal shear resistance in addition to the shear studs.

\[ P_{Rd} = \min \{ P_{Rd,steel}, P_{Rd,conc} \} \] (4.1.9)

Where:

\[ P_{Rd,steel} = \frac{0.8 f_u \pi d^2 / 4}{\gamma_V} \] (4.1.10)

\[ P_{Rd,conc} = \frac{0.29 d^2 \sqrt{f_{ck} E_{cm}}}{\gamma_V} \] (4.1.11)

Where:
\[ \alpha = 0.2 \left( \frac{h_{sc}}{d} + 1 \right) \quad \text{For} \quad 3 < h_{sc}/d < 4 \]
\[ \alpha = 1 \quad \text{For} \quad h_{sc}/d > 4 \quad (4.1.12) \]

\[ \gamma_V \] is the partial factor for design shear resistance of a headed stud, usually taken as 1.25
\[ f_u \] is the ultimate tensile strength of the stud
\[ f_c \] is the compressive strength of the concrete
\[ h_{sc} \] is the nominal height of the connector

Accurate determination of the degree of shear connection requires a definition of the ‘critical cross-section’, which is the point within the span where bending failure can be expected to occur. As discussed in Section 4.2, this point is difficult to define in SCS panels.

Application of Equations 4.1.1 and 4.1.2 requires that the compression plate does not buckle before the material yield strength is reached. The stress required for buckling to occur is predicted by strut buckling principles, using a buckling length back derived from tests. This principle is found in all of the design guides. For the purposes of the SCIENCE Design Guide\(^4\), these relationships were simplified into \(s/t\) ratios i.e. spacing / thickness ratios. The derivation is explained further in the background document\(^3\).

In cases where compression plate buckling occurs, the resistance can still be calculated by reducing the thickness of the compression plate to zero in Equations 4.1.1 and 4.1.2. For most designs, ignoring the compression plate only tends to lead to a small reduction in resistance, as there tends to be a sufficient depth of concrete within the concrete core to mobilise a stress block of sufficient depth to balance the tension force in the tension plate.

Conversely, it is possible to simplify the analysis by ignoring the contribution of the concrete in situations where the compression plate does not buckle and the steel strength and thicknesses of both plates are equal i.e. \(\mu_b f_y b t_b = \mu_t f_y t_t\). While this simplification implies that compression plate and tension plate failure would be occurring simultaneously, in reality the compression plate will not fail; yield of the compression plate would lead to force being transferred to the concrete. Cases where equal plates are not provided will not follow this assumption.

The distribution of force between the concrete and the compression plate depends on
the strain compatibility of the two materials, which changes with stress. The distribution is also affected by the degree of shear connection, since lower values tend to lead to increased slip, resulting in less force being transferred into the plate. While deformation compatibility is of interest, understanding the distribution is not particularly important for ULS design, since the compression plate tends not to govern design (as discussed in the previous paragraph). However, understanding the force distribution is important for accurately reproducing the panel deflection. This topic is explored further in Chapter 6.

4.2 Problems with the existing bending model

4.2.1 Misidentification of the critical cross-section

Traditional composite design considers the number of studs between the critical cross-section and the nearest point of zero moment when calculating the degree of shear connection, as per Equations 4.1.7 and 4.1.8. For simply supported panels, the nearest point of zero moment is the nearest support. The critical cross-section is usually taken as the cross-section subject to the highest applied moment. However, this simplification can be unconservative, particularly for beams with lower degrees of shear connection, subject to UDLs (as explained further is Section 4.9).

Conventional steel-concrete composite construction has been established in the UK for over 40 years, but research is still active. A recent European collaborative research project entitled 'Development of improved shear connection rules in composite beams (DISCCO)' was undertaken to investigate various issues around composite construction[100], one of which was misidentification of the critical cross-section, and the effect it has on design. Figure 4.2 shows a diagram taken from a DISCCO presentation[99].
Figure 4.2: Location of the critical bending cross-section for a typical composite beam subject to a UDL.

Figure 4.2 shows the applied moment and moment resistance at a number of locations within the span of a typical composite beam. Since the load applied is uniformly distributed, the bending moment at a given location is given by Equation 4.2.1. It can be seen that $M_{Ed}$ is proportional to $x^2$ (where $x$ is the distance between the cross-section and the nearest support), which means the bending moment diagram is parabolic.

$$M_{Ed} = \frac{wx}{2} (l - x) \quad (4.2.1)$$

The resistance of the composite section depends on the number of studs between the critical cross-section and the nearest point of zero moment, as discussed above in Section 4.1. Since the composite resistance is proportional to the number of studs, which are distributed linearly, the composite resistance also varies linearly. At the support the contribution of composite action to the resistance of the cross-section drops to zero, although the section itself still retains significant resistance.

Figure 4.2 also highlights the issue with the selection of the critical-cross section. To produce this plot, the authors have selected loads that give an applied moment $M_{Ed}$ at the centre of the span equal to the moment resistance of the span $M_{Rd}$. However, it can be seen that between the two green lines the black applied moment line dips underneath the red resistance line. This implies that the design moment exceeds the resistance at those locations, even though the load and resistance are equal at the centre point.
Both the reduction in composite action close to support and the misidentification of the critical-cross section issues have been recognised from the early conception of composite construction. Many of the early publications on composite construction, such as the seminal work by Johnson\textsuperscript{[88]}, explored the possibility of non-uniform stud spacings, such that the resistance envelope matches the curvature of the bending moment diagram as closely as possible. However, non-uniform shear connector layouts are not generally used in current construction practice for the following reasons:

- Designs with highly detailed stud arrangements will tend to be difficult to set-out and position correctly on site. Designs allowing for non-linear stud layouts will tend to have a higher risk of deviation from the original design.
- Design for non-regular shear connector placement requires considerably more calculation effort, given the effect of the connector placement must be evaluated at multiple locations with the span. This extra design work must be paid for by the client, and is usually uneconomic to the project.
- The effect of the enhancement provided by the ‘optimum’ placement of studs is small.
- Most composite beams in buildings are constructed using composite steel sheeting\textsuperscript{[143]}. The shapes of these profiles limit the locations that shear connectors can be placed. It should be noted that this problem does not occur in beams where the decking runs parallel to the beam (typical in primary beams), or where a solid slab is used, such as in bridges. Profiled steel sheeting is not used in SCS construction.

In conventional composite construction the difference between the location of the critical cross-section and the point of maximum moment are often small, since the beam itself usually has a significant resistance even without the contribution of any composite action.

However, the behaviour of SCS panels is different, since the moment resistance is derived entirely from development of composite action. Figure 4.3 shows a similar plot to Figure 4.2 for an SCS panel.
It can be seen in Figure 4.3 that at locations close to supports the number of studs may be very low. Even if this issue was overcome, it can still be seen that the critical cross-section would not occur at the point of maximum bending moment.

Researchers disagree on the extent to which the misidentification of the critical cross-section significantly effects the design\textsuperscript{[3]}. Many researchers argue that the effect is small, and that this kind of uncertainty is covered by the partial factor on resistance. It is for this reason that the design manual for SCS structures\textsuperscript{[4]} does not explicitly require the designer to consider the critical cross-section.

FE and test evidence presented in this suggests that the misidentification of the critical cross-section can significantly affect the prediction of the panel resistance in some circumstances. The problem can lead to unsafe designs for panels with low degrees of shear connection, subject to UDLs. This is explored further in Section 4.9.

### 4.2.2 Prediction of end slip & the minimum degree of shear connection

A final issue concerns ‘minimum degree of shear connection’. Detailed explanation of the concept of minimum degree of shear connection can be found in Steel Construction Institute publication P405\textsuperscript{[51]} . A typical composite beam consists of both a steel section and a concrete slab, which are mechanically attached to each other with shear connectors, in order to achieve composite action. The enhancement of resistance from composite action is not uniform across the span; for cross section locations near to the supports there is little or no composite action due to the short distance of mobilisation. However, the resistance of the steel beam, acting on its own without composite action, may be capable of resisting considerable moment without composite action, as can be seen in Figure 4.2.

In some design configurations the ‘optimum’ design may have an extremely low number of shear connectors in the span, when only a small degree of composite...
enhancement is needed. However, the deformation of the steel-concrete interface is proportional to the number of studs. When the number of studs is low the deformation can reach such a level that the deformation capacity of the stud is exceeded, such that it breaks off. The breaking of a single stud can lead to a catastrophic failure of the shear connection, since the force being carried by the stud that has broken must be carried by the remaining studs in the span. The additional load might be enough to fail the second stud, which will in turn then load other studs. This form of brittle failure is often referred to as ‘un-zipping’.

Un-zipping failure is prevented as long as slip does not exceed the deformation capacity of the connector. The generally accepted value for maximum slip is 6mm (see P405[51]).

Predicating slip for a given design is not possible using a closed form equation. Instead, researchers have correlated predicted slip with a number of studs, using finite element analysis[102]. The rules tell the designer how may studs the span must contain such that slip failure will not occur. Since adding additional studs will decrease slip, the designer is free to include more studs i.e. the number is a minimum, rather than an absolute.

The rules for minimum degree of shear connection for ordinary composite construction are a function of several parameters in the design, but most notably the span. For small spans, the degree of shear connection might be as low as 32% (see Figure A.1 of P405[51]). Since these rules are specific to parameters present in conventional composite construction, it is clear that the existing rules do not necessarily apply to SCS panel design. New rules must therefore be developed, as discussed in Section 4.10.

Section 4.9 presents evidence that excessive slip is not prevented by ensuring a minimum degree of shear connection is achieved, as the on-set of slip is affected by the presence of cracking on the tension face.

### 4.3 Methods of assessing bending behaviour

Assessing the accuracy of existing or new rules for bending resistance of SCS panels requires comparison with tests. As discussed in detail in Chapter 3, a large body of test evidence exists for panels failing in bending. However, the range parameters that can be varied in an SCS panel design (height, plate thicknesses, concrete and steel strength, bar spacing etc.) inevitably mean that some combinations have not been tested. Review of the available test data shows the following gaps in the available tests:
• Beams subject to a uniformly distributed load - In all of tests conducted loads are always applied at discrete points. While this is understandable for practical reasons, modelling shows a test subject to a UDL acts differently to a beam subject to 4 point bending, which is generally used to approximate behaviour of beams subject to a UDL (see Section 4.9.2).

• Large scale beams - Much of the testing performed in the literature is on scaled model tests. While this is fine for understanding the mechanics (and for practical reasons), there is little evidence for full size beams. SC panels in nuclear application designs are tending to reach very large depths, often in excess of 1m, for which testing has not been undertaken.

• Beams with low degrees of shear connection - The performance of panels with lower degrees of shear connection is more uncertain than the performance of panels with full shear connection, since the deformation of individual connectors and distribution of forces between them becomes critical. Some test evidence is available, but not enough to get a full understanding of the effect of the multiple parameters involved.

Given the lack of test evidence in a number of key areas, it is clear that some form of numerical modelling is needed to extend understanding into areas not covered by testing.

Of particular concern in bending is the need for accurate predictions of interface slip, to allow assessment of rules for minimum degree of shear connection (see Section 4.2). Analytical models exist for translation of load into connector slip (see Equation 9.2 of report by Aggelopoulos et al. [6]). While these relationships have proven to be reasonably accurate for conventional composite construction, the models rely on a predictable and smooth distribution of stud force between connectors. As the results in Section 4.8 show, this assumption is not valid, particularly on the tension side.

The finite element model described in Section 4.4 is capable of calculating the cracking of the section on the tension side, and is therefore capable of reproducing the actions of the mechanism that leads to uneven distribution of stud force. It is therefore a suitable tool for predicting slip in composite structures. Finite element modelling has been used by a number of other researchers investigating ultimate resistance and end-slip of composite beams [6,167].

While the FE techniques employed allow the effects of the force distribution to be predicted, precise prediction of end-slip relies on an accurate model for the strength and stiffness of individual connectors, which are entered into the model as spring
stiffnesses at the base of the stud (the modelling techniques for individual connectors are discussed in Section 4.5). The choice of load-slip model is discussed in Section 4.6.

4.4 Modelling using finite element analysis

Modelling of SCS panels presents a challenge over and above the normal challenges associated with reinforced concrete modelling. Stress in the structure is most highly concentrated around the studs, requiring a high level of discretization. Since this level of refinement is required at each of the many stud locations, this can make the model highly numerically expensive.

Modelling of the SCS panels has been carried out using the FE software ABAQUS. ABAQUS is commonly used for modelling of conventional concrete and SC panel structures[107].

4.4.1 2D vs. 3D Modelling

Early on the work, both 3D and 2D models with built. After evaluating both approaches, it was decided to model the structure in 2D rather than 3D. Although 3D represents an optimum in being able to account for differential deformation and stress distributions in-plane, it was observed from the tests described in Chapter 3 that much of the behaviour of interest occurs in a planner manner. 2D modelling offers considerably reduced computation time, which therefore offers the ability to create a much finer mesh. 2D also allows for consistency with the cross-sectional analytical models, such as the one described in Section 4.1, which are also 2D.

The key disadvantage of 2D modelling for SCS panels is that the effects of in-plane shear cannot be modelled. While not within the scope of this work, this means that the same model cannot be extended to either pure in-plane shear models or models that combine in-plane and out-of-plane loads. However, it should be noted that the modelling techniques described (such as the use of springs to model shear connection as described in Section 4.5) are equally applicable to 3D models, with little adjustment.

Modellers should be aware that a 3D model with sufficient elements to properly capture the cracking of the concrete will require considerable computation. 3D models should be avoided if the behaviour is planer in nature.
4.4.2 Elements, meshing & loading

2D models are typically meshed with CPS4R 2-dimensional reduced integration elements in ABAQUS. These elements have been shown to work well in other concrete modelling problems\(^63\).

Even for relatively simple arrangements a very fine mesh is needed. The mesh becomes even more refined around the studs, since the concrete here is subjected to concentrated forces, and shows the most propensity to crack.

There is considerable variation in the scale of the beams in the database; the largest beam has a height of 1219mm (see Section 3.2.11), while the smallest has a height of 150mm (see Section 3.2.5). With such variation, a single fixed mesh size would be inappropriate; small beams would not have enough elements to predict the behaviour correctly, while large beams would take too long to compute. Parametric mesh refinement studies showed that the best scaling parameter for the mesh was to set a fixed number of elements throughout the depth of the beam. This number was eventually set as 40.

The key test of mesh sensitivity is the prevalence of artificial strain energy. Artificial strain energy is introduced to elements to avoid the hour-glassing phenomena, where elements undergoing large deformations can be subject to spurious zero energy deformations. Specification of a greater number of elements through the depth of the beam usually leads to less need for artificial strain energy to stabilise the model, and is therefore a good indicator of mesh sensitivity.

ABAQUS documentation generally recommends that artificial strain energy be kept to less than 1% of total energy in the system\(^165\).

Figure 4.4 shows the artificial strain energy compared to the internal energy for test SP1 by Koukkari and Fülöp\(^98\) (as discussed in Section 3.2.13). This test is bending dominated, which tends to mean element deformation is spread between a large number of elements, meaning the average element deformation is low. Low element deformation is associated with low artificial strain energy, meaning the 1% criteria is easily met.
Figure 4.4: Graph showing the ratio of artificial strain energy and internal energy against time for test SP1 by Koukkari and Fülöp\cite{98}. Load-time is also shown to highlight significant events in the load cycle.

By contrast, shear failure is characterised by large deformation of small number of elements, at the locations of the discrete cracks. Test JZ3.0-1 by Leng and Song\cite{105} (as discussed in Section 3.2.15) fails by shear. As shown in Figure 4.5, this tends to result in a relatively high artificial strain energy.
It can be seen from plots like Figure 4.5 that the 1% energy balance criterion is often not achieved throughout the loading phase, even with a fine mesh. Coarser meshes were found to require even more artificial strain energy, often in the order of 20% or greater. However, given the available computational resources and the strong correlation with test results it was felt that this level of mesh refinement offers a good level of compromise. Future researchers may wish to specify an even finer mesh as computational power improves with time.

**Loading & Mass Scaling**

The loading applied to the model seeks to simulate the loading method to which the structure is subject as closely as possible. In reality, most loads are applied through the use of hydraulic jacks, whose loads are applied to the structure through a spreader plate. Loads in the model are applied are also applied through plates. In most cases, the width or thickness of the spreader plate is not recorded in the literature, so appropriate values must be assumed. However, the results do not appear to be sensitive to either choice, as long as the plate is wide enough not to unduly concentrate the load on a
particular spot, and the beam is sufficiently slender.

 Loads in this model are displacement controlled, rather than load controlled. In this loading scheme, the boundary condition given to the plate is a displacement, incremented over the model time. The application of this displacement produces a reaction force at the node of load application, which is equivalent to the load applied to the structure. As the structure cracks over the length of the test it is possible for the reaction force to drop.

 Figure 4.6 shows the outputs from both loading methods for a hypothetical concrete structure. It can be seen that load-control captures the failure point and the initial stiffness, it does not show any evidence of the formation of a second crack in the early stages. The model also stops at the point of maximum applied force, meaning none of the post failure behaviour is captured. This data is useful in understanding the behaviour of the test, since a large drop in applied force is characteristic of a shear failure, while a smooth plateau can indicate bending failure.

 The solver for this model is explicit, meaning the structure does not have to be in static equilibrium at each load-step. It is possible for the load to cause the structure to either oscillate or under-go free body motion if the load is applied too quickly. However, slow loading rates can result in increased computation times, since the model must be calculated at a fixed time increment, discussed below, that is independent of the loading rate. A balance must be found.
Chapter 4: Bending Resistance & Minimum Degree of Shear Connection

Figure 4.6: Schematic diagram of the load-deflection response prediction of a typical model of a concrete structure, using load control and displacement control.

Explicit solves were carried out at fixed time steps. The appropriate time increment is a property of the model, and is linked with the time taken for a stress wave to propagate through an element, which is assumed to occur at the speed of sound. This is known as the ‘stable time increment’. Discussion of the stable time increment concept can be found in ABAQUS help files\textsuperscript{[166]}. The stable time increment is calculated automatically by ABAQUS, for each element in the problem, with the lowest element stable time increment being the stable time increment for the model. The time increment is as follows:

\[
\Delta t = \frac{L_e}{\sqrt{\frac{E}{\rho}}} \quad (4.4.1)
\]

Where:

- \( L_e \) is the effective length of the element. For 2D planer elements (as is typical in this model), this is taken equal to the square root of the area of the element.
- \( E \) is modulus of elasticity of the element material.
- \( \rho \) is the element material density.
It can be seen that the element time increment is proportional to the effective length of the element, which means that the finer the mesh the smaller the time increment, which has the effect of increasing the computation time of the problem. However, maximization of element size to reduce the stable time increment conflicts with the general aim in finite elements of making the mesh as fine as possible, so as to linearise the stress state across the element as much as possible.

Maximisation of the element size is not the only strategy for increasing the stable time increment. As Equation 4.4.1 shows, the stable time increment is also proportional to the modulus of elasticity and the density of the element material. The modulus of elasticity cannot be manipulated, as specifying the correct stiffness is a vital feature of the model. However, the material density does not affect the result of a static analysis, the conditions for which this model is attempting to obtain as far as possible. Manipulation of the density to increase the stable time increment is known as ‘mass scaling’. The topic is extensively discussed in the ABAQUS theory manual\cite{166}.

Mass scaling can be specified as a fixed value, or as a function of the value required to give a set time increment i.e. for small elements with a low characteristic length a greater magnitude of mass scaling is applied. The later approach is used for this model.

![Figure 4.7: Mass-scaling factor for elements in descending order. Plot is for SCIENCE SP1 (see Section 4.7.2).](image)

The geometry of this problem means that a mapped mesh with a consistent sized element is not possible, so the automatic meshing tool is used. A feature of the meshes created by this tool is that a small number of small triangular elements are added to
achieve a smooth mesh. These triangular elements have relatively low characteristic lengths, so must be scaled to a greater degree. The mass-scaling magnitudes applied to the model are arranged by highest to lowest values, and are shown in Figure 4.7. It can be seen that a small number of elements have very large relative magnitudes of scaling.

Figure 4.7 shows an abrupt transition in mass-scaling ratios at around 26000 elements. Investigation of this point showed that elements on the left tended to be concrete elements, while elements on the right are steel plate elements. The steel elements tend to be more uniform in size, and additionally have a greater density than the concrete elements before mass-scaling is applied, so have relatively lower mass-scaling ratios.

As Figure 4.7 shows, the values of mass-scaling that are applied are very large. The largest value applied is 48746.48%. While such values are clearly not realistic, such large masses do not affect a static analysis. However, since the model is using the explicit solver static conditions are not a given. Conditions are close to a steady state, known as a quasi-static state, if the kinetic energy in the system is minimised with respect to the strain energy in the system (known as the internal energy in ABAQUS literature). ABAQUS recommends that quasi-static conditions are achieved if the kinetic energy is less than 10% of the internal energy.°

Much effort was expended in determining a loading rate and mass scaling factor that would produce quasi-static conditions in a reasonable computation time. The eventual value chosen for the mass-scaling factor used 0.003 seconds as a target stable time increment. A linear ramped deflection of 100mm is applied over 1000 seconds.

It can be seen in the model verifications described in Section 4.7 that a 10% kinetic energy ratio is achieved, even with the very large mass-scaling ratio that is applied.

4.4.3 Steel material model

Steel is relatively simple to model compared to concrete. For this model, each of the steel components were modelled using a tri-linear relationship. The first phase use the elastic modulus, up to the defined yield strength. From there, the second transition point occurs at 1.1% strain, up to the ultimate tensile strength. The strength then plateaus. No ultimate strain limit is set, to avoid convergence problems. This model is consistent with the model presented in EN 1993-1-5 Annex C.
Figure 4.8: Assumed stress/strain relationship for steel

Given this relationship, it is possible to describe all of the steel in the problem using three parameters; the yield strength $f_y$, the ultimate strength $f_u$ and the modulus of elasticity $E$. The strengths $f_y$ and $f_u$ tend to recorded in the test reports / literature, while the modulus of elasticity $E$ for carbon steel is usually taken as a typical value of 210GPa. Where these properties are not available, appropriate values are estimated.

For the purposes of the parametric study, all steel is assumed to be S355, and therefore has a yield strength of 355MPa. Unless explicitly tested and recorded in the test paper, the ultimate strength is taken as $1.3 \times 355 = 461.5$; 30% is a typical over-strength for carbon steel$^{[114]}$.

4.4.4 Concrete material model - Damaged Plasticity

The behaviour of concrete is highly non-linear, undergoing several changes of behaviour, depending on the strain to which is subjected. Its behaviour also changes according to the level of confinement, with stress from different directions fundamentally changing the mechanics of the material behaviour. In addition, concrete is particularly sensitive to stress history changes, with large stresses often significantly weakening the material when re-loaded.

The complexity of the material means that there is no single accepted material model that is capable of reproducing the exact behaviour of concrete. Those models that are
available usually include a large number of parameters, the effects of which can be insignificant, or may only govern behaviour in certain stress states. It is therefore important that each parameter is defined in detail, and specified with as high an accuracy as possible, given the data that may be available.

ABAQUS is widely accepted and used for modelling of concrete structures. The program offers two solvers; implicit integration in ABAQUAS standard, and explicit integration in ABAQUS explicit. The standard solver implements a solver based on establishing equilibrium in the structure. The standard solver can be used to understand the resistance of concrete structures, but suitable applications are usually limited to cases where the formation of small but numerous cracks occur (often the case in bending), with gradual changes of stiffness. However, this is not the case in analysis of SCS panels in shear, where the response is dominated by the formation of a single catastrophic crack. For this reason, ABAQUS explicit was chosen.

ABAQUS explicit includes two key models for concrete; concrete damaged plasticity and brittle cracking. The main difference between the models is how they consider the effect of crack formation across the element. The brittle cracking model uses a brittle failure model, where exceeding the maximum tensile stress of the material in the principle direction results in removal of the element from the analysis. The model works well for problems dominated by tensile cracking, but is weak for problems involving compression, since the model assumes a linear elastic compression relationship.

The alternative model is ‘concrete damaged plasticity’ (CDP). In contrast to the brittle cracking model, the CDP model allows for complete descriptions of both the compressive and tensile stress strain behaviours. This allowance is a considerable advantage over the brittle cracking model, meaning it is used readily in most recent publications concerning concrete modelling. Since the model is so widely used for problems of this type, it is the chosen model for this work.

The concrete damaged plasticity model requires the definition of a number of properties. These are discussed below.

**Yield function**

The majority of parameters used to define the behaviour of concrete according to CDP are used to define the shape of the yield function.

A yield function is used to define the relationship between the applied stresses and the point of yielding. The yield function is simple to define for uni-axial stress, but is
CHAPTER 4: BENDING RESISTANCE & MINIMUM DEGREE OF SHEAR CONNECTION

considerably more complicated when the combination of stress in either 2 or 3 directions is considered, for 2D and 3D problems respectively. The definition of the yielding point for combinations of stress is not straightforward even for a homogeneous isotropic material like steel. Yield criteria appropriate for steel include the Tresca yield surface (maximum shear stress) and the Von Mises yield criterion.

In-homogeneous materials like concrete require yield functions that do not centre on the zero stress origin. A typical example is the Mohr–Coulomb yield surface, which is often used for soil. The most typical yield criterion for the behaviour of concrete is the Drucker-Prager yield surface. A modified version is adopted for the CDP, allowing for the fact that degradation of stiffness occurs at different rates in compression and tension. The model is described by Lee and Fenves\textsuperscript{[103]}.

The yield function adopted for CDP is shown in Figure 4.9:

![Figure 4.9: Yield surface in plane stress for the CDP model. Reproduced from the ABAQUS user manual\textsuperscript{[164]}.

The shape of the yield function is described by a number of parameters. However, the effect of varying them is in this context is generally negligible, as the response of the structure is dominated by the assumptions about post-cracking tensile behaviour, with the shape of the yield function taking on secondary importance. Where these properties are needed they are taken from Jankowiak and Lodygowski\textsuperscript{[86]}, a publication extensively cited for this purpose\textsuperscript{[7]}.

**Tensile behaviour**

Modelling of tensile behaviour of concrete is much more difficult than its compressive behaviour. The initial response of concrete in tension is linear elastic, but at a
relatively low level of stress (compared to its compressive strength) cracking occurs. From this point, the evolution of cracking through the whole structure depends on a large number of parameters. Once cracked, the stress that can be carried through the material does not immediately drop to zero; this effect is known as ‘tension stiffening’. This parameter is particularly important in modelling of response to out-of-plane shear.

The variability of tensile behaviour with the design situation means that a single unified theory for concrete tensile behaviour is not given in any publication. Design standards, such as BS EN 1992-1-1\textsuperscript{[30]} typically specify that any tensile resistance of concrete should be conservatively neglected. While this may be adequate for design, back analysis demands a more precise representation.

The most comprehensive and widely used description of the tensile behaviour of concrete is given in Section 5 of the FIB model code\textsuperscript{[60]}.

**Mesh dependency**

Reproduction of the behaviour of structures that undergo cracking is complicated by a dependency on the mesh size, occurring because of the concentration of deformation in the crack itself once it has formed.

The problem is best explained by considering a test of a concrete bar subject to a tensile force at one end. The stress exerted by the force is uniform throughout the cross-section. The tensile force is displacement controlled i.e. the force applied is the one that gives the desired displacement. The required displacement is incremented over the time of the test.

The initial response of the structure is elastic. However, once the cracking stress is exceeded the total deformation of the bar is made up of 2 components; the initial elastic displacement, and an additional displacement occurring as a result of the tension stiffening in the crack. Figure 4.10 shows an indicative response for a short bar.
Figure 4.10: Indicative diagram of finite element prediction of displacement at the end of a concrete bar in tension. Curves show mesh sensitivity

The bar could be modelled using 1 finite element, with the stress strain relationship given as one measured from a test. However, consistent results are not produced when the number of elements is changed.

In most finite element problems, splitting the bar into a number of elements greater than 1 has no effect on the total displacement at the end of the bar. This is also true for the elastic response of the concrete bar. However, once the cracking stress is exceeded the predicted stress drops more quickly for those models with a higher number of elements.

The mesh dependency issue is particularly important in modelling either RC beams or SCS panels that fail by out of plane shear. Out-of-plane shear failure is characterised by the instantaneous formation of a single discreet crack. The extent to which the crack extends and controls the resistance of the structure is heavily dependent of the fracture energy, meaning any inconsistency caused by mesh sensitivity is particularly harmful to the ability of the model to provide an accurate resistance prediction.
ABAQUS and the concrete damaged plasticity model includes a correction for mesh sensitivity. Each element in the mesh is given a unique stress-strain curve, based on a characteristic length. For a 2D problem, the characteristic length is taken as the square root of the element area. For 3D problems, the characteristic length is taken as the cube root of the element volume. For smaller elements, the post cracking stress strain curve is adjusted to fall at a slower rate.

As can be seen by comparing Figure 4.10 and Figure 4.12 (although the values shown are only indicative), the curve adjustment is calculated to reverse the reduction in fracture energy occurring as a result of the reduction in mesh size.

This adjustment for mesh sensitivity is well known. The concrete properties given in the FiB model code 2010\cite{60}, as used in this model, are presented in two phases, with
an initial stress-strain response for the elastic component and a stress-displacement response for the crack relationship.

![Pre & Post cracking deformation relationships for concrete](image)

**Figure 4.13:** Pre & Post cracking deformation relationships for concrete, as given by Fib Model Code 2010[60]

ABAQUS and the CDP assume that there is no change in stiffness in the elastic part of the response. As such, the change in stiffness shown in Figure 4.13 cannot be reproduced. Two alternative strategies were explored for dealing with this inconsistency. Firstly, the modulus of elasticity specified by the standard was used for the entire response. The second strategy involved applying a reduction factor to the initial elasticity, such that the point of cracking failure that occurs after the change of stiffness is accurately reproduced. These two options are shown in Figure 4.14.
The difference between the two is relatively small, but it was found that use of strategy 1 resulted in early onset of the shear crack. The reduction in stiffness required to implement strategy 2 is relatively small, meaning the overall prediction of stiffness for the structure is not adversely affected. As such, strategy 2 was implemented for the remaining models.

Compressive behaviour

The compressive behaviour of concrete is relatively less complex than the behaviour in tension. The required input for the damaged plasticity model is a stress-strain relationship, which is available from a number of sources. The stress strain model described in EN 1992-1-1 Section 3\[30\] is implemented for this model. The compression behaviour does not require adjustment for mesh dependency.

Concrete in compression undergoes a number of distinct changes in stiffness as stress is increased. The first phase is close to linear elastic. In accordance with EN 1992-1-1 Figure 3.2\[30\] this phase is assumed to end at 40% of the maximum compression stress. From this point, the stiffness degrades gradually, up to the point of maximum compression stress. After this maximum stress is exceeded the concrete is assumed crushed, undergoing a gradual loss of strength and stiffness.

Concrete in compression is particularly affected by confinement. This effect occurs
when the presence of a confining structure, such as tightly placed reinforcement, prevents transverse deformation of the concrete in response to uni-axial stress. Although the change in mechanical response is complex, it is observed that confined concrete has increased strength and ductility.

In conventional RC design, it is typical to ignore the beneficial effects of confinement, since the reinforcement must be detailed in such a way as to ensure the confinement assumed is achieved, which can be very difficult. In SCS panels, the concrete layer has considerable inherent confinement, both from the plates and the tie bars. To ignore this increased confinement is unnecessarily punitive.

The effects of confinement were explored as part of back analysis of the test results. This analysis showed that ignoring the beneficial effects of confinement resulted in extremely conservative results. Confinement was therefore assumed for all further models. This is discussed further in Section 4.7.2.

Figure 4.15 shows the two alternative stress-strain models, for both non-confined and confined behaviour respectively.

**Figure 4.15:** Enhanced stress/strain response for confined concrete in compression
Damage parameters

As concrete is loaded micro-cracking occurs, even at relatively low levels of stress. These micro cracks ultimately affect the stiffness of the structure. The effect of damage to the concrete from loading is most evident if the structure is loaded and then subsequently unloaded.

Damage is manifested in the CDP model through the use of a stiffness reduction factor, which reflects the stiffness reduction observed in tests once the concrete has cracked[17]. The value is between 0 and 1; 0 for an undamaged state, 1 for concrete with no stiffness. A value of 1 is never specified however, as this would cause convergence problems in the FE solver.

While the concept of the damage parameter is relatively simple, its implementation within ABAQUS and CDP is difficult. The problem is further complicated due to the use of stress-displacement crack relationship, which means the stress strain relationship, and hence the damage parameter values, are different for each element.

A key issue that must be avoided is crossing of the unload-reload lines, which would indicate an increase in stiffness for a more heavily loaded element. This situation is shown in Figure 4.16.

![Figure 4.16: Indicative diagram of crossing un-load re-load lines as a result of misspecification of concrete damage parameters](image)

Calculating the correct damage relationship proved difficult, especially since a number of crack opening relationships were investigated in the initial stages of the modelling.
Damage parameter equations are available in the literature, but they are often only applicable to one crack stress / deformation relationship.

In order to overcome the limitation of the damage curve dependency on the crack opening relationship, a new set of equations was developed by the Author to allow a consistent damage curve to be specified, no matter the crack opening relationship assumed. This proved to be of considerable advantage during initial development of the model.

The concept implemented in the equations is one of an effective ‘fan’. The ‘spread’ of the fan is controlled by a damage parameter $\phi$, which controls the ratio between the ratio of the distances to either the top or bottom datum for each unload-reload line, as shown below in Equation 4.4.2:

$$\phi = \frac{\epsilon_3}{\epsilon_1 + \epsilon_2}$$  \hspace{1cm} (4.4.2)

The parameters for Equation 4.4.2 are shown on the left of Figure 4.17. The image on the right shows a number of the unload-reload lines.

The damage parameter $\phi$ takes a value between 0 and 1, and is a property of the concrete. A value of 1 gives an undamaged relationship i.e. the gradient of the unload-reload lines is exactly the same as the initial stiffness. A value of 0 is the most damage that the model can allow, since all unload-reload lines return to the first datum. These two extreme situations are shown in Figure 4.18.
Given this is a novel approach to specifying damage, no values for the damage parameter exist explicitly in the literature. However, it is possible to back-derive values from other researcher’s damage models. This process was undertaken for the model presented by Aslani and Jowkarmeimandi\textsuperscript{[17]}. The value derived was around 0.2. This value was subsequently taken for all further models.

ABAQUS accepts the damage parameters at points on the stress strain curve. A mathematical procedure must be followed to calculate which particular unload-reload line the point falls upon. Stress-strain curves in ABAQUS are entered from the point of maximum tensile stress, so the same convention is adopted here.

The effective fan is also compatible with displacement based crack behaviour definitions, as described in Section 4.4.4. Each element has a unique damage curve to go with its unique stress-strain curve, linked to the characteristic length of the element. To view the damage curve for each element, the following transformation is used:

\[ \varepsilon_1 = \frac{w_1}{l_0} \]  \hspace{1cm} (4.4.3)

Where:

- \( w_1 \) is the crack displacement at point 1
- \( l_0 \) is the characteristic length of the element

For entry of the parameters into the software, ABAQUS assumes the values are being
entered for an element with a characteristic length of 1m. No transformation from displacement to strain is needed.

The equations used are based only on coordinate geometry. The inputs required are:

\[ \begin{align*}
E_0 & \quad \text{is the initial stiffness} \\
\sigma_t & \quad \text{is the tensile stress at the point of cracking} \\
\Phi & \quad \text{the damage parameter (between 0 and 1)}
\end{align*} \]

The damage parameter \( d_1 \) is then evaluated at a number of points on the stress-strain curve, with coordinates \( \varepsilon_1, \sigma_1 \). Any reasonable post cracking relationship may be used to define the link between \( \varepsilon_1 \) and \( \sigma_1 \).

\[ \begin{align*}
\sigma_r &= \frac{\sigma_1}{\sigma_t - \sigma_1} \\
\varepsilon_o &= \frac{\sigma_1}{E_0}
\end{align*} \]

The parameters used in effective span specification of damage parameters are shown in Figure 4.19.

First, the strain offset is calculated, based on the initial stiffness and cracking stress:

\[ \varepsilon_o = \frac{\sigma_1}{E_0} \quad (4.4.4) \]

A stress ratio is calculated at point 1, as shown in Figure 4.19. This value tends toward infinity when \( \sigma_1 \) approaches \( \sigma_t \); in this case, \( d_1 \) is always equal to zero.

\[ \sigma_r = \frac{\sigma_1}{\sigma_t - \sigma_1} \quad (4.4.5) \]
Next, the strains used to define the fan are calculated:

\[ \varepsilon_2 = \frac{\varepsilon_1 (1 - \varphi) - \varepsilon_o}{\sigma_r + \varphi} \quad (4.4.6) \]
\[ \varepsilon_3 = \varphi (\varepsilon_1 + \varepsilon_2) \quad (4.4.7) \]

The equivalent undamaged values (\( \varepsilon_{2,u} \) and \( \varepsilon_{3,u} \)) can be found by setting \( \varphi \) equal to 1.

Finally, the gradient of the unload-reload line is calculated, leading to the damage parameter:

\[ \sigma_d = E_0 \left( \varepsilon_1 - \frac{\sigma_1}{E_0} - \varepsilon_3 + \varepsilon_o \right) \quad (4.4.8) \]
\[ d_1 = \frac{\sigma_d}{\sigma_1 + \sigma_d} \quad (4.4.9) \]

As a final check, the parameter \( \varepsilon_{t,pl} \) can be calculated. This parameter is shown in ABAQUS literature. This value can be calculated in two ways; if both values are equal, the calculation has been carried out correctly:

\[ \varepsilon_{t,pl} = c_1 - \frac{\sigma_1}{E_0} \left( \frac{d}{1 - d} \right) \quad (4.4.10) \]
\[ \varepsilon_{t,pl} = -\varepsilon_o + \varepsilon_3 \quad (4.4.11) \]

### 4.5 Proposed model for shear connection

The shear connection between the concrete and steel plates can have a significant impact on the stiffness and ultimate resistance of the section, so it is vital that the behaviour of the interface is accurately reproduced. Shear stud modelling is particularly challenging with finite elements, as the behaviour of shear studs is dominated by the formation of micro-cracking around the stud, which generally cannot be reproduced without an extremely fine mesh\(^{[142]}\). A model with many studs therefore becomes prohibitively expensive computationally.

The critical performance metric of a shear stud is its load-slip behaviour. It was therefore decided to use an effective model of this relationship in the global model.

The model finally adopted is similar to ones used in conventional composite beam modelling\(^{[76]}\). The model proposed involves using an embedded beam element to model the stud itself. At the base of the stud a non-linear spring connects the beam element to the plate. Distributed coupling is required at the tip of the stud and in the
steel plate to remove potential stress concentration problems. A schematic diagram is shown in Figure 4.20.

![Schematic diagram of proposed shear stud model (Red - Beam elements, Green - Non-linear spring, Blue - Distributed coupling)](image)

**Figure 4.20:** Schematic diagram of proposed shear stud model (Red - Beam elements, Green - Non-linear spring, Blue - Distributed coupling)

The spring stiffness is set to reproduce a load-slip curve produced from push out tests. Various curves were used in parametric studies, but the model given by Molenstra and Johnson was found to give good results[^118]; this is discussed in detail in Section 4.6.

**Contact**

The proposed model contains several areas of contact between different components of the system. Contact occurs between the plates and the concrete, with additional contact occurring between the plates and points of load application, which are also modelled as steel plates.

In most situations ABAQUS manages contact settings automatically. Contact stiffnesses are required in both the tangential and normal directions. In all cases, the normal direction contact is specified as ‘hard’, meaning nodal penetration between the surfaces is minimised as much as possible. The choice of algorithm (either Penalty or 2 forms of Lagrange) is automatic. Separation of the plates is allowed, and is essential in reproducing buckling of the compression plate between studs.

In the tangential direction, frictionless contact is specified. This is considered a

[^118]: Reference to section or page number.
conservative assumption, since the use of the plates as permanent formwork means that any bonding between the plates and concrete during the casting process remains in place during testing of the samples.

Presence of bond between the steel and concrete may be a source of conservativeness when back analysing test results, as described in Section 4.7. However, the bond force generated is likely to be considerably smaller than the force generated by the shear connectors, so should only result in small deviations.

Inclusion of the bond force in parametric studies used to develop design guidance is also not appropriate. In real structures, fluctuations in loading over time will tend to break any bond that may occur. This effect is well known; when testing studs according to Eurocode 4 Annex B.2.3\[29\] ‘Bond at the interface between flanges of the steel beam and the concrete should be prevented by greasing the flange or by other suitable means’.

### 4.6 Choice of load-slip relationships for studs and tie-bars

The main technique for assessing the load-slip behaviour of an individual connector is the push-test. Given the importance of the the load-slip relationship to composite construction, which is itself the most common form of steel frame construction in the UK\[72\] , it is unsurprising that considerable research effort has been devoted to push-tests, exploring a wide range of connector sizes and arrangements.

#### 4.6.1 Mechanical behaviour of shear studs

In any composite system, composite action is achieved when two layers of material are connected such that force is transferred from one layer to the other when one is deformed. The principles behind composite action in steel-composite beams is explained by Lawson and Chung\[101\] and Johnson\[89\].

In steel-concrete composite construction, composite action is generated using steel shear studs. These studs are welded to the steel at set intervals before concrete is poured. Once the concrete is cured any force applied to the steel serves to additionally load the studs in shear.

As with any material, any application of force is accompanied by deformation. The key deformation relationship for a shear stud is the load-slip relationship, which describes the relative movement of each of the material layers for a given force. This is shown in Figure 4.21, with an inwards slip of the tension plate with respect to the concrete and
an outwards slip of the compression plate.

![Diagram of slip and concrete compression](image)

**Figure 4.21:** Typical slips observed for a simply supported SCS panel

Figure 4.22 shows a typical load-slip curve from a push-test.

![Load-slip curve](image)

**Figure 4.22:** Typical load-slip relationship for a shear connector (taken from Figure B.2 of EN 1994-1-1\(^{[29]}\))

The behaviour of concrete in compression shows an initial elastic response, followed by a gradual reduction in stiffness with load. In shear studs, the elastic phase represents a low proportion of the behaviour. This lack of elasticity occurs due to the distribution of pressure along the stud, as shown in Figure 4.23:
The stud essentially cantilevers into the concrete, which results in a high pressure at the base. As the shear force increases, this pressure causes localised crushing failure in this area. Reproducing the correct initial stiffness of the shear stud relies on correctly reproducing the stress state in this area.

Some of the first research in this area in the UK research was conducted by Johnson and Oehlers\cite{91}. This work included analysis of 125 push-out test results from 11 sources, as well as an additional 101 new push-out tests. This work was used as the basis of the design rules presented in BS5950-3-1\cite{32} and subsequently BS EN 1994-1-1\cite{29}.

Figure 4.24 shows an example of a push-test rig\cite{91}. A central steel beam is sandwiched between two concrete blocks, each placed on the floor of the laboratory. The concrete beam is then pushed down from the top. The only connection between the concrete and steel is through two studs, welded to either side of the beam. Symmetry is required to avoid introducing additional moments. Slip can be measured
anywhere, but is typically measured from the bottom of the steel to the floor.

The studs used in SCS construction are the same as those used in conventional composite beam construction, and should therefore be expected to act in a similar way. However, three important differences exist:

- Increased concrete confinement - Concrete that is restrained from expansion has been shown to have considerably more strength. In SCS construction concrete expansion is restrained by the plates.

- Thin steel plates - In conventional composite construction the studs are welded to the top flange of an I-beam. In the UK flanges tend to be 10mm or greater. The steel plates used in SCS construction tend to be thinner, with some designs requiring plates as low as 6mm thick. Thinner plates are less effective at resisting rotation of the stud as it begins to shear. This could potentially result in considerable extra slip for the same load, or reduce the peak stud resistance.

- Lack of stiffening restraint from beam web - When used in conventional composite construction shear studs are attached to the top flange of the beam, which gains considerable restraint against local buckling from the web of the beam. Since this stiffening is not present in an SCS panel it can be expected that there would be more rotation at the base of the stud for a given load, as above.

Full height tie-bars in SCS panels present a greater challenge for modelling purposes. These are discussed separately in Section 4.6.4.

### 4.6.2 Load-slip relationships in the literature

Some of the first push-out tests to be discussed in the literature are described by Ollgaard et al.\(^{[135]}\). The paper describes the evolution of shear connectors from ‘spiral connectors’ and large angles to the headed studs that are in use today. The paper describes 48 push-tests, covering various concrete strengths and bar diameters. The emphasis of the study was on concrete material properties, with the study covering both light-weight and normal-weight concrete, with various types of aggregate. The authors explicitly state in the conclusion that all of the tests they performed had similar load-slip curves; as a result, Ollgaard et al. suggest the following equation for determining the load-slip curve:

\[
Q = Q_u \left(1 - e^{-18\Delta}\right)^{\frac{3}{2}}
\]  

(4.6.1)
In the years between 1971 and 1981 various researchers performed push-out tests. These tests were eventually compiled into a database by Johnson and Oehlers\[91\]. Johnson and Oehlers additionally performed 101 additional push-tests, leading to a total database of 226 tests.

An alternative load-slip curve was suggested by Oehlers and Coughlan\[132\]. Oehlers and Coughlan analysed 53 push-out tests for the derivation of their relationship, using regression analysis to predict slip at various levels of stud load. Their curve is given in table form, with a fixed component and a component accounting for concrete strength. As shown in Figure 4.25, this curve has notably less stiffness in the initial loading phase i.e. ‘setting in’ of the specimen. The paper does not give details of the test setup, meaning it is difficult to understand if this is a result of effects at the stud, or as a result of deformation of the test rig or loading point. Given this ‘setting in’ is not seen in any other test result or load-slip curve, it is likely that the latter explanation is a more reasonable explanation.

Molenstra and Johnson\[118\] provided two load-slip curves, additional relationships for calculating load-slip.

\[
P = P_d \left(1 - e^{-1.000s}\right)^{0.558}
\]

(4.6.2)

\[
P = P_d \left(1 - e^{-1.535s}\right)^{0.989}
\]

(4.6.3)

Where:

\[s\] is the slip in mm

The paper compares these relationships against existing relationships, such as the one proposed by Ollgaard et al., and finds that the curves tend to produce very similar results. The curve by Oehlers and Coughlan is found to deviate from the others due to the presence of splitting failure in the test sample, as a result of the tests being conducted on thin slabs. Splitting failure of this kind is very unlikely to occur in an SCS panel, since the concrete into which the stud is embedded is continuous across the width of the panel, and can therefore be considered to be a solid slab. Oehlers and Coughlan found no splitting failure when the solid slabs were tested.

It should be noted that the purpose of the load-slip curves given in this paper is to
provide inputs for computer modelling of total end slip, in a similar manner to the analysis attempted in this thesis. The authors suggest that in their model “Maximum slips are obviously not sensitive to the shape of the P-S curve”.

A further load-slip curve is proposed by Gattesco and Giuriani\textsuperscript{[73]} . The model is described as a “refinement of analytical models proposed in the literature”. Following the same form as previous models, the model is given by:

\[
Q = Q_u \left( 1 - e^{-\frac{1.3s}{s_0}} \right)^{0.5} + 0.0045s \tag{4.6.4}
\]

In more recent publications, simplified multi-linear models tend to be used, rather than the curved relationships proposed in the literature. In all cases the curves by Molenstra and Johnson are used as a reference case, from which simplifications are made. Queiroz et al.\textsuperscript{[139]} \textsuperscript{[140]} built a simplified parametric model of a composite beam, using non-linear springs for representing load-slip behaviour. The model was calibrated against a large number of beam tests recorded in the literature, and was found to be in good agreement.

Queiroz et al. performed a parametric study to understand the effect of variations in the load-slip relationships assumed. The parametric study included all of the load-slip relationships described thus-far in this section. The authors find that a precise definition of load-slip behaviour is not warranted, on account of the inherent variation in stud behaviour and conditions from test to test. The authors also raise the point that push-testing is not necessarily representative of the behaviour of studs in a beam, since the boundary conditions of the push-tests, which are close to the stud to allow the test to be performed on reasonably proportioned testing apparatus, are not the same as those found in a fully constructed beam. The authors eventually conclude:

“the importance of the load–slip curve is mainly to calibrate the structural responses [of the model] against [beam tests] available in the literature. The load–slip curve has no significant effect on the final results/conclusions”

Figure 4.25 compares the curves proposed. The case presented is a standard 19mm stud embedded in C30 concrete.
The results show that the curves have varying stiffness levels in the initial 1mm of deformation, though there is consistency in shape. Ultimate stud capacity tends to be consistent. The curve by Oehlers and Coughlan shows some deviation from the others, but this is as a result of failure modes not relevant to SCS construction, as discussed earlier.

Based on this literature search, two conclusions are reached:

- The results of push-tests can show considerable variation depending on the test arrangement used, material strengths etc., but there is a strong consensus as to what is a representative curve for load-slip behaviour. All the curves presented in the literature show strong agreement for typical cases.

- Researchers such as Oehlers and Coughlan and Queiroz et al. who have used non-linear springs to model composite beams have found their results tend not to be sensitive to the load-slip relationship assumed. Strong agreement with beam tests is usually achieved, in respect to ultimate resistance and end-slip.
The conclusions of this literature search apply only to modelling of conventional composite construction. To extend the conclusions to SCS panels, further consideration must be given to the differences between conventional composite construction and SCS panels, with respect to stud behaviour. This is explored in Section 4.6.3.

4.6.3 Expected differences between load-slip curves for studs used in conventional composite construction and studs used in SCS panels

A literature search, as described in Section 4.6.2, has found a large number of potential load-slip curves that may be used to model stud behaviour. These relationships have been successfully applied in several studies of composite beams, and have been found to give good results. However, differences exist between conventional composite construction and SCS panels that mean these relationships are not necessarily applicable to SCS modelling without further consideration, as discussed in Section 4.6.4. Test evidence does exist that explores the effect of small plate sizes and additional confinement with respect to conventional studs. In both cases these parameters are found to effect the results, but the conventional load-slip relationships are still found to hold.

Hicks and Smith\cite{79} describe the results of comparisons between stud behaviour in push-tests and behaviour of studs in real beams. The authors propose that the standard push-test underestimates the ductility and resistance of studs due to the lack of force normal to the shear plane of the stud, which is inherently present in beams due to self-weight and any applied load. The authors suggest that this force has its largest effect at the interface between the steel flange and concrete, leading to increased friction and stabilisation of the decking rib. However, the test arrangement applies the additional lateral force through the concrete, and will therefore inevitably lead to an increased level of confinement around the stud. The results presented show the effects that can be expected from increased confinement; a small increase in resistance, and a substantial increase in ductility. Stud resistance is maintained at high slips (greater than 10mm), far in excess of the 6mm slip capacity required by the Eurocodes.

The same push-test arrangement is applied by Nellinger et al.\cite{125}. The paper explores a number of different mechanisms by which addition of lateral load increases resistance and ductility (“These stresses effect the crushing of the concrete in front of the stud, as higher stresses can be reached in multi-dimensional compression”). Section 6.2 of the paper is dedicated to discussion of the effects of a “multi-axial stress state” on the results. These results show that confinement results in a 3% enhancement in
resistance. An 8% reduction in displacement capacity is observed, but the 60mm slip reached is far in excess of the 6mm slip capacity required by the Eurocode.

In both cases the push-tests were conducted on slabs constructed with profiled steel sheeting. This difference does alter the stress state and behaviour considerably, due to the increasing ability of the deck rib to rotate about the base of the stud, which acts as a fulcrum. However, these tests do indicate that a small increase in resistance (in the order of 5-10%) is the only significant effect of confinement on shear stud behaviour. It can be stated with reasonable confidence that the additional confinement present is SCS panels will cause only small deviations from the existing models for load-slip behaviour.

The other significant difference is plate thickness. Goble\textsuperscript{[74]} conducted a series of tests to understand the behaviour of thin flange push-out specimens. In 13 of the 41 test specimens ‘flange pull-out’ failure was observed, where the flange becomes incapable of sustaining the stress induced by the stud shear. The test series includes 41 specimens with 12.7mm, 16mm, and 19mm diameter shear studs, with flanges ranging from 3.2mm to 11mm. Goble subsequently developed a design model, which predicts that flange pull-out occurs when the flange is less than 2.7 times the diameter of the stud. Based on the standard 19mm diameter stud, this means SCS panels constructed using 8mm or greater plates should not be at risk of flange pull-out. Of greater significance to the modelling, Goble finds that the effect of decreased flange thickness on load-slip behaviour is negligible, with only the thinnest specimens showing a slight increase in flexibility in the early stages of loading. Further corroborating evidence for this conclusion is described by Rambo-Roddenberry\textsuperscript{[144]}.

The evidence presented suggests that there should be no fundamental difference between the behaviour of studs in conventional composite construction and in SCS panel systems. Such a conclusion is supported by engineering judgement, given there are no fundamental differences that would alter the mechanics of the deformation mechanisms. Correlation between the FE models utilising the load-slip relationships suggested are good, as can be seen in Section 4.7.

### 4.6.4 Push-test evidence for tie-bars

Since the bars are continuous throughout the depth of the panel the degree of restraint against rotation at the top of the stud can be expected to be greater. These bars may also be carrying axial forces, especially when working in shear. It is unclear what effect this might have on the load-slip behaviour, if any.
While the studs in an SCS panel can be expected to follow established models, it is not so certain that the tie-bars will work in the same way. Key differences between tie-bars and studs include:

- **Increased embedment depth** - Tie-bars extend throughout the depth of the section, meaning the length over which they resist any shear forces can be much larger than the typical height of a stud (100mm). Since the pressure on the connector per mm height decreases with depth (for a fixed shear force), taller connectors should be stronger. This effect is recognised in the Eurocode, but the enhancement stops at a $h/d$ ratio of 4, which is considerably exceeded for a tie-bar.

- **Additional tensile force** - Continuity of the tie-bar throughout the section means that the bar can develop considerable tensile force, either from interfacial shear itself or from other mechanisms, such as out-of-plane shear. This tensile force may have a positive effect in shear, since it may help prevent rotation at the base of the stud (in a similar mechanism to the one discussed by Hicks and Smith\textsuperscript{[79]}). Conversely, the coexistent stress state may prove detrimental to performance.

Figure 4.26 shows an exaggerated plot of the slip deformation that can be expected to occur as the SCS system is loaded. From these shapes, Figure 4.27 shows the expected pressure profile on a tie-bar compared to a stud. As shown, the pressure at the base of the connector is much larger than the pressure higher in the connector, and as such is more critical to the connector performance. Aside from the additional tensile force the pressure distributions are expected to be mostly the same, and should therefore behave in a similar manner.

![Figure 4.26: Deformation of studs and tie-bars](image-url)
Tie-bars are not used in conventional composite construction, meaning there is much less push-test evidence available in the literature.

One of the first publications to describe push-testing on tie-bars is by Clubley et al.\cite{48}. This paper describes push-tests on Bi-steel tie-bars, which are friction welded. The test arrangement pushes a concrete core in two directions from an internal jack. The bars are confined on all sides with additional studs included to increase confinement. It is not clear if measures were taken to reduce friction with the side walls.

11 tests are described by Clubley et al.\cite{48}. However, problems with the loading arrangement led to a considerable reduction in resistance from the accepted models. The authors suggest that the 400mm and 700mm specimens tested were not subjected to the conditions required to give an effective result. 7 tests are therefore removed from further consideration. Figure 4.28 shows these tests against the behaviour predicted by Molenstra and Johnson.

---

Figure 4.27: Indicative force and pressure distributions on stud connectors and tie-bars
The results show the load-slip behaviour of the tests is consistent with the model. Three of the four cases show stiffer behaviour than the prediction, but are still within an acceptable level of accuracy.

In all cases the tie-bars did not reach the typical slip capacity of 6mm. It is not clear from the text whether the tests were stopped prematurely due to the peak load being reached, or whether failure of the stud occurred. Two of the tests were conducted on 8mm plates, which testing by Goble\textsuperscript{[74]} suggested may have problems with premature failure. The text also suggests an issue with poor weld quality, though this may be exacerbated by the friction welding technique employed.

The only other test series conducted on conventional tie-bars is by Xie et al.\textsuperscript{[196]}. Again, these tests are conducted on the Bi-steel system, utilising friction welded bars. However, this series includes confinement by steel plates only on two sides. Stabilisation of the specimen on the open sides is provided by hardboard batons, but this is very unlikely to provide much confining effect. 24 tests are conducted in total, with the main variable being the thickness of the plates. Plates between 8mm and
15mm are included.

Previous results have shown the susceptibility of samples with thin-plates to suffer tear-out failure at the weld. Xie et al. again find that this is the predominant failure mechanism for thin-plates. For thicker plates the bars show acceptable performance (full resistance sustained at 6mm slip) in all but one case. Figure 4.29 shows the 12mm tests against the behaviour predicted by Molenstra and Johnson, while Figure 4.30 shows the 15mm tests.

![Figure 4.29: Comparisons between load-slip predictions by Molenstra and Johnson and tests on samples with 12mm plates by Xie et al.][196]
In both cases the load-slip shows general agreement with the predictions by Molenstra and Johnson. In both cases the predicted resistance in shear is exceeded. The resistance of the 12mm plate tests is higher than the 15mm, though the two tests predictions are within acceptable experimental bounds. It is likely change can be attributed to an inconsistency in the test set-up, for example misalignment of the loading jack or extra friction. In all cases the initial stiffness prediction is higher than observed in the testing. Consideration of the connector behaviour does not suggest a reason for this loss of stiffness, so it is expected the effect may be due to small deformations (‘setting in’) of the test rig. The test arrangement used by Xie et al. is likely to be more flexible than the one used by Clubley et al., due to the lack of continuity around the concrete core leading to less restraint against plate buckling. Clubley et al. did not observe the same loss of stiffness.

A final series of push-tests is recorded in Koukkari and Fülöp\cite{98}. This test series is arranged using a rectangular hollow section with bars through the centre, with the concrete then being pushed out through the section. The results of the tests show
resistance far in excess of the resistances predicted by the standard equations, with resistances 3-4 times greater than the equations suggest. Further interrogation of these results show that no steps were taken to reduce friction with the side walls of the RHS. An attempt was made to account for the friction by subtracting a reference test that contained no shear connectors, but this lead to negative forces in the initial loading region. Given these problems, these tests are not considered to be useful in understanding the behaviour of the bars, and are therefore ignored.

Based on the testing available, the following conclusions are reached:

- Tie-bars perform in a similar way to ordinary shear studs when loaded in shear. No significant deviations between existing models for load-slip and the tests are observed.

- There is some evidence that tie-bars may be slightly less stiff in the initial stages of loading. However, this may be a result of the push-test setup, rather than additional deformation of the bar.

- Tie-bars are also susceptible to premature failure when welded to thin plates. Minimum thickness ratio rules for studs should be respected.

It is proposed for further work that tie-bars will be modelled using the same load-slip relationships used for ordinary shear connectors. It is also clear that the test evidence available is not high quality, which suggests that further testing of tie-bars in shear is recommended for future researchers.

Debate has arisen about whether tie-bars should be included in when accounting for degree of shear connection (as per Equations 4.1.7 and 4.1.8). The push-test evidence that is presented in this section suggests that tie-bars have an almost identical load-slip curve to conventional shear connectors. Application of the design rules that include tie-bars also shows that exclusion of tie-bars leads to extremely conservative predictions for bending resistance when compared to tests. However, it has also been recognised that the utilisation of the bars at the construction stage to hold the steel plates together when fresh concrete is added to the core generates significant tensile stresses, which can remain ‘locked-in’\[65\]. It is possible that these locked-in stresses may prevent the shear connectors reaching their predicted ultimate capacity. This area is a key topic for further research, as discussed in the conclusion (Chapter 8). However, in the absence of any evidence of tie-bar tension-shear interaction failure, it is recommended that the shear strength of tie-bars is included when calculating degree of shear connection.
4.7 Comparison of numerical model predictions with test results

This section presents comparisons between the model described in Section 4.4 and a number of beam tests from the test database, described in Chapter 3.

The beam tests presented below were chosen to study a variety of the potential behaviours and failures that can occur. In particular, beam tests with reduced degrees of shear connection were included, since the beam spring shear connector technique described in Section 4.4.4 has a much greater effect on tests of these type. Verification against tests with stiff shear connection is unlikely to properly test the appropriateness of the technique, since the studs will not be highly stressed.

4.7.1 Parametric assembly of models

Since this investigation has required a large number of FE models to be constructed, efficiency has been achieved by parametrically constructing the ABAQUS models. This is achieved through the use of ABAQUS Python environment, which allows any feature that would be enacted through the graphical interface to be entered as a command.

The assembly code is split into three parts. Appendix A and C accept a set list of parameters, designed such that they cover the full range of beam tests that can be found in the literature. Appendix B gives these parameters for the tests described in Chapter 3.

The first part takes parameters from a geometrical description of the test and turns them into lists that can be used to assemble the model. In addition to geometric descriptions of the plates, concrete and shear connectors, these lists also interface & contact descriptions and loading information. This part of the script is given as Appendix C.

The final part of the code takes the geometric description of the problem and builds the model. The modelling described in Section 4.4 is implemented here. The script is given as Appendix A.

4.7.2 SCIENCE SP1

SCIENCE SP-1 is a beam test performed as part of the SCIENCE project, and is described in the test report by Koukkari and Fülöp\textsuperscript{[98]}. Figure 4.31 shows an illustrative diagram of the loading and stud arrangement.
The test itself has a very stiff shear connection, and fails by bending. The test is therefore relatively simple to model compared to the other tests in the sample, since the behaviour is controlled mostly by the mechanical properties of the steel plates, which are well understood.

Figure 4.32 shows comparison between the FE model and the test result.

Figure 4.32 shows that the correlation between the FE model and the test is almost perfect up to the point of failure. As discussed previously, the relative simplicity of the failure mode in this test makes such correlation possible.

The model assuming no confinement shows an abrupt loss of resistance at around
50mm deflection. This drop in resistance is linked with buckling of the compression plate, which results in a sudden transfer of compressive force from the steel to the concrete. Comparisons between the test and the FE model show that this failure mode is very well predicted; this is shown in Figure 4.33 and Figure 4.34.

Figure 4.33: Compression of plate buckling mode of SCIENCE SP-1, taken from Koukkari and Fülöp [98]
In the case where confined concrete properties are assumed the concrete is capable of resisting this newly transferred force, meaning only a small drop in moment resistance is seen when buckling occurs, which can be attributed to the reduced lever arm of the concrete stress block. The point of plate buckling is predicted to occur at around 65mm deflection.

The test appear to have been conducted under load-control, meaning comparisons between the curves once the peak resistance has been achieved are not appropriate. However, the test report is clear that buckling of the compression plate is not accompanied with catastrophic failure of the entire beam, as predicted by the model assuming no confinement. This would suggest that the model assuming confinement is more appropriate.

### 4.7.3 Parametric study of different load-slip curves

As discussed in Section 4.6.2, four load-slip curves have been identified from the literature (2 from Molenstra and Johnson\cite{118} and 1 each from Queiroz et al.\cite{139} and Oehlers and Coughlan\cite{132}). These curves are shown in Figure 4.25. It can be seen that the curves by Molenstra and Johnson are middle predictions, with the curve by Queiroz et al. being relatively stiff, and the curve by Oehlers and Coughlan showing more flexibility. This section explores the effect of variation in the load-slip behaviour assumed in the model on the overall predictions of the model, regarding strength, stiffness and end-slip.

Despite its importance in assessing composite action, only one set of tests explicitly

---

**Figure 4.34**: Prediction of compression plate buckling of SCIENCE SP-1 by the FE model
records end-slip. These tests are described by Koukkari and Fülöp[98] . Unfortunately, these tests were designed using relatively high degrees of shear connection, meaning recorded slips in all but one of the tests are relatively small i.e. less than 1mm. Slips of this magnitude are not useful for calibration purposes, since the measurement error of the recording devices is not insignificant when compared to the absolute value of the slips recorded, leading to high variation.

The only test that showed significant slip was SP3, end 2. This test was modelled using the modelling approach described in Section 4.4. A separate analysis was performed for each of the load-slip curves. Figure 4.35 shows the prediction of overall load-deflection response of the test compared to the recorded results.

![Figure 4.35: Prediction of load-deflection response of SCIENCE SP3-E2 using a number of shear stud load-slip relationships](image)

It can be seen that the predicted stiffness from each of the models is almost exactly the same for each of the stud-models. There is strong agreement with the results from the tests.

The change in load-slip model has an effect on the prediction of ultimate resistance. In all cases a shear failure is observed, consistent with the results of the test which
contains relatively low amounts of shear reinforcement. The stiffer stud models result in an earlier prediction for the point of failure, which can be explained by the ‘Kani’s valley’ effect, as discussed in Section 5.3. However, in each of these cases the load at which the crack is predicted to form varies by less than 10% of the total load, which is still relatively accurate by modelling standards. In all cases the model is conservative when compared to the test. This is typical of FE models for shear, which tend to be more brittle than the behaviour in testing.

Figure 4.36 shows the predictions for end-slip. It can be seen that the absolute value for the slip at the point of failure varies considerably, with the stiffer models predicting end-slips in the order of 1mm, while the more flexible models predict end-slips in the order of 3mm. In terms of this prediction, the model is therefore extremely sensitive to the assumed load-slip behaviour. It is particularly notable that the two models by Molenstra and Johnson, which show little difference in prediction of load-slip result in considerably different predictions for end-slip.

However, it should be noted that prediction of end-slip for a given load is not particularly relevant to design. Instead, the typical requirement is that end-slip should be limited to an absolute value of 6mm. In this sense, all of the models are showing a similar prediction for the load at which excessive slip would occur, since each of the end-slip curves plotted is showing a plateau at around 1700kN. This suggests that predictions of the point of excessive end-slip is far more sensitive to the ultimate resistance of the connectors, which is kept constant in these models.
While explicit measurement of slip is not available in any other test series, the effects of any error in load-slip estimation should be reflected in the overall load-deflection curves of other tests. Tests with low degrees of shear connection should show the most deviation.

Varma SP1-2\cite{184} is recorded as an interfacial shear failure. The layout of studs was deliberately designed in this case to ensure this failure mode occurred, in order to test the bounds of acceptable levels of shear connection. As such, it is the best candidate for observing any changes resulting from assumptions regarding the load-slip behaviour. Figure 4.37 shows the load-deflection predictions for 4 different models for connector load-slip behaviour.
It can be seen that changing the assumed load-slip relationship has an appreciable effect on the overall load-deflection behaviour, but the change is small. There is no change in the prediction of the ultimate resistance, which reflects the fact that the ultimate stud resistance is constant for all of the models. The load-slip behaviour directly influences the load-deflection relationship, with the stiffest stud model (Queiroz et al.) giving the stiffest load-deflection response.

The predictions for end slip are also reasonably consistent for all of the load-slip models. The predictions are shown in Figure 4.38.
Figure 4.38 shows that the prediction of the load at which 6mm slip occurs is constant in all cases. Again, this reflects the consistent assumption of the failure load of the individual connectors. Deviation in the slip prediction does occur, but only in the 1mm to 2mm slip range, where the differences between the assumed behaviours are most acute. Prediction of slips of these magnitudes is desirable, but not essential, for design purposes.

The results of these comparisons studies shows that the techniques presented for modelling of the shear connectors produce good predictions for the slips observed in testing.

The results of the sensitivity studies show that the assumptions made regarding load-slip behaviour of the connectors do not particularly affect the outcome of the modelling. In both of the cases looked at, consistent results are obtained for both the ultimate failure load and the whether slip failure (6mm of greater slip) occurs. This conclusion is supported by additional parametric studies that are not recorded herein, and is consistent with findings by other researchers of composite structures, including Queiroz et al. [139] and Ollgaard et al. [135].
With respect to these conclusions, any of the load-slip curves presented in this section will likely lead to acceptable results, meaning the choice of curve is somewhat arbitrary. Given its long history, and successful use in a number of studies, it is decided that the Equation 4.6.2, as proposed by Molenstra and Johnson, is used for all further modelling.

4.8 Understanding the distribution of stud forces in the span using finite element analysis

As described in Section 4.4, a finite element model has been developed for predicting the performance of SCS panels subject to out-of-plane loads. The model shows strong correlation with test results, and is capable of predicting the point of ultimate bending failure, the panel stiffness and the interfacial slip. The model can therefore be used for further prediction and parametric study with a high degree of confidence.

A key advantage of the numerical model that has been developed over testing is that the force in each individual shear connector can be output over the loading time-domain. This level of detail is impractical for a test, because (1) stud connector force is impossible to measure using any available measurement technique; and (2) the number of studs in a typical test would require an impractically high number of measurement apparatus.

It is through the measurement of individual stud forces that the existing assumption of a ‘smooth’ distribution of force at the shear connection interface was found to be incorrect, on the tension plate. Instead, the model shows that flexural cracking causes discontinuities, that result in changes in the behaviour. As discussed in Section 4.9, this can lead to unconservative design predictions for bending in some circumstances.

The only other work to explore this misconception about stud force distribution is the thesis of Foundoukos. In his work (Chapter 5.7), modelling using ABAQUS is again used to suggest the presence of slip discontinuities. However, the focus of this work is only on the prediction of slip; there is no discussion on the potential influence this effect might have on resistance. This topic was not included in any further publications by the author, suggesting the effect was viewed with minor importance. The thesis itself is not widely available.

Section 4.8.4 and Section 4.8.3 show the evolution of stud force for a demonstration case. Section 4.8.4 shows the distribution of stud force on the compression face, which demonstrates that the cosine based model used by Lawson et al. for conventional
composite beam design works well, though the contribution of the concrete to resisting the applied moment is underestimated. Section 4.8.3 shows the results for the tension face, which show the distributions are not correctly predicted by the cosine model. The implications of this on design are explored in Section 4.9.

4.8.1 Existing understanding of stud force distribution

Existing understanding of stud force distributions are currently taken from understanding of conventional composite beams. As suggested previously, stud force is difficult/impossible to measure in a test, so there are no definitive measurements. Despite this, a large number of analytical studies of end-slip predictions for whole beam tests have been performed, and the results are generally found to strongly correlate with the analytical models. The assumptions are also supported by numerous finite element studies[^5].

Papers concerning this subject tend to refer to work by Newmark et al.[^126]. Despite its extensive citation, this paper is not readily obtainable. Instead, the derivation can be found (in more detail according to Newmark) in the appendix of a separate report[^160]. It should be noted that the equations presented are difficult to solve, even for the single point-load case for which the derivation is presented. Ranzi and Zona[^145] present a solution for uniformly distributed loads. The most recent work on this topic, by Lawson et al.[^102], considers both the previous works. The final derivation bases the stud force distribution on a cosine function, as below:

\[
\bar{F} = \int_0^x \frac{k S_{sc}}{S_{sc}} \cos \left( \frac{\pi x}{L} \right) \quad (4.8.1)
\]

Sections 4.8.3 and 4.8.4 compare the force distributions predicted by this model to the forces calculated by FE modelling, for the tension and compression sides respectively.

4.8.2 Demonstration case

In order to demonstrate the deviation of the true stud distribution from the predictions, a test case is demonstrated.

The test case presented has been designed specially with a low degree of shear connection, in order to highlight the behaviour. It is not expected that such a design would ever be specified by a designer. However, the effects of the behaviour observed in this test case do impact the design of at least one specimen found in the literature review; this is discussed further in Section 4.9.2.
Figure 4.39 shows an indicative diagram of the test case.

![Figure 4.39: Indicative diagram of cracking induced slip demonstration case](image)

Figure 4.39 shows the tie-bar layout in the transverse direction.

![Figure 4.40: Indicative diagram of position of tie-bars in the demonstration case](image)

When designing this case, the aim was to ensure that slip failure occurs before either shear or cross-sectional bending. To produce this, the design of has the following features:

- **Large Height** - A relatively deep cross-section enhances both out-of-plane shear resistance and cross-sectional bending resistance. The test case is set at 1200mm, which is larger than most of the cases in the literature, but still within acceptable bounds.

- **Relatively high-strength concrete** - Higher strength concrete mostly enhances shear resistance. Shear connector resistance is not increased in this case, as the strength is governed by the strength of the connector steel. $40 \text{ N/mm}^2$ concrete was used.

- **Thick, strong plates** - Strong plates give a reduced degree of shear connection, as per Equation 4.1.7, and enhance cross-sectional bending resistance. However, it is important that the plate reflects practical steel grades and available thickness’s. The design uses 12mm S355 plates.
• Tie-bar reinforcement only - The design does not include any shear studs; instead, all interfacial shear resistance is provided by tie-bars. This is chosen for two reasons:

1. Tie-bars provide both interfacial shear resistance and out-of-plane shear resistance.
2. More superficially, including only one shear connector type removes the effect of differential allocation of force due to connector stiffness differences. This makes plots like Figure 4.41 easier to understand.

The bars used were S355, 24mm in diameter.

The calculated degree of shear connection, measured to the centre of the beam, is 83% on both plates. This design is therefore within the bounds of acceptability of the SCS design manual[4].

Section 4.8.3 presents the development of the force profile on the critical tension side. Section 4.8.4 presents the compression side.

4.8.3 Tension face

Figure 4.41 shows the end slip predicted by the FE model for the example case. As predicted, the design failures by excessive slip at the interface on the tension side. Figure 4.41 focuses on the slip up to 3mm.
Figure 4.41 shows that a significant change in the stiffness of the tension plate shear connection occurs at around 1750 kN of applied load. This drop in stiffness is associated with the development of a tension side crack in the concrete, which affects the distribution of forces in the studs.

Figure 4.42 shows the stud forces from the FE model against theoretical predictions. The plots show the total stud force between the nearest support and a given location within the beam (blue points), at the failure load (defined as 2500 kN, as discussed in Section 4.8.2). The orange line shows the theoretical maximum total stud capacity, assuming each stud reaches its own resistance. The green line shows the stud force required to satisfy local cross-sectional bending equilibrium.
It can be seen that the stud forces from the FE model (shown in blue) generally follow the demand on the cross-section, as per the applied bending moment. However, it is also clear that not all the studs are fully mobilised, as suggested by previous design models (discussed in Section 4.8.1).

The plot suggests that satisfaction of local cross-section equilibrium is not the only mechanism driving the distribution of force in the studs. The studs placed at locations close the supports tend to be fully utilised, to an extent not predicted by the traditional model (shown on Figure 4.42 as a green line).

Figure 4.43 shows the stud force distribution before the formation of the first cracks, while Figure 4.44 shows the distribution afterwards.
Figure 4.43: Concrete plastic strain (above) and connector force distribution (below) before the formation of the first tension side cracks

Figure 4.44: Concrete plastic strain (above) and connector force distribution (below) after the formation of the first tension side cracks

Figure 4.43 shows the initial stud force distribution follows a smooth profile. Since the concrete is carrying much of the bending force in tension, the stud forces are not as high as those observed in Figure 4.44.
Once the first group of tensile cracks form, a significant change in the force profile occurs. Discontinuities form at each crack location. The stud forces also significantly increase, since the tensile capacity of the concrete in bending is no longer utilised. Since stud force is proportional to slip in the initial stages, the slip also increases suddenly, which can be seen in Figure 4.41.

The formation of cracks leads to ‘segmentation’ of the span. Each of the 3 segments formed is now essentially in it’s own equilibrium. Further segments are formed as additional cracks develop. Figure 4.45 shows the stud force distribution before the formation of the second group of cracks, while Figure 4.46 shows the distribution afterwards.

**Figure 4.45:** Concrete plastic strain (above) and connector force distribution (below) before the formation of the first tension side cracks
Once segmentation occurs, only studs within the segment are utilised in resisting the longitudinal shear forces, and therefore preventing excessive slip. Figure 4.46 shows the slip distribution at the point of excessive slip failure i.e. 6mm end slip. The discontinuity in the slip profile is readily apparent.

In all of the cases modelled, cracks tend to initiate at the locations of shear connectors or tie-bars. This effect is expected, since these locations are the most heavily stressed. For cases with UDLs the crack patterns are consistent, with the first being formed at the point of maximum moment, while the second tends to form at roughly the quarter-
The implications of these slip discontinuities on design is explored in Section 4.9.

4.8.4 Compression face

The force distribution on the compression side is also poorly approximated by the co-sinusoidal rule. Figure 4.48 shows the cumulative force profile for the studs on the compression plate.

![Cumulative stud force, summated between location and the nearest support, at the failure load of 2500kN](image)

The plot suggests that the force distribution up to a quarter of the span follows the force distribution required for equilibrium of the cross-section. Between the quarter-spans the studs contribute very little, if any, force, which can only occur if the concrete is significantly stressed. Cross-section equilibrium is then satisfied by a concrete stress block only.

Evidence from other tests, such as SCIENCE SP1 (as discussed in Section 3.2.13), suggests that the plate tends to be loaded before the concrete, which typically leads to buckling of the plate. In the case shown in Figure 4.48, it is likely that the decreased level of shear connection has led to a higher proportion of slip at the interface compared to tests with higher levels of shear connection (such as SP1). As the plate slips, the concrete becomes loaded.

In many ways, understanding the behaviour of the compression plate is superfluous, since failure by excessive slip or loss of equilibrium will always occur in the tension plate before the compression plate. While this statement is only valid for designs that
have equivalent stud layouts on either face, this is rarely not the case in modern
designs. This is even the case where compression plate buckling occurs, since the
concrete working on its own is typically more than capable of providing cross-section
equilibrium up to the point of failure.

Figure 4.49 shows the load-end slip relationship for the compression plate. It can be
seen that the maximum slip observed on the compression plate was 0.7mm, which is
considerably short of the 6mm slip that constitutes failure.

![Figure 4.49: Load vs. end-slip for the compression plate of the demonstration case](image)

### 4.9 Implications of slip discontinuities on design

The presence of slip discontinuities in the span affects the ability to predict both
end-slip and ultimate resistance. In both cases, the model presented in Section 4.1
tends to over predict resistance, since the degree of shear connection calculated in
accordance with Equations 4.1.7, and used in Equation 4.1.4, relies on all of the studs
in the span being fully utilised, which FE modelling shows is not the case.

Unconservative predictions are demonstrated in two cases. The first case is the
demonstration case used in Section 4.8.2. The second case is Roberts et al., as
described in Section 4.9.2.

### 4.9.1 Demonstration case

Figure 4.50 shows the load-deflection response of the demonstration case, used to demonstrate the stud force distribution in Sections 4.8.3 and 4.8.4. As described in Section 4.8.2, this case was designed with a relatively low degree of shear connection, which leads to a characteristic curved response in the load-deflection curve.

![Load-deflection response](image)

**Figure 4.50**: Predicted load-deflection response of the demonstration case, from the finite element model, against predicted resistances assuming critical cross-sections at quarter and mid-span

Figure 4.50 shows two horizontal design lines. In both cases, the resistance of the cross-section is predicted using Equation 4.1.1, but different locations are taken for the critical cross-section. To produce design line ‘R,0.5 Span’ the critical cross-section is taken at the centre of the span, as per traditional design. The design line ‘R,0.25 Span’ is produced by taking the critical cross-section at a quarter of the span, which can be seen in Figure 4.45 to be the location of the critical segmentation crack. In both cases, the calculated degree of shear connection is less than 100%; 82% for the mid-span design and 41% at the quarter-span.

It can be seen that the traditional approach of taking the critical cross-section at the
centre of the span produces an unconservative result for this case. The design load predicted is around 30% is excess of the result predicted by the FE model. Taking the critical cross-section at the quarter-span produces a more realistic and conservative value compared to the FE model.

The reason for this conservative prediction can be seen when the bending resistance envelope is plotted. This can be seen in Figure 4.51.

![Figure 4.51: Bending resistance envelope for the demonstration case](image)

The green line shows the bending moment generated by a 4193kN total uniformly applied load. The bending resistance envelope touches the moment profile only at the exact centre of the beam. It should be noted that the envelope produces a spike at this location because there is a centrally located tie-bar, which is included when summating the available shear connectors. Inclusion of this row of connectors is debatable, but even if it were excluded, the predicted resistance would still be unconservative. For the rest of the span, the applied moment significantly exceeds the resistance.

The red line shows the bending moment generated by a 2624kN total uniformly applied load. In this case, the applied bending moment profile roughly follows the resistance envelope. In some locations (between \( x = 200 \) and \( x = 2000 \)) the envelope is exceeded. However, cross-section equilibrium is maintained by catenary action of the bottom plate between studs; only small forces may be developed in this way, but this is enough to maintain equilibrium.

It should be noted that the beneficial effects of arching action between the supports and the closest load points is not accounted for in this analysis. While arching action
is more critical for accurate determination of shear resistance (as discussed in Section 5.3.2), direct transfer of the load to the support also occurs. It is this effect that has produced the conservative prediction for the design at the quarter-span seen in Figure 4.50.

4.9.2 Test B3 by Roberts et al.\textsuperscript{[151]}

Very few of the tests in the database of tests described in Chapter 3 have a low enough degree of shear connection to be effected by the misidentification of the critical cross-section. In addition, most of the tests in the database utilise four-point bending to produce the desired bending profile; as shown in Figure 4.52, the linear nature of the bending moment profile tends to result in less relative separation between the resistance and applied moment at the worst cross-section.

Of the tests in the database that are potentially affected by this issue, the test that shows the clearest indication of an effect is the test B3 by Roberts et al.\textsuperscript{[151]}. This test utilises 4 point loads to produce the bending moment, which means that the bending moment diagram is a close approximation of the bending moment diagram for a UDL. The degree of shear connection is also low; 43% at mid-span. As with the previous example, the difference in applied moment profile based on the two assumptions can be seen in Figure 4.52.

![Figure 4.52: Bending resistance envelope for Test B3 by Roberts et al.](image)

When compared to the recorded failure load of 315kN, design at the centre of span produces an unconservative result; the model predicts a failure load of 373kN. As before, design at the quarter-span produces a conservative result i.e. 196kN. Again, accounting for arching action for the closest loads would improve the precision of this
Evidence is found in the crack pattern presented for this case of ‘segmentation’, which is critical to understanding the behaviour of the shear connection, as discussed in Section 4.8.3. Figure 4.53 shows the crack pattern taken from the paper:

Figure 4.53: Crack pattern for Test B3, taken from Roberts et al. [151]

Figure 4.53 shows 4 discreet cracks, occurring in two groups. The first two cracks occur at around 50% of the final load, at the points of maximum moment. These cracks are segmentation cracks, leading to three separate ‘blocks’ that must maintain their own equilibrium. Similar segmentation cracks are observed for a beam subject to a UDL, as seen in Figure 4.46.

The final segmentation crack occurs at the failure load. The crack occurs immediately underneath the closest point load to the support, as predicted in Figure 4.52. Formation of this crack leads to considerable slip, which appears to have allowed a shear crack to open, causing failure.

Figure 4.54 shows the load-deflection curve from the test and the FE model. Following the precedent used in Figure 4.50, two lines are shown for two different assumptions regarding the location of the critical cross-section. Design line ‘R,0.5 Span’ is produced by taking the critical cross-section at the centre of the span, as per traditional design. The design line ‘R,0.25 Span’ is produced by taking the critical cross-section at the location predicted by the resistance envelope, as shown in 4.52.
It can be seen that the traditional design line over-predicts the results from the test, though the predicted failure load is close. On first inspection, the refined model appears to be conservative to a large degree. However, analysis of the test results suggests that this enhancement can mostly be attributed to the beneficial effect of tensile membrane action in the bottom plate.

Tensile membrane action allows significant tensile stress to develop in the bottom plate. As loads are applied to the system, significant friction develops between the support and bottom of the plate. This friction anchors the plate in position at the support. The bottom plate then spans between the supports, developing resistance through axial tension rather than bending through the cross-section. Figure 4.55 shows a diagram of how this effect can occur.
Tensile membrane action is potentially significant in Test B3 for two reasons:

1. **Model scale** - Test B3 was tested at model scale, which results in a lesser failure load than other tests in the database. However, the supports were not scaled proportionally. This means that the friction force that can be developed before slipping, which is proportional to the area of contact between the plate and the support, is high in proportion to the failure load.

2. **Thick tension plate** - Test B3 has a relatively high $t/d$ ratio; the tension plate is around 5% of the total depth of the cross-section, compared to a typical value of 2% for other cases. A thicker plate can carry more tensile force.

Tensile membrane action cannot be utilised in design, as it relies on developing friction, which can be uncertain. It is also largely insignificant in most full scale designs. It is likely that the FE model has overestimated the resistance generated by this effect.

It should be noted that Roberts et al. recognised an issue with low degrees of shear connection leading to unconservative results for analysis of the bending resistance of the section. In a subsequent analysis paper\textsuperscript{[195]}, the same authors present a review of a number of test results, with the authors finding that the results are often unconservative. A figure from this paper is shown below:
In the subsequent discussion the authors attribute the loss of resistance to a reduction in shear connector capacity when utilised in tension. A 50% reduction factor on connector resistances is suggested for application in design. In typical designs, this reduction factor produces similar results to the design rules advocated by this author, since the distance between the critical cross-section and the support is usually around half of the distance between the support and mid-span. However, this is a misrepresentation of the mechanics of the failure.

4.10 End-slip and the minimum degree of shear connection

As discussed in detail in Section 4.2.2, limiting excessive slip in individual shear connectors is an important consideration in design, since excessive slip can cause both increased deflections and ‘unzipping’ failure. EN 1994-1-1[^29] limits slip to 6mm at any given shear connector (see clause 6.6.1.1) if plastic design principles are to be applied, as they are in Equation 4.1.1.

Slip is usually measured at the end of the beam, where slip is greatest. Excessive slip is prevented by provisioning more shear connectors, since the design force on each connector is then reduced, resulting in less deformation. The minimum number of shear connectors required to achieve a slip less than 6mm determines the minimum degree of shear connection. The design force is measured relative to the maximum possible force that can be taken by the plate before tensile failure occurs.

For conventional composite beams, the slip that can be expected to occur for a given design is a function of a large number of parameters, of which there are too many to
capture in a precise numerical expression. Instead, a recent publication by Lawson et al.\cite{102} describes how simplified bounded expressions can be developed that will ensure the slip criteria is met, as long as the beam is of reasonable proportions. This conclusion is backed by a large number of parametric finite element models of beams, with varying spans, shear connector spacings, beam sizes etc.

It was initially expected that a similar FE study would be required for SCS panels. However, results of initial models showed that excessive slip failure tended to occur even at high degrees of shear connection.

A fundamental difference exists between the mechanical behaviour of an SCS panel compared to a conventional composite beam that explains the increased slips in SCS panels for similar degrees of shear connection.

In a composite beam, compressive and tensile stress blocks can exist in the steel simultaneously, such that adequate resistance can be achieved even when the concrete is not being fully mobilised. In SCS panels, the tensile stress block is either linked to the capacity of the plate or the force that can be carried by the studs. If less than 100% shear connection is specified, it is inevitable that the studs will reach their capacity, as there is no other means of providing tensile force. If the studs reach full capacity, excessive slip will always occur at ULS.

It can be shown that any design where bending failure is critical that allows for less than 100% shear connection on the tension plate will fail by excessive end-slip, at ULS. Figure 4.57 shows the load vs. end slip relationship for the demonstration case, described in Section 4.8.2. The plot shows slips in excessive of 20mm, though it is likely that a test would show shear connector failure before a slip of this magnitude could be reached.
Some regulators insist that interfacial slip failure is an undesirable failure mode, and must therefore not be allowed in design, as it is considered non-ductile\cite{137}. If this criteria is to be met, the implication of the FE analysis presented above is that the minimum degree of shear connection for SCS panels should be set as 100%, if excessive slip is to be prevented entirely.

A new rule setting the minimum degree of shear connection to 100% will mean considerably more studs are required for many designs, compared to designs according to codes that have no specific MDOSC requirements, such as the Bi-steel manual\cite{24}. This is potentially harmful to the economy of the system. However, there is a significant caveat to this new rule that make it less onerous than the rules first appear; excessive slip tends not to occur in designs where bending is not critical. For beams with short spans, where it can be hard to fit in the required number of studs, it is often the case that shear resistance is the governing check.

Figure 4.58 shows a configuration of the demonstration case where the outer two tie-bars have been replaced by studs. The shear resistance of these studs is set to be equal to the shear resistance of the tie-bars, meaning there are no changes to the level of shear connection. The reduction in the number of tie-bars leads to a significant drop in the shear resistance of the panel, such that the shear resistance check is now critical. Figure
4.59 shows the resistance envelope, showing that the bending moment at failure is no longer touching the resistance line at any location.

**Figure 4.58:** Indicative diagram of position of tie-bars in the demonstration case, with reduced number of tie-bars

**Figure 4.59:** Bending resistance envelope for the demonstration case, with reduced number of tie-bars

Figure 4.60 shows the load-slip curve for this case. As expected, excessive slip does not occur at the design load.
Further parametric studies were unable to find a combination of parameters where excessive slip preceded any other failure mode. Bending resistance tends to be proportional to shear connection percentage when a lower degrees of shear connection are specified, meaning the calculated bending resistance also determines the point of on-set of excessive slip failure. Given bending failure and excessive slip failure occur simultaneously, there is then no need to perform a separate check for slip.

### 4.11 Increased stud density at supports

The proposed design rule requiring 100% shear connection at a quarter-span for panels subject to a UDL can be expected to lead designers to have to specify increased stud densities for these cases, compared to existing understanding, where shear connection is assessed at mid-span. This might be considered harmful to the economic efficiency of the system, as a greater number of connectors requires an increase in the amount of welding required, which can be expensive.

A potential solution that could decrease the density of studs required is non-uniform
studs layouts. The mechanical model has suggests that shear connectors close to the support are utilised to a greater extent than studs in the central portion of the slab when the panel is subject to a UDL. A route to better economy may therefore be to increase stud density near to the support, and to reduce stud density in the central portion.

To explore this possibility, two cases are compared below. The first case is identical to the demonstration case, described in Section 4.9.1. The second case is identical to the demonstration case, except that the number of studs per row has been decreased from 3 to 2 in the central half of the span, while the number of studs per row has been increased in the two quarters nearest the supports. A diagram of the stud layout is shown in Figure 4.61.

Figure 4.61: Plot showing the layout of tie bars for the non-uniform demonstration case. Direct comparison can be made with the uniform layout case, shown in 4.39.

Figure 4.62 compares the two bending moment envelopes, with the black line showing the case with the uniform layout of shear connectors, while the red line shows the envelope for the case with a non-uniform layout of studs.
It can be seen that there is a considerable difference in the envelopes, despite the fact the two cases use virtually identical numbers of shear connectors. The gap between the two lines is greatest at the quarter-span, which as described in Section 4.9 is the location where failure occurs. The analysis suggests that the non-uniform arrangement of studs should produce a considerable increase in moment resistance, given there are no points at which the green applied moment line touches the red resistance line.
It can be seen that there is a difference in ultimate resistance of the two cases, with the non-uniform stud layout producing a higher resistance as expected. Figure 4.63 shows that taking the critical cross-section at the quarter-span produces conservative results for both models, with around 20% extra capacity predicted by the FE model. In both cases the degree of conservativeness is the same, suggesting that the model is correctly accounting for the changes in stud layout (as opposed to the previous rules, which would have predicted both cases to have the same resistance). If the critical cross-section is taken at mid-span both FE models would be predicted to have the same resistance, which the plot shows is not correct. In both cases the prediction is unconservative.

Although difficult to quantify, much of the increased level of conservativeness can be attributed to arching action from the loads placed close to the supports. The effect is particularly acute in this case because the demonstration cases are deep, in order to increase the bending resistance (as discussed in Section 4.8.2). Arching action is discussed further in Section 5.3.2.
4.12 Design procedure summary

This section presents a design procedure that reflects the new insights regarding the location of the critical cross-section. The determination of the plastic resistance of the cross-section is unchanged from the model presented in Section 4.1.

1. For a given span, decide the dimensions of the panel, including depth, plate thickness’s and material strengths.

2. Place studs in the span. The minimum and maximum stud spacings can be found in the SCIENCE design manual[4].

3. Determine the possible locations of the critical-cross-section, in accordance with the guidance given in Section 4.9. For point loads, every load point should be checked. For UDLs, checks should be made at quarter and mid-span.

4. For each cross-section location, sum the resistance of each shear connector between the cross-section and the nearest support, for both plates.

5. Determine the degree of shear connection, using the following equation:

   \[ \mu = \min \left\{ \frac{f_y t}{f_y t}; nP_{Rd} \right\} \]  

   (4.12.1)

   If the value of \( \mu \) is less than 1, and the checking authority does not allow failure by excessive slip, the shear connector spacing should be reduced. Alternatively, the thickness or strength of the plate can be reduced.

6. Determine the bending resistance using either of the following equations:

   \[ M_{sag} = \left\{ +f_y b \left( \frac{t_b}{2} \right) \mu_b - f_y b t_c \left( h - \frac{t_b}{2} \right) \mu_t - f_c h_c \left( h - t_t - \frac{h_c}{2} \right) \right\} b_w \]  

   (4.12.2)

   \[ M_{hor} = \left\{ -f_y b \left( \frac{t_b}{2} \right) + f_y b t_c \left( h - \frac{t_b}{2} \right) + f_c h_c \left( t_b + \frac{h_c}{2} \right) \right\} b_w \]  

   (4.12.3)

7. Check the bending resistance against the applied bending moment. If the resistance is not sufficient at all locations, the panel should be re-proportioned.
4.13 Conclusion

This chapter has described an investigation of the bending resistance of SCS panels. Plastic cross-section analysis has been shown to work well in most cases, and is therefore justifiably the basis of all of the present design methods, both in Europe and the rest of the world.

Few instances of designs that feature lower degrees of shear connection were found in the literature. For those cases that did feature lower degrees of shear connection, existing rules designed to account for the lower degrees of shear connection were predicted not to give conservative results in many cases. To investigate the reason for this deviation, an advanced non-linear finite element model was developed to reproduce the test results. The model proved to have the desired level of accuracy. This model was then extrapolated to a number of test cases, which proved to have resistances less than the resistances produced by the plastic analysis based design model.

The key problem identified with the existing model was misidentification of the location of the critical cross-section. It was found that the distribution of stud force on the tension side of the panel does not follow existing design models, and is heavily affected by the presence of tension side cracking. These tension cracks lead to ‘segmentation’ of the span (as discussed in Section 4.8.3). This segmentation is not accounted for in the existing model, which tends to suggest that more studs are mobilised in the span than are actually mobilised.

The location of the critical cross-section can be identified by preparing a resistance envelope for the span, and example of which is shown in Figure 4.52. However, preparation of a plot like this is beyond the capacity of most designers, since it requires iterative resistance calculations along the span, which are a considerable undertaking when not automated.

The process can be simplified for most of the cases a designer will typically encounter. For panels subject to point loads, the critical cross-section tends to occur under the load, so each load location should be checked in turn. For uniformly distributed loads, the critical cross-section should be checked at mid-span and quarter-span. For more complex cases, such as UDLs and point loads in combination, a full resistance envelope should be calculated.

The implications of the new design rules on slip were explored. In all cases it is found that excessive slip failure always occurs if bending failure is the critical failure mode.
and the degree of shear connection at the critical cross-section is not 100%. This is a deviation from the behaviour of conventional composite beams, where the degree of shear connection can be as low as 40% without slip failure occurring\textsuperscript{[51]}.

The majority of designs used in nuclear power plants can be expected to have high degrees of shear connection, with the proportions of the panel being controlled by either in-plane resistance or construction stage checks, as shown by Tuscher\textsuperscript{[179]}. In these cases, it is expected that the findings of this chapter will not affect the design. For the majority of the cases in the test database, described in Chapter 3, no changes to the predicted bending resistance are expected. However, in cases where the designer wants to reduce the shear connection to reduce the amount of welding the designer may find a greater number of shear connectors are required to develop the required bending resistance. Although the new design rules are more conservative, the increased clarity gained through the work described in this chapter should allow designers to specify lower degrees of shear connection with greater confidence, knowing that the possible effects have been properly accounted for in the design rules.
Out-of-plane shear resistance

Shear failure represents one of the two ultimate limit states of SCS panels subjected to out-of-plane forces (the other being bending, as discussed in Chapter 4). Shear failure is known to be considerably more complicated to both model and produce design guidance for than bending, meaning models for shear resistance tend to be semi-empirical.

As with bending, models for the resistance of conventional RC beams have been applied to SCS panels\cite{24}, and have proved to be sufficient for design in Europe up to the current date. However, the existing Eurocode compliant model does not include any provision for changes in behaviour due to the effects of reduced shear connection. This chapter aims to address this limitation.

Firstly, the existing Eurocode compatible guidance is reviewed against the test evidence that is available, as described in Section 5.2. Limitations in the predictions are addressed, including discussion of the arching mechanism, which allows considerable enhancement in resistance against loads applied close to supports. Examination of the existing database also shows that insufficient test data was available to explore the effects of shear connection on shear failure, so additional data was generated through finite element analysis, as discussed in Section 5.4. Based on these results, a new adjustment to the Eurocode model is presented (See Equation 5.5.1).

Finally, a range of alternative resistance models are tested, with the limitations of each discussed.
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

5.1 Existing Eurocode compliant model

5.1.1 Panels without shear reinforcement

A Eurocode compliant resistance model for out-of-plane shear was first given in the Bi-Steel Manual\textsuperscript{[24]}. This model includes a contribution from friction welded connectors that are unique to the Bi-steel system and a contribution for the concrete acting alone. The concrete contribution was directly taken from ENV 1992-1-1, the prototype to the current Eurocode 2, and was based on the model presented in CEB-FIP Model Code 78\textsuperscript{[59]}.

Since publication in 1978 significant research has been undertaken on the behaviour of concrete beams loaded in shear. Based on this work, the shear resistance model for the concrete used in the Bi-Steel Manual has been superseded in the Eurocode by Equation 5.1.1, originally given in Section 6.2.2 of EN 1992-1-1\textsuperscript{[30]} (as described by Walraven\textsuperscript{[191]}).

\[ V_{Rd,c} = \frac{C_{Rd,c}}{\gamma_c} \left[ k \left(100\rho_l f_{ck}\right)^{1/3} \right] b_w d \]  

(5.1.1)

Where:

- \( V_{Rd,c} \) is the design shear resistance of the member without shear reinforcement
- \( \gamma_c \) is the partial factor for resistance of concrete in shear
- \( k \) is the size effect factor = \( 1 + \left(\frac{200}{d}\right)^{0.5} \leq 2.0 \) (\( d \) in mm)
- \( f_{ck} \) is the characteristic concrete cylinder strength (MPa)
- \( b_w \) is the effective width (taken as the full width \( b \) in SCS panels)
- \( d \) is the effective depth of the section
- \( C_{Rd,c} \) is an empirical factor; a lower bound of 0.18 is given by König and Fisher\textsuperscript{[96]}
- \( \rho_l \) is the longitudinal reinforcement ratio

For SCS panels, the Bi-Steel Manual\textsuperscript{[24]} defines the the longitudinal reinforcement ratio as:

\[ \rho_l = \min \left(\frac{f}{d} : 0.02\right) \]  

(5.1.2)

\( \gamma_c \) is generally taken as 1.5 for design of concrete beams in shear. This value is
relatively high when compared to those applied to structural steel or steel reinforcement bars, which reflects both the uncertainty with which the strength of the concrete can be accurately known and the precision of the model, both of which are accounted for by a statistical analysis\cite{191}. Development of a new partial factor for use with SCS panels is described in Chapter 7.

Additionally, the resistance cannot be less than the minimum value given by Equation 5.1.3, which was derived for beams with flexural reinforcement steel of 500MPa yield strength.

\[ V_{Rd,C,\text{min}} = \left[ 0.035k^{3/2}f_{ck}^{1/2} \right] d \]  

(5.1.3)

The derivation of this equation is covered by Walraven and Gmainer\cite{192}. This equation is presented for completeness only; it does not govern the resistance of any of the tests described in the database of tests (see Chapter 3).

### 5.1.2 SCS panels with shear reinforcement

The shear resistance model given in EN 1992-1-1 Section 6.2.3\cite{30} may be used for panels containing shear reinforcement. The design method is based on the plastic truss model developed in the 60s and 70s in Copenhagen\cite{128} and Zurich\cite{177}. For cases without axial forces, the resistance (Equation 5.1.6) is taken as the minimum of Equations 5.1.4 and 5.1.5:

\[ V_{Rd,s} = \frac{A_{sw} f_{w,tk}}{s} \gamma_s \cot \theta \]  

(5.1.4)

\[ V_{Rd,max} = b_w z v_1 \frac{f_{ck} / \gamma_c}{(\cot \theta + \tan \theta)} \]  

(5.1.5)

\[ V_{Rd} = \min (V_{Rd,max} : V_{Rd,s}) \]  

(5.1.6)

The truss angle \( \theta \) varies with the amount of shear reinforcement and effective concrete strength of the diagonal struts, as shown in Equation 5.1.7. Equation 5.1.7 is obtained by equating \( V_{Rd,s} \) to \( V_{Rd,max} \) and then solving for \( \theta \):

\[ \theta = \sin^{-1} \sqrt{\left( \frac{A_{sw} f_{w,tk} \gamma_c}{b_w z v_1 f_{ck} \gamma_s} \right)} \]  

(5.1.7)

With \( 1 \leq \cot \theta \leq 2.5 \) i.e. \( 45^\circ \geq \theta \geq 21.8^\circ \).
Where:

- $V_{Rd}$ is the design shear resistance of the member with shear reinforcement
- $V_{Rd,s}$ is the design value of the shear force sustained by the yielding shear reinforcement
- $V_{Rd,max}$ is the design value of the maximum shear force limited by crushing of the diagonal struts.
- $A_{sw}$ is the cross-sectional area of the shear reinforcement
- $s$ is the spacing of the shear reinforcement
- $z$ is the effective depth of the section, normally taken as 0.9 $d$
- $f_{ywk}$ is the characteristic yield strength of the shear reinforcement
- $\gamma_s$ is the partial factor for reinforcement (taken as 1.15 for persistent and transient design)
- $\nu_1$ is a strength reduction factor for cracked concrete in shear $= 0.6 \left( 1 - \frac{f_{ck}}{250} \right)$
- $f_{ck}$ is the characteristic concrete cylinder strength
- $\gamma_c$ is the partial factor for concrete (generally taken as 1.5 for persistent and transient design)

According to the plastic truss model, the resistance varies with the value adopted for $\theta$. In cases where $\theta$ given by Equation 5.1.7 is between 21.8° and 45°, both Equation 5.1.4 and Equation 5.1.5 are governing the design i.e. both equations are giving an equal value. In cases where Equation 5.1.7 gives values lower than 21.8° or larger than 45°, the truss angle should be taken as 21.8° or 45° respectively, with Equation 5.1.4 and Equation 5.1.5 governing the resistance respectively.

To demonstrate this concept, Figure 5.1 shows the curves produced for three identical SCS panels, with varying percentages of shear reinforcement. The panel is 300mm high and 500mm wide, and constructed with C30/35 concrete, 10mm thick plates, and shear reinforcement consisting of 12mm diameter bars spaced at 100mm with 500MPa yield strength steel. Where 0.9% reinforcement is provided, the optimum truss angle is around 30°. Where 0.45% reinforcement is provided, the optimum angle is around 20°; this is however outside the limit of 21.8°, so 21.8 is taken as the design value. For the heavily reinforced case, the angle is taken as 45°, as this is the upper bound.
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

Figure 5.1: Calculation of the concrete strut angle for varying percentages of reinforcement - 0.45% reinforcement, 0.9% reinforcement and 2.04% reinforcement.

The design domains can be split into four regions (Figure 5.2), where each of the resistance equations is governing.

Region 1 represents cases where the amount of shear reinforcement is sufficiently low that $V_{Rd} \leq V_{Rd,C}$. In this case the resistance is taken as $V_{Rd} = V_{Rd,C}$, given by Equation 5.1.1.

In region 2, the resistance is determined by Equation 5.1.4 and the ‘truss angle’ is limited to $21.8^\circ$. The linear cut-off in the plastic truss model was introduced in the code for simplicity, as the effective strength of the concrete ($v_1f_{ck}$ in Equation 5.1.5) is assumed constant. In reality, the effective strength reduces with lower values of $\theta$ due to higher transverse strains in the concrete. The linear cut-off was also introduced to indirectly control the width of shear cracks, for serviceability purposes[147].

In region 3, the resistance of the panel is set by both Equation 5.1.4 and Equation 5.1.5 giving equal values, which implies that both yielding of the reinforcement and crushing of the concrete occur simultaneously. It should be noted that the point of transition between regions 2 and 3 depends on the partial factors used.

In region 4, resistance is determined entirely by crushing of the inclined concrete.
struts, which given the large area of steel provided occurs before the shear reinforcement yields. As Figure 5.2 shows, adding additional reinforcement in region 4 does not increase the resistance of the structure, since the steel that has been provided is not being fully utilised at this point.

Figure 5.2: Graphical representation of regions of design, depending on ratio of concrete to steel strength

Figure 5.2 shows the cases from the database that were determined to fail by shear. It can be seen that the cases are entirely concentrated on the left of the plot, in region 1 and 2. The reinforcement ratio of the reinforced tests tends to be low, implying that the design resistance is almost entirely controlled by the steel reinforcement, with little interaction with the concrete. Cases with a higher reinforcement ratio are found in the database, but these cases are found to fail in bending before failing in shear.
5.1.3 Specific considerations for SCS panels affecting shear resistance

As previously discussed, the Eurocode compatible model for shear resistance of SCS panels is taken directly from Eurocode 2\[^{30}\], and was developed for conventional reinforced concrete construction. As such, a number of adjustments are required in order to apply the model to SCS panels.

Equation 5.1.1 includes the term $\rho_l$, which allows explicitly for the contribution of flexural reinforcement to the shear resistance actions. A key advantage of SCS panels over conventional RC members is that the steel plates may act as bending reinforcement, meaning it is possible to vastly reduce or completely remove internal bending reinforcement.

In most design methods for SCS panels, including the Bi-Steel manual\[^{24}\], an equivalent longitudinal reinforcement ratio of $\rho_l$ equal to $t_b/d$ is adopted, where $t_b$ is the thickness of the bottom plate and $d$ is the depth of the panel used in flexure. No attempt is made to account for the effects of the shear studs. This approach was adopted in this work for consistency with Eurocode 2.

Whilst simple, this approach is a potential source of uncertainty in the resistance predictions, since it relies on the assumption that the shear resisting actions (e.g. aggregate interlock, tensile stresses in the crack, dowel action and shear in the compression zone\[^{62}\]) remain mechanically similar between RC and SCS. The degree of shear connection is not accounted for in this approach. Comparisons between RC and SCS resistance mechanisms are explored further in Section 5.3.

The lever arm ($z$) in the plastic truss model for shear reinforced members, taken as $0.9d$ where $d$ is the distance from the top of the concrete to the centre of the bottom steel plate. This again relies on mechanical similarity between SCS and RC members.

Additional flexural reinforcement bars embedded in the concrete were present in 3 tests out of the 29 tests considered; 1 with and 2 without shear reinforcement. This additional reinforcement was found to have a negligible effect on $\rho_l$, so was ignored.

Limitations on modelling parameters, including $k \leq 2$, $\rho_l \leq 0.02$ and $\theta = 21.8^\circ - 45^\circ$ were adopted as recommended in EN1992-1-1, for RC beams. Since all of the tests had normal concrete strengths ($f_{ck} < 50\,\text{MPa}$), further limits on $f_{ck}$ in the shear calculations were not required.
5.2 Analysis of experimental data

The suitability of the model described in Sections 5.1.1 and 5.1.2 for the determination of the resistance of SCS panels to out-of-plane shear forces is established through comparison with a range of tests. As first described in Chapter 3, a large database of tests has been gathered from the literature, covering a large range of potential designs. However, the majority of these cases fail via bending failure modes, and must therefore be excluded.

The tests that remain still cover a range of designs. The majority of tests are simply supported 3 or 4 point bending tests, though a smaller number are continuous beam tests (statically determinate with point loads) with reversing bending moment diagrams. Concrete strength is typically between 30MPa and 40MPa. The steel yield strength for the plates and shear reinforcement varies between 270MPa and 640MPa as is standard in industry. The thickness of the steel plates range between 4mm and 19mm, with flexural reinforcement ratios ranging from 1.2% to 3%. A smaller number of tests have high reinforcement ratios, at around 4 or 5%). Member height ranges between 150mm and 914mm.

Most of the tests were conducted as part of mixed programs that also investigated out-of-plane bending and other type of failures such as slip failure (interfacial-shear failure). In this chapter, only tests failing in shear and bending/shear were considered. The moment for the cross-section ($M_{Rd}$) was calculated for each test, in accordance with the final rules developed in Chapter 4, as presented in Section 4.12. Cases in which the experimental applied moment at failure exceeded 10% above $M_{Rd}$ were excluded from subsequent analysis. This limit is consistent with other works considering databases of shear resistance, including Reineck et al.\textsuperscript{[148]} and Collins et al.\textsuperscript{[49]}.

After the considerations above, a total of 29 beam tests (16 without and 13 with shear reinforcement) remain in the set. These are shown in Table 5.5.

Utilisations in both shear and bending are presented. The bending unity factor ($U_{bending}$) is calculated as follows:

\[
U_{bending} = \frac{M_{Ed}}{M_{Rd}} \quad (5.2.1)
\]

Where:

$M_{Ed}$ is the applied bending moment
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

\[ M_{Rd} \] is the moment resistance, calculated in accordance with the rules presented in 4.12

The shear unity factor \( U_{\text{shear}} \) is calculated as follows:

\[ U_{\text{shear}} = \frac{V_{Ed}}{V_{Rd}} \quad (5.2.2) \]

Where:

\[ V_{Ed} \] is the applied shear force
\[ V_{Rd} \] is the moment resistance, calculated using Equation 5.1.1 or Equation 5.1.6

The failure load is set so that either \( U_{\text{bending}} \) or \( U_{\text{shear}} \) is equal to 1. This check is then the critical check for that design.

<table>
<thead>
<tr>
<th>Test</th>
<th>Table No.</th>
<th>( h )</th>
<th>( a/d )</th>
<th>( \rho_{\text{flexural}} )</th>
<th>( \rho_{\text{shear}} )</th>
<th>( \gamma )</th>
<th>( d_{p,\text{min}} )</th>
<th>( U_{\text{bending}} )</th>
<th>( U_{\text{shear}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>E2</td>
<td>3.2</td>
<td>150</td>
<td>3.67</td>
<td>4.00</td>
<td>-</td>
<td>148.1</td>
<td>6.0</td>
<td>0.51</td>
<td>1.00</td>
</tr>
<tr>
<td>SP1-1</td>
<td>3.8</td>
<td>457</td>
<td>3.18</td>
<td>1.40</td>
<td>-</td>
<td>112.6</td>
<td>12.7</td>
<td>0.58</td>
<td>1.00</td>
</tr>
<tr>
<td>SP1-3</td>
<td>3.8</td>
<td>457</td>
<td>3.18</td>
<td>2.08</td>
<td>-</td>
<td>75.9</td>
<td>12.7</td>
<td>0.58</td>
<td>1.00</td>
</tr>
<tr>
<td>SP1-4</td>
<td>3.8</td>
<td>457</td>
<td>2.48</td>
<td>1.40</td>
<td>-</td>
<td>78.8</td>
<td>12.7</td>
<td>0.58</td>
<td>1.00</td>
</tr>
<tr>
<td>3</td>
<td>3.5</td>
<td>450</td>
<td>1.00</td>
<td>1.00</td>
<td>-</td>
<td>9.9</td>
<td>9.0</td>
<td>0.91</td>
<td>1.00</td>
</tr>
<tr>
<td>4</td>
<td>3.5</td>
<td>450</td>
<td>1.00</td>
<td>1.00</td>
<td>-</td>
<td>54.2</td>
<td>9.0</td>
<td>0.24</td>
<td>1.00</td>
</tr>
<tr>
<td>5</td>
<td>3.5</td>
<td>450</td>
<td>2.00</td>
<td>1.00</td>
<td>-</td>
<td>39.5</td>
<td>9.0</td>
<td>0.71</td>
<td>1.00</td>
</tr>
<tr>
<td>8</td>
<td>3.5</td>
<td>600</td>
<td>2.00</td>
<td>2.00</td>
<td>-</td>
<td>14.1</td>
<td>16.0</td>
<td>1.00</td>
<td>0.83</td>
</tr>
<tr>
<td>9</td>
<td>3.5</td>
<td>600</td>
<td>2.00</td>
<td>2.00</td>
<td>0.32</td>
<td>34.1</td>
<td>16.0</td>
<td>0.82</td>
<td>1.00</td>
</tr>
<tr>
<td>S1</td>
<td>3.5</td>
<td>500</td>
<td>1.00</td>
<td>1.80</td>
<td>-</td>
<td>25.6</td>
<td>16.0</td>
<td>0.63</td>
<td>1.00</td>
</tr>
<tr>
<td>S2</td>
<td>3.5</td>
<td>500</td>
<td>2.00</td>
<td>1.80</td>
<td>-</td>
<td>41.1</td>
<td>16.0</td>
<td>0.77</td>
<td>1.00</td>
</tr>
<tr>
<td>S3</td>
<td>3.5</td>
<td>500</td>
<td>1.00</td>
<td>1.80</td>
<td>0.05*</td>
<td>33.2</td>
<td>16.0</td>
<td>0.49</td>
<td>1.00</td>
</tr>
<tr>
<td>S4</td>
<td>3.5</td>
<td>500</td>
<td>2.00</td>
<td>1.80</td>
<td>0.05*</td>
<td>47.3</td>
<td>16.0</td>
<td>0.72</td>
<td>1.00</td>
</tr>
<tr>
<td>S5</td>
<td>3.5</td>
<td>500</td>
<td>1.00</td>
<td>1.80</td>
<td>0.21</td>
<td>34.2</td>
<td>16.0</td>
<td>0.65</td>
<td>1.00</td>
</tr>
<tr>
<td>S6</td>
<td>3.5</td>
<td>500</td>
<td>2.00</td>
<td>1.80</td>
<td>0.21</td>
<td>44.6</td>
<td>16.0</td>
<td>0.96</td>
<td>1.00</td>
</tr>
</tbody>
</table>

154
### Table 5.5: Tests included in study of shear resistance

<table>
<thead>
<tr>
<th>Test</th>
<th>Table No.</th>
<th>$h$ (mm)</th>
<th>$d/h$</th>
<th>$\rho_{b,\text{mean}}$ (%)</th>
<th>$\rho_{\text{shear}}$ (%)</th>
<th>$\gamma$</th>
<th>$d_{b,\text{eff}}$ (mm)</th>
<th>$U_{b,\text{ref},\text{R}}$</th>
<th>$U_{b,\text{shear}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>NR-0R-3S400-4ST</td>
<td>3.7</td>
<td>500</td>
<td>3.20</td>
<td>1.80</td>
<td>0.19</td>
<td>70.3</td>
<td>16.0</td>
<td>0.60</td>
<td>1.00</td>
</tr>
<tr>
<td>NRC-0R-4S400-4ST</td>
<td>3.7</td>
<td>500</td>
<td>3.20</td>
<td>1.80</td>
<td>0.25</td>
<td>76.4</td>
<td>16.0</td>
<td>0.74</td>
<td>1.00</td>
</tr>
<tr>
<td>WS'1.5</td>
<td>3.9</td>
<td>734</td>
<td>1.46</td>
<td>1.91</td>
<td>0.46</td>
<td>90.5</td>
<td>19.0</td>
<td>0.76</td>
<td>1.00</td>
</tr>
<tr>
<td>SP3_E2</td>
<td>3.10</td>
<td>800</td>
<td>3.75</td>
<td>1.88</td>
<td>0.11</td>
<td>180.9</td>
<td>19.0</td>
<td>0.74</td>
<td>1.00</td>
</tr>
<tr>
<td>SP4_E2</td>
<td>3.10</td>
<td>800</td>
<td>3.75</td>
<td>1.88</td>
<td>0.08</td>
<td>176.7</td>
<td>19.0</td>
<td>0.87</td>
<td>1.00</td>
</tr>
<tr>
<td>SP6_E2</td>
<td>3.10</td>
<td>800</td>
<td>3.75</td>
<td>1.50</td>
<td>0.11</td>
<td>219.6</td>
<td>19.0</td>
<td>0.85</td>
<td>1.00</td>
</tr>
<tr>
<td>JZ2.5-1</td>
<td>3.12</td>
<td>300</td>
<td>2.50</td>
<td>2.00</td>
<td>0.24</td>
<td>180.4</td>
<td>13.0</td>
<td>0.55</td>
<td>1.00</td>
</tr>
<tr>
<td>JZ3.0-1</td>
<td>3.12</td>
<td>300</td>
<td>3.00</td>
<td>2.00</td>
<td>0.20*</td>
<td>153.8</td>
<td>13.0</td>
<td>0.60</td>
<td>1.00</td>
</tr>
<tr>
<td>JZ3.0-N</td>
<td>3.12</td>
<td>300</td>
<td>3.00</td>
<td>2.00</td>
<td>-</td>
<td>165.5</td>
<td>8.0</td>
<td>0.52</td>
<td>1.00</td>
</tr>
<tr>
<td>JZ2.5-2</td>
<td>3.12</td>
<td>380</td>
<td>2.50</td>
<td>2.26</td>
<td>0.12*</td>
<td>132.2</td>
<td>8.0</td>
<td>0.34</td>
<td>1.00</td>
</tr>
<tr>
<td>JZ3.0-2</td>
<td>3.12</td>
<td>380</td>
<td>3.00</td>
<td>2.26</td>
<td>0.12*</td>
<td>160.9</td>
<td>8.0</td>
<td>0.40</td>
<td>1.00</td>
</tr>
<tr>
<td>JZ3.0-3</td>
<td>3.12</td>
<td>380</td>
<td>3.00</td>
<td>2.26</td>
<td>0.23</td>
<td>172.9</td>
<td>8.0</td>
<td>0.49</td>
<td>1.00</td>
</tr>
<tr>
<td>JZ3.0-4</td>
<td>3.12</td>
<td>380</td>
<td>3.00</td>
<td>2.26</td>
<td>0.17</td>
<td>171.0</td>
<td>8.0</td>
<td>0.52</td>
<td>1.00</td>
</tr>
<tr>
<td>JZ3.5-2</td>
<td>3.12</td>
<td>380</td>
<td>3.50</td>
<td>2.26</td>
<td>0.23</td>
<td>200.2</td>
<td>8.0</td>
<td>0.57</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Note: Cases marked with an asterisk (*) contain a small amount of shear reinforcement, but are included in the unreinforced group, due to a greater resistance being given by Equation 5.1.1 than Equation 5.1.4.

### 5.2.1 Comparisons between the resistance formulation and the test results

For the purposes of comparison, the tests results are split into two populations; reinforced and unreinforced in shear. In a small number of cases the level of reinforcement included in the test was low enough that the model for an unreinforced section provides a higher resistance than the reinforced model (Design region 1, as discussed in Section 5.1.2). In these cases, as marked in Table 5.5 with an asterisk, the results are included in the unreinforced population. Inclusion of these results in the unreinforced population is somewhat contentious; it is possible that the strength of these samples is slightly influenced by the presence of shear reinforcement (perhaps by initiating cracks), although this effect is generally neglected in practice.

Figure 5.3 shows the ratio between the test and model resistances for SCS panels without shear reinforcement for different shear span to-effective depth ratios.
Figure 5.3: Ratio of test to model resistances for different shear span to effective depth ratios for members without shear reinforcement

Figure 5.3 shows a varying level of precision across various values of $a/d$. For lower values of $a/d$ the model produces an extremely conservative result. For higher values of $a/d$ the precision is much better, with the majority of resistances close to the model predictions.

Further examination of the test results suggested that the conservative results were obtained due to lack of provision for arching action. This is discussed further in Section 5.3.2.

Figure 5.4 shows the ratio between the test and model resistances for SCS panels with shear reinforcement.
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

Figure 5.4: Ratio of test to model resistances for different shear span to effective depth ratios for members with shear reinforcement

The model predictions in this case show better consistency with the test results across a range of $a/d$ ratios. Although fewer data points are available in the $a/d \geq 2$ range, it can be seen that the extremely conservative predictions found in Figure 5.3 are not found for the reinforced beams. These results, including the scatter, are consistent with RC beams\cite{154}.

5.2.2 Comparison of strength predictions between SCS panels and RC beams

The precision of the shear strength predictions obtained for SCS panels with shear reinforcement is similar to the precision of prediction of resistance of RC beams, although the amount of experimental data of SCS panels is limited.

For members without shear reinforcement, Equation 5.1.1 appears to give better predictions for RC beams for which it was originally calibrated for. Comparisons for the two data sets are shown side-by-side in Figure 5.5. Whilst safe, Equation 5.1.1 gives a significant scatter in the accuracy of the shear strength predictions of SCS panels. Data for conventionally reinforced beams was obtained from Regan\cite{147}.

The reasons for the large scatter in the SCS panels are related to the differences in the mechanical behaviour of these panels described in Section 5.1.3. Such considerations are not realistically captured in the empirical formula for shear and more advanced
shear models would need to be developed to reduce this scatter, considering the different shear resistance mechanisms mobilised in this case, as discussed in Section 5.3.

![Figure 5.5: Comparison between tests on conventional reinforced concrete beams (left) and SCS panels (right), both without shear reinforcement. Data for conventionally reinforced beams was obtained from Regan[147].](image)

### 5.3 Shear transfer actions in members without shear reinforcement

As previously shown in Section 5.2, the shear resistance models for conventional reinforced concrete construction can be applied to SCS panels, with reasonable precision expected. However, it can also be seen that the database of tests used to make this comparison is small, once filtering for alternative failure modes is complete. The data available is insufficient to draw any definitive conclusion about the effects of shear connection on the accuracy of the resistance formulation.

In order to overcome this limitation, an investigation based on finite element modelling is proposed. A large range of models, covering cases with a range of different levels of shear connection are designed and computed, with the intention of observing any changes in resistance.

Designing an appropriate parametric study requires a detailed understanding of the mechanical behaviour of beams in shear, since this allows cases to be targeted toward designs where changes in behaviour are most obvious. This section explores this behaviour, and discusses the changes that might arise as a result of reduced degrees of
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

shear connection.

Research on the shear resistance of conventional RC beams has attracted considerable research attention in recent years\textsuperscript{[62,153,188]}. As a result, a number of resistance mechanisms that contribute to the overall resistance of the beams have been classified and modelled.

Evidence from tests of SCS panels without shear reinforcement, as described in Section 5.2, show that the critical shear crack develops at failure from a flexural crack in a similar way to a RC member\textsuperscript{[152]}.

Fernández Ruiz et al.\textsuperscript{[62]}, among others, characterise a number of shear resisting actions:

- Cantilever action - Cantilever action occurs when wedges or ‘teeth’ that form between cracks act as a cantilevered beam between the tension and compression stress blocks. No inconsistency is expected between RC and SCS with regard to this mechanism, though some changes in crack patterns may occur.

- Aggregate interlock - Contact friction between surfaces post-cracking is capable of carrying considerable stress, both normally and tangentially. This mechanism is expected to occur in the same manner in SCS panels as in RC construction.

- Dowel action - Steel reinforcement spanning across cracks can act as a dowel i.e. in shear. SCS panels tend not to include embedded longitudinal reinforcement. The plates themselves cannot act as dowels, as the location of the plate at the bottom of the beam does not allow it to span a crack. Conversely, the presence of embedded shear connectors may provide some dowel action which would not be present in RC beams.

- Residual tensile strength of concrete - Shear may be carried in uncracked sections of the beam i.e. areas of low moment. Again this mechanism is expected to remain consistent between SCS and RC construction.

- Arching action - Leads to significant enhancements in resistance. See section 5.3.2.

The contributions and interactions between the mechanisms change depending on distance over which the shear is mobilised, compared to the beam size. In accordance with the behaviour of RC members, the shear strength of SCS panels is highly influenced by the slenderness of the panel. Slenderness in this context is defined by the shear span $a$ to effective depth $d$ ratio due to differences in crack development within the span. For simply supported tests, with point-loads, the shear span ($a$) is
the distance measured between the load and the nearest support. For continuous spans, the shear span is measured from the loading point to the point of contra-flexure or zero moment.

In many design models, including the ones presented in 5.7.1 and 5.7.2, the $a/d$ is approximated using the relationship given in Equation 5.3.1. The $a/d$ ratio is presented as $\lambda$.

$$\lambda = \frac{M_u}{V_u d}$$

(5.3.1)

Where:

- $M_u$ is the maximum moment in the beam
- $V_u$ is the shear in the beam at the location of the maximum moments
- $d$ is the effective depth of the section

The influence of flexural cracks on shear strength is less relevant for shear reinforced members compared to unreinforced members (and beams in design region 1) in which failure is governed by the formation of a localized crack. At a certain level of $a/d$, which depends on a number of design parameters, the development of flexural cracks which can penetrate into the theoretical strut leading to a reduction in shear strength, reducing it’s effectiveness, and leading to an overall drop in resistance. This effect is generally described using ‘Kani’s valley’[^92]. An illustrative graph of the concept can be seen in Figure 5.6.

![Illustration of Kani’s valley](image)

*Figure 5.6: Illustration of Kani’s valley, taken from the original publication by Kani[^92]*

Kani’s valley occurs due to the development of flexural cracks which are closer to the section under the loading point. This effect is of interest, since the development of cracks in this region is found to be influenced by strain concentrations influenced by the degree of bond between the tension reinforcement and the concrete. It can therefore be reasonably postulated that a change in degree of shear connection, which is analogous in this case to a change in bond stiffness, might produce a change in resistance. Evidence for this conclusion is found in the test database, as discussed by Sagaseta and Francis. It is also notable that this change can have a positive or negative effect on the resistance, depending on the slenderness of the specimen i.e. the position in the valley. Such an effect might be considered counter-intuitive, since it would normally be expected that an increase in degree of shear connection should produce a higher resistance.

An interesting example which illustrates one extreme was reported by Muttoni and Ruiz in RC tests from Leonhardt and Walther (a/d = 2.77) with critically low bond-slip conditions using smooth bars; the failure load was 72% higher compared to an identical beam with deformed bars. Although this example does not represent bond-slip conditions of SCS panels, it does give an idea of the potential implications of different bond-slip conditions affecting the shear strength.

5.3.1 Dowel action

Testing by Varma et al., as reproduced in Figure 5.7, shows that for some configurations of SCS panels, a delamination crack occurs on a plane above the top of the shear connectors. This localised crack, which combines de-bonding of the steel plate from the concrete and a pull-out of the studs at the bottom, reduces the contribution of dowel action toward shear strength. Recent work by Cavagnis et al. for conventional reinforced concrete beams suggest that the contribution of dowel action to shear strength is not significant, which means the potential reduction in dowel action shown in Figure 5.7 cannot be expected to significantly affect the accuracy of predictions for SCS panels when using models for reinforced concrete in design.
5.3.2 Arching action

One of the key conclusions of the initial comparison exercise shown in Figure 5.3 is that panels with low $a/d$ ratios tend to show much larger resistances in testing than the resistances predicted by the design model.

This effect is well known in RC beams. The enhancement can be attributed to ‘arching action’.

The compressive strength of concrete tends to be much higher than the tensile strength. As a result, cracking of the tensile face can occur at low proportions of the failure load, as discussed in Section 4.8.1. This cracking causes an upward migration of the neutral axis level, which is further accompanied by an outward expansion of the member at the supports. If the tendency to expand is restrained (in this case by the combination of the studs and the tension plate acting as tensile reinforcement), compressive forces can be developed, leading to direct transfer into the support.

There are a large number of parameters that affect the ability of arching action to be developed. Restraint of outwards expansion is critical, and can be difficult to quantify. The enhancements available through arching action can be significant (as shown in Figure 5.3), though small changes in the design or detailing may stop these enhancements being realised in practice.

Arching action is the main mechanism contributing to the resistance members on the left of Kani’s valley. The drop in resistance in the valley itself can be attributed to the disruption of effectiveness of the arching action by the presence of flexural cracks\(^{[92]}\).

A single analytical formula that is capable of calculating the strength enhancements available through arching action, that covers both slender and non-slender beams, has
proven to be illusive\textsuperscript{[62]}. Design methods from around the world adopt a number of different approaches, with varying levels of precision (as discussed further in Section 5.7).

Research on arching action has progressed in recent years. It has been established that arching action can be modelled through the use of the Strut & Tie method\textsuperscript{[152]}. To apply this method, load paths are established between application points and supports. Based on the inclination of the strut and the effective stiffness of the support, a maximum permissible compressive stress can be obtained, which can be used to establish the shear resistance. Figure 5.8 shows how the Strut & Tie method might be applied to an unreinforced SCS panel.

![Figure 5.8: Application of the Strut & Tie method to obtain the shear resistance of an SCS panel (Taken from Sagaseta and Francis\textsuperscript{[152]})](image)

Uncertainty remains in the understanding of the effect of shear connectors on the effectiveness of the strut. The method has been successfully applied to a number of the tests in the test database, as described by Sagaseta and Francis\textsuperscript{[152]}. Despite its success in modelling the tests in the database, the model does not explicitly cover the degree of shear connection.

EN 1992-1-1\textsuperscript{[30]} includes provision for arching action, but the approach is extremely simplified. Rather than attempt to quantify the arching contribution, a load reduction factor is applied to loads with an $a/d$ ratio of 2 or less. For loads with an $a/d$ ratio less than 1, a 50\% reduction is applied. A linearly proportional reduction factor ($\beta$) is applied between these two points. This is shown in Figure 5.9.
Figure 5.9: Reduction factor for shear force produced by loads applied close to supports

The same arching model is applied to cases where shear reinforcement is included. However, an additional limitation on the force that can be carried is included in EN 1992-1-1[^30] Equation 6.19, as reproduced below in Equation 5.3.2:

$$V_{Ed} \leq A_{sw}f_{ywd}$$  \hspace{1cm} (5.3.2)

Where:

$A_{sw}f_{ywd}$ is the resistance of the shear reinforcement crossing the inclined shear crack between the loaded areas, as illustrated in Figure 5.10.
For cases with a single point load, such as those included in the parametric study, Equation 5.3.2 limits the magnitude of the load applied to the total force that can be carried in the shear reinforcement in the middle 75% of the shear span. For cases like those presented in Section 5.6.1, the resistance limit can be substantially less than the resistance given by Equation 5.1.6.

5.4 Investigation of the effects of shear connection on out-of-plane shear resistance using FE

The data required for obtaining a full understanding of the effect of degree of shear connection on shear resistance is insufficient, as first discussed in Section 5.2. In order to overcome this, a parametric investigation using finite element modelling is proposed.

As shown in Chapter 4, finite element modelling is capable of producing an accurate approximation of test results governed by flexure.

The details of the model are discussed in Section 4.4. Verification of the model for tests failing in bending is described in Section 4.7. The same modelling approach is adopted for this Chapter, with no shear specific changes.

Section 5.4.1 shows a comparison between the FE predictions and the test results available. Section 5.4.2 discusses the design of the cases that are included in the study. Sections 5.5 and 5.6 then show the results of the study for panels unreinforced and reinforced in shear, respectively.
5.4.1 Verification of the FE model

The reliability of the FE parametric study depends on the ability of the FE model to reproduce the behaviour of the panels in shear accurately. The accuracy of the model is tested against a number of tests from the test database.

It is important to verify the behaviour across a range of parameters. Panels with high percentages of shear reinforcement will tend to fail in a relatively ductile manner, while panels with little or no shear reinforcement will tend to fail in a much more brittle manner. The more brittle behaviours are known to be harder to reproduce in an FE analysis\cite{55}.

The test program by Leng and Song\cite{105} provides the best data to be used in the verification, since it includes tests on both reinforced and unreinforced panels. The material properties, dimensional parameters, load-deflection curves and crack patterns are also well recorded.

Figure 5.11 shows a comparison between the model and the results from testing for case JZ3.0-N. It can be seen that the prediction is slightly stiffer than the recorded result in this case. This case contains no shear reinforcement. As is characteristic of panels of this type, the failure observed in the test is sudden and brittle. Unfortunately the crack pattern is not available for this case, but it is likely that the prediction of the FE model would strongly correlate. The FE model shows the typical diagonal shear crack patterns shown in other tests, like Figure 5.13.
Figure 5.11: Comparison between recorded load-deflection curve and a curve produced by FE analysis for case JZ3.0-N by Leng and Song \cite{105}

Figure 5.12 shows the same level of accuracy for case JZ3.0-2. The presence of a small number of bars affects the post-failure behaviour, which shows a less abrupt drop in load carrying capacity. This affect is captured by the FE model.
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

Test Result
FE prediction
Predicted resistance using Equation 5.1.1

Figure 5.12: Comparison between recorded load-deflection curve and a curve produced by FE analysis for case JZ3.0-2 by Leng and Song\[105\]

Figures 5.13 and 5.14 compare the crack patterns predicted by the FE and the crack pattern observed in testing.

Figure 5.13: Photograph of cracks for case JZ3.0-2, taken from paper by Leng and Song\[105\]
It should be noted that the model also predicts the non-critical cracking that can be observed in Figure 5.13, though they are hidden in the contours of the plot. These cracks can be seen on earlier plots, as shown in Figure 5.15.

The predictions are also good for the other test programs. The prediction for case SP4 by Koukkari and Fülöp\cite{98} is shown in Figure 5.16. Again, strong correlation between the FE model predictions of the overall failure load and load-deflection response compared to the test results is observed.
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

Figure 5.16: Comparison between recorded load-deflection curve and a curve produced by FE analysis for case SP4 by Koukkari and Fülöp.[98]

Figure 5.17 shows another load-deflection curve, from the series by Varma et al.[184]. This test is similar to the JZ3.0-N, as shown in Figure 5.11, with the lack of reinforcement producing a brittle failure. Again, the result is close to result recorded in the test.
The parametric nature of the FE model (as described in Section 4.4) means that it was possible to model all of the cases presented in Table 5.5. In all cases, failure occurs within 10% of the failure load recorded in the tests, which supports the approach adopted.

Figure 5.18 shows a plot of all of the cases included in the calibration study.
Figure 5.18: Ratio of test to model resistances for different shear span to effective depth ratios for members without shear reinforcement

Figure 5.18 shows that good consistency is seen for a range of \( a/d \) ratios. The majority of the results are predicted within 10% of the resistance observed in testing. This level of accuracy in the strength predictions is consistent with results from other researchers of shear in reinforced concrete structures, and within the range of acceptability\[^{55}\].

5.4.2 Case design

As with any computational parametric study, availability of computer resource and processing times means that the number of cases that can be included is limited. As such, it is important that the study is designed in such a way as to produce results that highlight the key behaviours, in as few cases as possible.

As discussed in detail in Section 5.2, the results of testing of SCS panels tend to show that the behaviour is similar to RC beams of equivalent size. The key characteristic parameter for concrete beams is the \( a/d \) ratio, which defines the tendency of the various resistance mechanisms (arching, dowel action etc.) to contribute to the overall resistance. For this reason, the study splits the cases into three groups, with three values of \( a/d \). Unreinforced and reinforced panels are considered separately, leaving six separate groups. Within each group, the key parameter for investigation is the degree of shear connection.
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

Once the key parameters for the study are identified, it becomes desirable to vary the remaining parameters as little as possible, to eliminate potential sources of uncertainty when changes in behaviour occur.

In all cases a 1m width beam is used. The span is set at 8m, with 200mm additional overhang allowed at either end. No evidence of anchorage failure was observed, despite the low number of studs in this area, which confirms that the anchorage is sufficient. Shear connectors are placed at 200mm centres in the longitudinal direction, with 3 connectors per row. For the cases where tie-bars are included, the centre row is replaced.

Figure 5.19 shows an indicative diagram of the layout used in the shear parametric study.

![Indicative diagram of the beam design used for all cases in the shear parametric study](image)

**Figure 5.19:** Indicative diagram of the beam design used for all cases in the shear parametric study

Figure 5.19 shows the stud and tie-bar layout in the transverse direction.

![Indicative diagram of position of shear stud and tie-bars used for all cases in the shear parametric study](image)

**Figure 5.20:** Indicative diagram of position of shear stud and tie-bars used for all cases in the shear parametric study

Sections 5.5 and 5.6 include tables that show a summary of the remaining key design parameters.

It should be noted that the parameters presented in the tables below (e.g. Table 5.8) are not independent. For example, the percentage of flexural reinforcement $\rho_{\text{flexural}}$ is determined by both the thickness of the tension plate and the height of the section. The
shear connection percentage $\mu$ is linked to the diameter of the studs, the thickness of the plate and the shear span length. If the plate thickness is decreased, perhaps with the aim of increasing the percentage of flexural reinforcement, this will also increase the degree of shear connection, which may not be the aim. Changing the height of the section results in more changes to the design, since the height is proportional to the shear span, through the $a/d$ ratio. This means the degree of shear connection is then sensitive to the height.

The cases eventually presented below are a selection of the groups studied. Further studies, not presented in this thesis for clarity, showed that the changes in behaviour could be reasonably classified by degree of shear connection. The results presented in Section 5.5.4 suggest that the difference between the performance of panels with dissimilar stud spacings but similar degrees of shear connection tend to show the same behaviour.

It should be noted that many of these designs are not necessarily achievable in real construction conditions. In particular, shear studs tend to be standardised, with only 16 or 19mm diameters with a height of 95mm readily available. However, similar levels of shear connection can be reproduced by using standard stud sizes at either closer or wider stud spacings.

### 5.5 FE study - Panels without shear reinforcement

This section describes the results of the FE study on shear resistance of unreinforced SCS panels, with varying degrees of shear connection. The results are split into three groups, each with a different $a/d$ ratios.

#### 5.5.1 Group 1 - $a/d = 1$

Group 1 has loads placed very close to the support, giving an $a/d$ ratio of 1. The close proximity of the load means that the primary shear transfer mechanism is expected to be arching action.

Table 5.8 shows the cases included in this group.
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

Table 5.8: Summary of cases included in Group 1

<table>
<thead>
<tr>
<th>Test</th>
<th>$h$ (mm)</th>
<th>$a/d$</th>
<th>$\rho_{f/leural}$ (%)</th>
<th>$\rho_{shear}$ (%)</th>
<th>$\gamma$</th>
<th>$d_{stud}$ (mm)</th>
<th>$U_{bending}$</th>
<th>$U_{shear}$</th>
<th>$R_{Test} / R_{Design}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>800</td>
<td>1.00</td>
<td>1.00</td>
<td>-</td>
<td>45</td>
<td>19.0</td>
<td>0.58</td>
<td>1.00</td>
<td>2.22</td>
</tr>
<tr>
<td>2</td>
<td>800</td>
<td>1.00</td>
<td>1.00</td>
<td>-</td>
<td>55</td>
<td>21.0</td>
<td>0.47</td>
<td>1.00</td>
<td>2.51</td>
</tr>
<tr>
<td>3</td>
<td>800</td>
<td>1.00</td>
<td>1.00</td>
<td>-</td>
<td>66</td>
<td>23.0</td>
<td>0.39</td>
<td>1.00</td>
<td>2.83</td>
</tr>
<tr>
<td>4</td>
<td>800</td>
<td>1.00</td>
<td>1.00</td>
<td>-</td>
<td>78</td>
<td>25.0</td>
<td>0.33</td>
<td>1.00</td>
<td>3.00</td>
</tr>
<tr>
<td>5</td>
<td>800</td>
<td>1.00</td>
<td>1.00</td>
<td>-</td>
<td>91</td>
<td>27.0</td>
<td>0.29</td>
<td>1.00</td>
<td>2.95</td>
</tr>
<tr>
<td>6</td>
<td>800</td>
<td>1.00</td>
<td>1.00</td>
<td>-</td>
<td>105</td>
<td>29.0</td>
<td>0.26</td>
<td>1.00</td>
<td>3.16</td>
</tr>
</tbody>
</table>

The parameters of the tests have been chosen to be as close as possible to the parameters that would be selected in real projects. In this case a plate thickness of 8mm is used, so that degrees of shear connection above 100% could be created without extremely large diameter studs. The subsequent degree of shear connection of 1% is low compared to those values from the database (see Table 5.5), though it is within reasonable bounds\(^4\). Figure 5.21 shows the results of the FE modelling.
The results for $\mu \geq 66\%$ show that the resistance predicted by the FE model is considerably higher than the resistance predicted by Equation 5.1.1, even when the arching adjustment factor $\beta$ is applied. This result is consistent with the results of testing, as shown in Figure 5.3.

Plots obtained from the FE show the formation of an effective strut, as discussed in Section 5.3.2. This is highlighted by the Von-Mises stress plot shown in Figure 5.22. The presence of shear connectors appears to provide sufficient confinement at the base of the strut, which is essential for restraining the outward expansion that develops the arching effect, as discussed in Section 5.3.2. Failure occurs when the compressive strength of the strut is exceeded. These can be seen in the strain plot shown in Figure 5.23.
The results show sensitivity to the level of shear connection. Test 1 shows a considerable drop in resistance from cases with full shear connection, though the prediction by the Eurocode model is still on the safe side. Given it is unlikely that panels with degrees as low as 45% will be utilised in real designs, detailed investigation of this effect is not required.

5.5.2 **Group 2 - \(a/d = 3\)**

Group 2 has loads placed to give an \(a/d\) ratio of 3. At these distances the influence of arching action is greatly diminished. Comparisons between the existing model and tests suggest that this is the area where the Eurocode model is least conservative. Table 5.9 shows the cases included in this group. A 12mm plate is used in this case, to allow the lower degrees of shear connection to be achieved.

<table>
<thead>
<tr>
<th>Test</th>
<th>(\zeta)</th>
<th>(a/d)</th>
<th>(\rho_{flexural})</th>
<th>(\rho_{shear})</th>
<th>(\gamma)</th>
<th>(d_{final})</th>
<th>(U_{bending})</th>
<th>(U_{shear})</th>
<th>(R_{Test}/R_{Design})</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>800</td>
<td>3.00</td>
<td>1.50</td>
<td>-</td>
<td>42</td>
<td>13.0</td>
<td>1.00</td>
<td>0.71</td>
<td>1.10</td>
</tr>
<tr>
<td>2</td>
<td>800</td>
<td>3.00</td>
<td>1.50</td>
<td>-</td>
<td>56</td>
<td>15.0</td>
<td>1.00</td>
<td>0.95</td>
<td>0.97</td>
</tr>
<tr>
<td>3</td>
<td>800</td>
<td>3.00</td>
<td>1.50</td>
<td>-</td>
<td>72</td>
<td>17.0</td>
<td>0.82</td>
<td>1.00</td>
<td>0.95</td>
</tr>
<tr>
<td>4</td>
<td>800</td>
<td>3.00</td>
<td>1.50</td>
<td>-</td>
<td>90</td>
<td>19.0</td>
<td>0.66</td>
<td>1.00</td>
<td>1.06</td>
</tr>
<tr>
<td>5</td>
<td>800</td>
<td>3.00</td>
<td>1.50</td>
<td>-</td>
<td>110</td>
<td>21.0</td>
<td>0.59</td>
<td>1.00</td>
<td>1.21</td>
</tr>
<tr>
<td>6</td>
<td>800</td>
<td>3.00</td>
<td>1.50</td>
<td>-</td>
<td>132</td>
<td>23.0</td>
<td>0.59</td>
<td>1.00</td>
<td>1.20</td>
</tr>
<tr>
<td>7</td>
<td>800</td>
<td>3.00</td>
<td>1.50</td>
<td>-</td>
<td>156</td>
<td>25.0</td>
<td>0.59</td>
<td>1.00</td>
<td>1.19</td>
</tr>
</tbody>
</table>
Figure 5.24 shows the results of the FE modelling for the first two cases included in the study, the first of which is predicted to fail by bending induced interfacial shear.

The FE results show the accuracy of the bending model prediction when the degree of shear connection is low. It can be seen that the character of the load-deformation curves for bending is different to the character of the load-deformation curves for shear. Shear failure tends to show sudden drops in resistance, while the bending failures show a smooth increase in deflection as load increases.

![Figure 5.24: Load-deflection curves for Table 5.9 cases 1 & 2](image)

Figure 5.25 shows the load-deflection curves for the tests that are predicted to fail in shear.
The results show sensitivity to the level of shear connection. For the lower degrees of shear connection the Equation 5.1.1 prediction is unconservative. As first explored in Section 5.3, increased flexibility of the shear connection allows the flexural and flexure/shear cracks to open more easily. It can now be clearly seen that this flexibility detrimentally effects resistance.

Equation 5.1.2 assumes full shear connection when calculating the resistance. The following amendment to Equation 5.1.2 is suggested to account for cases with reduced shear connection, based on the results presented in Figure 5.25.

\[
\rho_l = \min \left\{1 : 0.8\mu_b \right\} \rho_{l,full} = \min \left\{1 : 0.8\mu_b \right\} \frac{t_b}{d} \tag{5.5.1}
\]

Figure 5.26 shows a comparison between the models and the FE prediction. The green squares show the comparison when \(\rho_l\) is set as \(t_b/d\), as per Equation 5.1.2. The blue dots show the comparison when Equation 5.5.1 is applied. It can be seen that the resistance predictions are consistent and conservative when Equation 5.5.1 is applied.
A 0.8 factor is applied to the shear connection, since shear connectors subject to loads up to 90% of their failure load still undergo significant deformation, as shown in the connector load-slip curve (Figure 4.22). While conservative results are produced without the 0.8 adjustment factor, the factor of 0.8 produces a consistent margin of conservativeness, as Figure 5.26 shows.

It should be noted that the shear span for this value of $a/d$ tends to be much larger than the maximum stud spacings allowed by the design codes\cite{4,182}. If this is the case, a large number of rows are typically utilised in providing shear connection, which means that high degrees of shear connection are typical. To produce a shear connection percentage matching Case 1 utilising standard 19mm studs requires a combination of a large stud spacing, a low panel depth (to reduce $a$) and thick plates. Such a combination is extremely unlikely to be specified by a designer, since plate thickness is typically specified in proportion to the panel thickness. For this reason, further detailed characterisation of the effect of shear connection on Equation 5.1.1 is not justified. The amendment suggested in Equation 5.5.1 is sufficient for design.

Figure 5.26: Resistance predictions for panels taken from Table 5.9, with Equation 5.1.2 (green) and Equation 5.5.1 (blue) applied.
5.5.3 Group 3 - $a/d = 2$

The final group of unreinforced models has loads placed to produce an $a/d$ ratio of 2, which is intended to explore the region close to the bottom of Kani’s valley, as discussed in Section 5.3. Table 5.10 shows the cases included in this group.

Table 5.10: Summary of cases included in Group 3

<table>
<thead>
<tr>
<th>Test</th>
<th>$h$ (mm)</th>
<th>$a/d$</th>
<th>$\rho_{flexural}$ (%)</th>
<th>$\rho_{shear}$ (%)</th>
<th>$\gamma$</th>
<th>$\delta_{total}$ (mm)</th>
<th>$U_{bending}$ (mm)</th>
<th>$U_{shear}$</th>
<th>$R_{Test}/R_{Design}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>-</td>
<td>28</td>
<td>13.0</td>
<td>1.00</td>
<td>0.71</td>
<td>1.17</td>
</tr>
<tr>
<td>2</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>-</td>
<td>37</td>
<td>15.0</td>
<td>1.00</td>
<td>0.95</td>
<td>1.12</td>
</tr>
<tr>
<td>3</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>-</td>
<td>48</td>
<td>17.0</td>
<td>0.82</td>
<td>1.00</td>
<td>1.21</td>
</tr>
<tr>
<td>4</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>-</td>
<td>60</td>
<td>19.0</td>
<td>0.66</td>
<td>1.00</td>
<td>1.19</td>
</tr>
<tr>
<td>5</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>-</td>
<td>73</td>
<td>21.0</td>
<td>0.54</td>
<td>1.00</td>
<td>1.23</td>
</tr>
<tr>
<td>6</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>-</td>
<td>88</td>
<td>23.0</td>
<td>0.45</td>
<td>1.00</td>
<td>1.32</td>
</tr>
<tr>
<td>7</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>-</td>
<td>104</td>
<td>25.0</td>
<td>0.40</td>
<td>1.00</td>
<td>1.45</td>
</tr>
<tr>
<td>8*</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>-</td>
<td>121</td>
<td>27.0</td>
<td>0.40</td>
<td>1.00</td>
<td>1.15</td>
</tr>
<tr>
<td>9*</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>-</td>
<td>140</td>
<td>29.0</td>
<td>0.40</td>
<td>1.00</td>
<td>1.14</td>
</tr>
<tr>
<td>10*</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>-</td>
<td>160</td>
<td>31.0</td>
<td>0.40</td>
<td>1.00</td>
<td>1.16</td>
</tr>
<tr>
<td>11*</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>-</td>
<td>112</td>
<td>26.0</td>
<td>0.40</td>
<td>1.00</td>
<td>1.33</td>
</tr>
</tbody>
</table>

Note: cases marked with an asterisk were added to the study later to improve the precision of the study.

Figure 5.27 shows a selection of results of the FE modelling.
The results in this group do not show a resistance proportional to the shear connection, as found in Group 1 and Group 2. For low degrees of shear connection, the resistance observed in the model drops, which is consistent with the other groups. This trend continues for shear connections up to 120%. At this level of shear connection a steep drop in resistance is observed, with higher degrees of shear connection then retaining the lower resistance. It can be seen in Figure 5.27 that the stiffness of the tests with 116% and 136% have almost identical stiffnesses, but the point at which the crack opens occurs at a considerably lower load.

A number of extra models were added to the group to help define this relationship with higher resolution. The drop in resistance for higher degrees of shear connection can clearly be seen in Figure 5.28.
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

Figure 5.28: Shear resistance of unreinforced panels with an $a/d$ ratio of 2 for various degrees of shear connection

The drop in resistance at higher degrees of shear connection is a consequence of a shift in Kani’s valley. As first discussed in Section 5.3.2, Kani’s valley predicts that beams with more reinforcement can become more brittle, as a result of an over-concentration of stress in critical areas. The effect is seen in the test results, as discussed by Sagaseta and Francis[152].

5.5.4 Reduced stud spacing

The results of the modelling described in Sections 5.5.1, 5.5.2 and 5.5.3 have shown that the shear resistance of the panels without shear reinforcement is influenced by the level of shear connection. An adjustment to the Eurocode model is presented in Equation 5.5.1 that accounts for the shear connection stiffness.

The previous groups have all included shear connectors at a set spacing of 200mm, with changes in degree of shear connection produced by changes to the diameter of the connector. As a final check, a series of cases were modelled with studs placed at a spacing of 100mm. If can expected that the difference in behaviour of panels with similar degrees of shear connection should be negligible, even if the spacing is different.

Figure 5.29 shows a comparison between Group 2 Model 2 (See Table 5.9) and a case
with a reduced shear connector spacing. Since shear connector resistance is proportional to the square of the diameter, a 25% reduction in diameter leads to a 50% reduction in resistance. Since the case includes twice as many rows in the shear span, this produces an identical degree of shear connection.

![Graph showing load-deflection curves for different spacing and diameter cases.](image)

**Figure 5.29:** Load-deflection curves for Table 5.9 case 2, and an alternative design with reduced connector spacing and diameter

It can be seen that very similar load-deflection curves are found from the model. The 100mm spacing case shows a slight increase in resistance, though not enough to require enhanced consideration. It is expected that these cases would have slightly increased resistance, as the longitudinal shear force is more evenly distributed along the interface, leading to lower stress concentration in the concrete at the base of the studs.

Figure 5.30 shows a similar result for case 5 of group 2, for a higher degree of shear connection ($\mu = 110\%$).
On the basis of this analysis, it can be concluded that characterisation on the basis of degree of shear connection is appropriate. No additional adjustment factors are required for Equation 5.5.1 to account for connector spacing.

### 5.5.5 Comparison with the Eurocode model

Figure 5.31 shows an updated version of Figure 5.3, with the FE cases included.
Reflecting the same data as Figure 5.26, this plot shows that most of the FE cases are conservatively predicted, though a small number are cases are predicted unconservatively. Figure 5.32 shows an updated plot, with Equation 5.5.1 applied when calculating the percentage of bending reinforcement.
Again reflecting the data shown previously, this plot shows that the improved predictions are found for the FE tests that were predicted unconservatively. However, this plot also shows that the predictions for the a number of test results have become more conservative.

However, this plot also shows that application of Equation 5.5.1 to beams with an $a/d \leq 2$ tends to overestimate the reduction in resistance resulting from the reduced degree of shear connection. This conservativeness is produced by two effects; firstly, beams with low $a/d$ ratios also tend to have a low degree of shear connection. Secondly, the effect of reduced shear connection on arching action is less than it is on other shear transfer actions, as shown by Figure 5.21.

Further work could be carried out to improve these overly conservative predictions. A possible route would be application of an enhancement factor to Equation 5.5.1 for cases with low $a/d$ ratios, with a similar form to the $\beta$ factor shown in Figure 5.9. Justification of such a factor would require an extended parametric study, with inclusion of intermediate groups of $a/d$ ratios i.e. 1.25, 1.5, 1.75 etc. Even with this study considerable improvement may not be possible, given the model would have to account for both the presence and shift of Kani’s valley with degree of shear connection (as discussed in Section 5.5.3).

Although a more extensive parametric study may bring improvements to the Eurocode
model (with the adjustment included in Equation 5.5.1), it has been recognised that alternative approaches have shown better results, especially when compared to testing. The Strut & Tie method, as discussed in 5.3.2, is known to produce better predictions for cases with low $a/d$ ratios. For cases with an $a/d$ less than 2, the Strut & Tie method is recommended. The method presented in Sagaseta and Francis[152] may be used in design. Alternatively, strain based iterative modelling approaches have also shown better predictions in most cases. This is explored further in Section 5.7.4.

5.5.6 Conclusion

The FE study of unreinforced panels subject to shear loads has supported the previous design consensus, which suggests that theoretical models for behaviour of conventional RC construction are applicable to SCS panel design. The contribution of the tension plate to the resistance has been shown to be accounted for well be the percentage of reinforcement term $\rho_l$, despite the fact the plate is not embedded in the concrete. Evidence for the presence of Kani’s valley is seen, with the position of the valley being affected by the degree of shear connection.

The degree of shear connection has been shown the have an effect on resistance. For panels with loads placed close to the support the effect is negligible, as the resistance in all cases are still conservative. For the panels with loads at a larger distance away from the support, the change is significant, with predictions for panels with lower degrees of shear connection sometime proving to be unconservative. For this reason an adjustment is suggested to the $\rho_l$ term, as presented in Equation 5.5.1. The new adjustment produces a consistent and conservative results for different levels of shear connection. However, the new model is extremely conservative for cases with a low $a/d$ ratio. For these cases, alternative methods, such as the Strut & Tie method, may produce better results.

A final check on a series of models with reduced stud spacings has shown that characterisation based on degree of shear connection is appropriate. The difference in stiffness and failure point for models with similar degrees of shear connection is found to be small. The evidence also suggests that the cases modelled as part of the parametric study are the most conservative.
5.6 FE study - Panels with shear reinforcement

This section describes the equivalent parametric study for panels reinforced in shear. For the reasons described in Chapter 2, modern SCS panels typically include tie-bars, which hold the plates together at the construction stage, while the concrete is curing. Once the concrete is cured, the tie-bars can act as shear reinforcement.

As discussed previously in Section 5.4.2, many of the cases included in the parametric study would tend not to be specified in real design, since the stud sizes specified are non-standard. For larger \( a/d \) ratios, it becomes increasingly difficult to define cases for investigation with lower degrees of shear connection, since the shear span is large enough to mobilise a number of stud rows.

For reinforced panels, designing test cases with lower degrees of shear connection becomes increasingly difficult. To produce failure via a reinforced mechanism the resistance of the panel must include a sufficient number of bars to make the resistance produced by the reinforced model (Equation 5.1.1) higher than the resistance produced by the unreinforced model (Equation 5.1.6). These steel bars contribute significant interfacial-shear resistance, which increases the degree of shear connection. Tie-bar sizes in typical designs will tend to be 20mm or 24mm, which is in excess of the 16mm or 19mm stud connectors typically employed.

Group 5, as described in Section 5.6.2, proved to be the hardest case to design, given the high \( a/d \) ratio of 3. The eventual study included cases with a depth of 500mm, which is the lowest practical thickness that would be used in SCS panel design.

5.6.1 Group 4 - \( a/d = 1 \)

Group 4 includes loads placed close to the support, producing an \( a/d \) ratio of 1. Like the unreinforced cases modelled in Group 1, the dominant resistance mechanism at these low distances is expected to be arching action, though arching action contributes a lower proportion of resistance to the overall resistance than in unreinforced cases.

Table 5.11 shows the cases included in this group. As with group 1, 8mm plates are used to allow higher degrees of shear connection.
Table 5.11: Summary of cases included in Group 4

<table>
<thead>
<tr>
<th>Test</th>
<th>n</th>
<th>a/d</th>
<th>P/ftannl</th>
<th>P/shear</th>
<th>γ</th>
<th>d_stud</th>
<th>U_bending</th>
<th>U_shear</th>
<th>R_test / R_design</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>800</td>
<td>1.00</td>
<td>1.00</td>
<td>0.23</td>
<td>54</td>
<td>19.0</td>
<td>1.00</td>
<td>1.00</td>
<td>0.99</td>
</tr>
<tr>
<td>2</td>
<td>800</td>
<td>1.00</td>
<td>1.00</td>
<td>0.23</td>
<td>61</td>
<td>21.0</td>
<td>0.89</td>
<td>1.00</td>
<td>1.14</td>
</tr>
<tr>
<td>3</td>
<td>800</td>
<td>1.00</td>
<td>1.00</td>
<td>0.23</td>
<td>68</td>
<td>23.0</td>
<td>0.80</td>
<td>1.00</td>
<td>1.17</td>
</tr>
<tr>
<td>4</td>
<td>800</td>
<td>1.00</td>
<td>1.00</td>
<td>0.23</td>
<td>76</td>
<td>25.0</td>
<td>0.71</td>
<td>1.00</td>
<td>1.21</td>
</tr>
<tr>
<td>5</td>
<td>800</td>
<td>1.00</td>
<td>1.00</td>
<td>0.23</td>
<td>85</td>
<td>27.0</td>
<td>0.64</td>
<td>1.00</td>
<td>1.26</td>
</tr>
</tbody>
</table>

The resistance predicted by Equation 5.1.6 for the cases presented in Table 5.11 is in excess of the resistances presented, due to the provision presented in Equation 5.3.2, which limits the resistance to the force that can be carried in the middle 75% of the shear span.

Figure 5.33 shows the results of the FE modelling.
As expected, the results show that the design model is conservative for all the cases in this group. The effect of changes in shear connection looks to be small, with drops in resistance only evident for the lowest level of shear connection. The unity factors for bending in these cases are close to 1 (as shown in Table 5.11), which suggests that the small drops in resistance may be due to the beginnings of bending failure, rather than shear.

### 5.6.2 Group 5 - \( a/d = 3 \)

Group 5 includes designs where the load is positioned at a relatively large distance away from the support, producing an \( a/d \) ratio of 3. In the equivalent unreinforced group (Group 2), the ultimate resistance of the cases was observed to be most sensitive to degree of shear connection at these \( a/d \) ratios. However, models for behaviour of RC beams suggest that reinforced beams should be less sensitive to degree of shear connection, as the cracks that form are controlled by the shear reinforcement. The strain localisation observed in unreinforced cases is less severe when reinforcement is
present.

Table 5.12 shows the cases included in this group.

Table 5.12: Summary of cases included in Group 5

<table>
<thead>
<tr>
<th>Test</th>
<th>$h$ (mm)</th>
<th>$a/d$</th>
<th>$\rho_{flexural}$ (%)</th>
<th>$\rho_{shear}$ (%)</th>
<th>$\gamma$</th>
<th>$d_{stud}$ (mm)</th>
<th>$U_{bending}$</th>
<th>$U_{shear}$</th>
<th>$R_{Test} / R_{Design}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>500</td>
<td>3.00</td>
<td>2.40</td>
<td>0.16</td>
<td>58</td>
<td>18.0</td>
<td>0.89</td>
<td>1.00</td>
<td>1.22</td>
</tr>
<tr>
<td>2</td>
<td>500</td>
<td>3.00</td>
<td>2.40</td>
<td>0.16</td>
<td>67</td>
<td>20.0</td>
<td>0.78</td>
<td>1.00</td>
<td>1.26</td>
</tr>
<tr>
<td>3</td>
<td>500</td>
<td>3.00</td>
<td>2.40</td>
<td>0.16</td>
<td>76</td>
<td>22.0</td>
<td>0.68</td>
<td>1.00</td>
<td>1.25</td>
</tr>
<tr>
<td>4</td>
<td>500</td>
<td>3.00</td>
<td>2.40</td>
<td>0.16</td>
<td>86</td>
<td>24.0</td>
<td>0.60</td>
<td>1.00</td>
<td>1.25</td>
</tr>
<tr>
<td>5</td>
<td>500</td>
<td>3.00</td>
<td>2.40</td>
<td>0.16</td>
<td>97</td>
<td>26.0</td>
<td>0.53</td>
<td>1.00</td>
<td>1.33</td>
</tr>
</tbody>
</table>

Figure 5.33 shows the results of the FE modelling.

Figure 5.34: Load-deflection curves for SCS panels subject to shear loading, as shown in Table 5.12
It can be seen that the shear connection percentage has an effect on the stiffness of the panels as they reach loads close to failure. This can be explained by the increased stiffness of the tension chord, which reduces the propensity of cracks to open on the tension side.

For the majority of cases the ultimate resistance appears to be unaffected, with only a small change in peak resistance observed. Case 1 (blue line in Figure 5.34) is the only case that shows a drop in resistance. However, this appears to be as a result of bending failure, rather than shear; the peak load of 960kN shown is close to the predicted point of bending resistance (990 kN), which previous models has shown to be an accurate predictor.

The model gives a predicted resistance less than the resistance observed in the FE model for all cases. It is therefore proposed that the design model is suitable for use with SCS panels, without specific adjustments for the system.

5.6.3 Group 6 - $a/d = 2$

The final group in the study is group 6, concerned with the behaviour of reinforced panels subject to a load producing an $a/d$ ratio of 2. Theory suggests that Kani’s valley is not as prevalent for reinforced members, since the behaviour is characterised by the formation of a number of cracks, rather than a small number of concentrated cracks seen in unreinforced panels.

Table 5.13 shows the cases included in this group.

<table>
<thead>
<tr>
<th>Test</th>
<th>$h$ (mm)</th>
<th>$a/d$</th>
<th>$\rho_{flexural}$</th>
<th>$\rho_{shear}$</th>
<th>$\gamma$</th>
<th>$d_{stud}$ (mm)</th>
<th>$U_{bending}$</th>
<th>$U_{shear}$</th>
<th>$R_{test}/R_{design}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>0.16</td>
<td>58</td>
<td>18.0</td>
<td>0.68</td>
<td>1.00</td>
<td>1.31</td>
</tr>
<tr>
<td>2</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>0.16</td>
<td>67</td>
<td>20.0</td>
<td>0.59</td>
<td>1.00</td>
<td>1.50</td>
</tr>
<tr>
<td>3</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>0.16</td>
<td>76</td>
<td>22.0</td>
<td>0.52</td>
<td>1.00</td>
<td>1.53</td>
</tr>
<tr>
<td>4</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>0.16</td>
<td>86</td>
<td>24.0</td>
<td>0.46</td>
<td>1.00</td>
<td>1.59</td>
</tr>
<tr>
<td>5</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>0.16</td>
<td>97</td>
<td>26.0</td>
<td>0.41</td>
<td>1.00</td>
<td>1.66</td>
</tr>
<tr>
<td>6</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>0.16</td>
<td>109</td>
<td>28.0</td>
<td>0.40</td>
<td>1.00</td>
<td>1.77</td>
</tr>
<tr>
<td>7</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>0.16</td>
<td>122</td>
<td>30.0</td>
<td>0.40</td>
<td>1.00</td>
<td>1.80</td>
</tr>
<tr>
<td>8</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>0.16</td>
<td>136</td>
<td>32.0</td>
<td>0.40</td>
<td>1.00</td>
<td>1.88</td>
</tr>
<tr>
<td>9</td>
<td>800</td>
<td>2.00</td>
<td>1.50</td>
<td>0.16</td>
<td>150</td>
<td>34.0</td>
<td>0.40</td>
<td>1.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>
Figure 5.35 shows the results of the FE modelling.

![Graph showing load-deflection curves for SCS panels subject to shear loading, as shown in Table 5.13]

**Figure 5.35:** A selection of load-deflection curves for SCS panels subject to shear loading, as shown in Table 5.13

As with the behaviour observed in Group 4 (Section 5.6.1), the model is shown to be conservative in all cases. The resistance predictions show sensitivity to the degree of shear connection, but the failure loads are consistent with the failure loads predicted by the bending model. There is no abrupt drop in resistance for higher degrees of shear connection, as shown in Figure 5.36.
Figure 5.36: Shear resistance of reinforced panels with an $a/d$ ratio of 2 for various degrees of shear connection

5.6.4 Comparison with the Eurocode model

Figure 5.37 shows an updated version of Figure 5.4, with the FE cases included.
5.6.5 Conclusion

A parametric study of the behaviour of SCS panels reinforced in shear is largely consistent with theoretical predictions, based on the behaviour of conventional reinforced concrete beams. In all of the cases tested the Eurocode based design model, as presented in Equation 5.1.6, has produced conservative results. Cases where arching action is prevalent have proved to have significantly higher resistances than model predictions, again in line with expectations.

Sensitivity to degree of shear connection has been found to be less than equivalent unreinforced designs. Decrease in stiffness have been observed with higher degrees of shear connection, which can be accounted for in analysis (see Chapter 6).

On the basis of these results, no changes to the model for reinforced panels are suggested. Designers who apply the model as presented in Section 5.1.2 will obtain conservative resistances for designs within the limitations presented in P414\(^4\). This conclusion is supported by both the test results (as presented in Figure 5.4) and the FE work.

Figure 5.37 shows that the results are reasonable and conservative in all cases, for both the FE models and the tests.
5.7 Alternative resistance models for members without shear reinforcement

As explored in the previous sections, the current Eurocode based design model for SCS panels has been shown to produce reasonable results, but can be very conservative when arching taken into account. The model for unreinforced panels does not currently account well for reduced degree of shear connection, though an amendment given in Equation 5.5.1 is capable of accounting for these effects. No amendments are required for reinforced SCS panel design.

Alternative models from around the world have varying degrees of sophistication. A number of the models take into account the degree of shear connection (for example, Equation 5.7.13). For these reasons, it might be expected that one or more of these models might produce more accurate predictions than the current Eurocode based model. The aim of this section is therefore to test the accuracy of these models against the test database.

While the test database is sufficiently large for this purpose, it has been recognised that the database lacks sufficient data for tests on panels with low degrees of shear connection. This makes it impossible to judge the accuracy of models which include terms that account for changes in shear connection. For this reason, the comparison study is supplemented with FE evidence. The cases included are taken from Tables 5.8 to 5.13.

For ease of comparison terms have been translated into their Eurocode equivalents where possible.

5.7.1 AISC N690-12

Guidance for design of ‘Safety-Related Steel Structures for Nuclear Facilities’ (as SCS panels may be defined) in the US is governed by ANSI / AISC N690-12[9]. This document refers to the American Concrete Institute document ACI 349-06[8] Section 11.3 for a design model for out-of-plane shear, with no additional modifications suggested.

The resistance of an unreinforced SCS panel against out-of-plane shear is given in Equation 5.7.1:

\[ V_{C,ACI} = \min \left\{ \left( 0.16\sqrt{f_{ck}} + 17\rho_{w} \frac{1}{\lambda} \right) b_{wy}d : 0.29\sqrt{f_{ck} b_{wy}d} \right\} \]  

(5.7.1)
With $\lambda$ taken as 1 if $\lambda$ is greater than 1 i.e. $\lambda \leq 1$. All remaining terms are as defined in Section 5.1.1.

The design model for out-of-plane shear is relatively simple. The base concrete model is similar to the original equation for resistance presented in the Bi-Steel Manual[^24]. This model was eventually superseded by the resistance equation from Eurocode 2[^30], as presented in Equation 5.1.1. As with the Eurocode model, the equation is dimensionally inconsistent, meaning it is only correct when the parameters are entered in the correct units. Somewhat unusually for a US standard, the units required are SI (e.g. concrete strength in $N/mm^2$).

Some enhancement is allowed for arching action. However, limiting $\lambda$ to 1 means the enhancement is only available for loads placed particularly close to the support i.e. $a/d \leq 1$. The maximum load ceiling stops this enhancement tending toward infinity as $\lambda$ approaches zero.

Figure 5.38 shows a comparison between the resistances predicted by the model and test results. Points above the red line show the model is conservative for that case.

![Figure 5.38: Ratio of test (blue) or FE predictions (green) to resistances predicted by Equation 5.7.1 for members without shear reinforcement](image)

The results presented in Figure 5.38 suggest that the AISC model is a good predictor
of resistance for $a/d$ ratios greater than 2. The model for arching appears to be overly conservative, with tests with an $a/d$ ratio less than 2 showing considerably more resistance than predicted. The results produced are very similar to those produced by the Eurocode model, as shown in Figure 5.3.

Three of the cases with low degrees of shear connection from the FE study with $a/d$ ratios equal to 3.0 are predicted unconservatively. Should this model be used for design of panels of this character an additional adjustment factor should be applied. However, for the reasons given in Section 5.6 it is considered that designs with low degrees of shear connection and high $a/d$ ratio are extremely unlikely to be realised in practice.

5.7.2 JEAC-4618

The Japanese SCS code JEAC-4618\[87\] includes models for shear resistance. Although not readily available in English, the work in this case is based on a translation provided as part of the SCIENCE project.

The code presents a considerable departure from the Eurocode philosophy, with resistance models being provided for both ‘sustained load’ and ‘temporary load’. Section 1.2.4 of the guide elaborates that these limit states are the equivalent to the ULS limit state and earthquake accidental limit states in the Eurocode respectively.

It should be noted that this code is an allowable stress code, as opposed to the limit state models proposed by the other guides. Direct comparison between the models should still be possible, since the starting point for both models should still be the characteristic resistance. However, direct comparisons between the design values of the resistance are not possible. Examination of the allowable stresses presented in Table 1.2.4-1 suggest that the allowable stresses are particularly onerous, given only one-third of the concrete strength may be utilised.

The differences between the models can be considerable, with resistances provided by Equation 5.7.3 (temporary) often double the resistances predicted by Equation 5.7.2 (sustained). However, both models should still compare favourably with the test results. For this reason, both models are presented in the comparison below.

**Sustained loading**

The resistance of an SCS panel against a sustained out-of-plane shear force is given by Equation 5.7.2:
Chapter 5: Out-of-plane shear resistance

\[ V_C = 0.16 \sqrt{f_c b_w d} \]  \hspace{1cm} (5.7.2)

The model provides a virtually identical resistance to one predicted by the AISC model (shown in Equation 5.7.1), though there is no enhancement for arching action. A comparison plot is shown in Figure 5.39.

![Comparison plot](image)

**Figure 5.39:** Ratio of test (blue) or FE predictions (green) to resistances predicted by Equation 5.7.2 for members without shear reinforcement

As with the AISC model, the JEAC model provides an appropriate model for use in design. The model is conservative in the large majority of cases, though a small number of cases with low degrees of shear connection from the FE study are predicted unconservatively.

**Temporary loading**

The resistance of an SCS panel against a sustained out-of-plane shear force is given by Equation 5.7.3. The philosophy of this model differs considerably from Equation 5.7.2, with terms accounting for the effects of arching action and longitudinal shear transfer.
\[ V_C = \min \left( 0.31 \sqrt{f_c b_w d} : 0.8 \left( Q_{arch} + Q_{bond} \right) \right) \]  \hspace{1cm} (5.7.3)

Where:

\[ Q_{arch} = \frac{1}{9 \lambda} v_2 f_c b_w d \]  \hspace{1cm} (5.7.4)

With \( \lambda \) taken as 1 if \( \lambda \) is less than 1, or 3 if \( \lambda \) is greater than 3 i.e. \( 1 \leq \lambda \leq 3 \).

\[ v_2 = 0.8 + 0.05 \lambda \]  \hspace{1cm} (5.7.5)

\[ Q_{bond} = 0.45 \mu_{avg} b_w d \]  \hspace{1cm} (5.7.6)

\[ \mu_{avg} = \frac{P_{rd}}{B_1B_2} \]  \hspace{1cm} (5.7.7)

Where:

- \( P_{rd} \) is the resistance of a shear connector
- \( B_1 \) is the connector spacing in the longitudinal direction
- \( B_2 \) is the connector spacing in the transverse direction

Figure 5.40 shows a comparison between the resistances predicted by the model and test results. Points above the red line show the model is conservative for that case.
The results show that Equation 5.7.3 gives unconservative results for many of the tests in the database.

As discussed previously, the JAEC code is an allowable stress code. Given the allowable stresses are particularly onerous, it can be reasonably expected that the designers have calibrated their model to give suitably reliable results. However, the model should not be used in Eurocode design without a new calibration study, which will likely find a very large partial factor on resistance is needed. This conclusion is supported by Sener and Varma\textsuperscript{[157]}.

### 5.7.3 KEPIC-SNG

The South Korean code for SC walls KEPIC-SNG\textsuperscript{[97]} also includes a model for shear resistance. Much of the calculation is consistent with the Japanese short term loading model, though the arching model (Equation 5.7.10) is a considerably more complex equation, though based on the same terms.

The resistance for unreinforced panels is given by Equation 5.7.8:
\[ V_{n,SNG} = V_{cr,SNG} + V_{ar,SNG} \]  
(5.7.8)

Where:

\[ V_{cr,SNG} = 0.16 \sqrt{f_c' b_w d} \]  
(5.7.9)

\[ V_{ar,SNG} = \frac{1}{2} \left[ \sqrt{4 \phi_r (1 - \phi_r) + \lambda^2 - \lambda} \right] v_2 f_c b_w d \]  
(5.7.10)

\[ \phi_r = \frac{\mu_{avg}}{f_c} \lambda \]  
(5.7.11)

\[ \lambda = \frac{s}{d} \]  
(5.7.12)

With \( \lambda \) taken as 0.5 if \( \lambda \) is less than 0.5, or 6 if \( \lambda \) is greater than 6 i.e. \( 0.5 \leq \lambda \leq 6 \).

\[ \mu_{avg} = \frac{P_{rd}}{s^2} \]  
(5.7.13)

\[ v_2 = 0.8 + 0.05 \frac{\lambda}{2} \]  
(5.7.14)

Figure 5.41 shows a comparison between the resistances predicted by the model and test results. Points above the red line show the model is conservative for that case.
It can be seen that many of the points are below the red line, indicating unconservative predictions.

The effect of changes in the shear connection provision is substantial in this model, particularly for the case where $a/d$ is equal to 2. Results from Section 5.5.3 suggest that this is not appropriate. The key driver of this inconsistency is that the arching term has a large dependency on the shear connection term $\mu_{avg}$, as given by Equation 5.7.13, which is the longitudinal shear resistance per meter of plate. No attempt is made to limit the shear connection term if the plate is not capable of sustaining the forces developed by the shear connectors. Additionally, the arching term allows for increased resistances up to $a/d$ ratios of 6, which work on RC beams suggests would not be enhanced.

Given the low predictions, for both the tests and the FE results, it is recommended that these formulas are not implemented in any limit state design guide unless further work is carried out.
5.7.4  *fib* Model Code 2010

One of the most recent codified models for shear resistance of concrete structures is included in the *fib* Model Code 2010\[^{61}\]. This model is based on international research from a number of groups. As described by Sigrist et al.\[^{161}\], the aim of the model code is to move away from fully empirical models, instead focusing on developing a physical interpretation of behaviour on the “meso-scale”\[^{161}\].

A key unique feature of the *fib* Model Code is the concept of ‘Level of approximation’. In this paradigm, the designer starts with a simple model, which removes complexity through the use of conservative assumptions. If the design resistance is found to be insufficient the designer is able to escalate to a more accurate and less conservative model, though the design model requires more effort to apply. In the case of the *fib* model, increased accuracy is achieved through the use of an iteration of the strain term $\varepsilon_{x'}$, as presented in Equation 5.7.18.

The level of approximation approach is considered particularly helpful for SCS panel design, since the proportions of the panel are usually not sized by shear resistance requirements, as discussed in Chapter 2. It is expected that the majority of designs will be covered by the first ‘level of approximation’. Comparisons between tests and both levels of approximation appropriate for unreinforced panel design are shown below.

**Level of approximation (LoA) I**

The characteristic resistance for panels according to level of approximation I is given by Equation 5.7.15:

$$V_{Rd,c} = \frac{180}{1000 + 1.25z} \sqrt{f_{ck} b_w}$$  \hspace{1cm} (5.7.15)

Figure 5.42 shows a comparison between the resistances predicted by the model and test results. Points above the red line show the model is conservative for that case.
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

As expected, the model shows a high level of conservativeness in all cases. Such a result is consistent with the aims of the first level of approximation (as shown in Figure 1 of the paper by Sigrist et al.\cite{161}). The design model does not include any terms accounting for arching action, meaning cases with low $a/d$ ratios are more conservatively predicted than cases with higher $a/d$ ratios.

Level of approximation (LoA) II

The resistance for panels according to level of approximation II is given by Equation 5.7.16:

$$V_{Rd,c} = \frac{0.4}{1 + 1500\varepsilon_x} \frac{1300}{1000 + k_{dg}z} \sqrt{f_{ck}z b_w}$$  \hspace{1cm} (5.7.16)

Where:

$$k_{dg} = \max \left\{ \frac{32}{16 + d_g} : 0.75 \right\}$$  \hspace{1cm} (5.7.17)
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

\( d_g \) is the aggregate size in mm

The strain term is calculated as shown in Equation 5.7.18. This equation is a reproduction of Equation 7.3-16 of the code, with terms related to axial force or pre-stress removed.

\[
\varepsilon_x = \frac{1}{2E_s A_s} \left( \frac{M_{Ed}}{0.9d} + V_{Ed} \right)
\]

(5.7.18)

Iteration is required to produce a design resistance. For cases with a single point load, this is achieved by setting a value of \( \varepsilon_x \) such that \( V_{Rd,c} = V_{Ed} \).

For the purposes of this calculation, the stiffness of the panel is assumed to be entirely contributed by the steel. As the results in Chapter 6 show, significant stiffness is contributed by the concrete. However, the assumption of a steel only contribution is conservative in this case, since increasing assumed stiffness in Equation 5.7.18 leads to a reduced value of \( \varepsilon_x \) for a given load.

![Figure 5.43: Ratio of test (blue) or FE predictions (green) to resistances predicted by Equation 5.7.16 for members without shear reinforcement](image)

Again as expected, the higher level of approximation model shows better predictions.
for the test results than the first level of approximation. The results for \(a/d\) ratios above 2.5 are particularly good, with all tests resistances being predicted within 50% of the resistance recorded in testing.

Comparisons between the FE results and the model show that the resistance predictions are lower than the FE result in a number of cases. Further investigation of these results showed that the unconservative cases tended to be those with degrees of shear connection less than 100%, with the worst predictions being calculated for the tests with the lowest degrees of shear connection (for example, Test 1 of Table 5.9 is the most unconservative prediction). In accordance with the model applied in Section 5.5.2, the following adjustment is made to Equation 5.7.18:

\[
\varepsilon_x = \frac{1}{2E_s A_{s,\text{min}}} \left\{ 1 : 0.8 \mu_b \right\} \left( \frac{M_{Ed}}{0.9d} + V_{Ed} \right) \quad (5.7.19)
\]

Figure 5.44 shows the same results as Figure 5.43, with the adjustment presented in Equation 5.7.19.

![Figure 5.44: Ratio of test (blue) or FE predictions (green) to resistances predicted by Equation 5.7.16 and Equation 5.7.19 for members without shear reinforcement](image)

The adjustment results in an increase in resistance in all of the points that lie below
the comparison line. All but the cases with the lowest degrees of shear connection are predicted conservatively. Despite this, a number of the points from the continuous tests by Takeuchi et al.\cite{174} are predicted much more conservatively. This compromise is considered acceptable to achieve a design model on the safe side.

### 5.7.5 Comparison between models

Figures 5.38 to 5.44 show comparisons between a number of different models for shear resistance of unreinforced panels, from a number of different countries. Despite modelling the same behaviour, the predictions produced by each of the models varies considerably, even for the same tests.

The models used in the USA (Figure 5.39) and Europe (Figure 5.3) tend to produce conservative results for all design situations. The effect of arching action tends to be considerably underestimated, which leads to conservative results for low $a/d$ ratios.

The models used in Japan and Korea are found to produce unconservative results, often considerably higher than those produced by testing. However, these models are taken from allowable stress codes, which have a different reliability philosophy to the Eurocodes. The codes appear to have particularly onerous limits on the allowable stress in the concrete, which may compensate for the unconservative results from the models. However, given these results over-estimate the resistance in many cases, a large partial factor would need to be applied if these models were to be used in a code that follows a limit-state philosophy, like the Eurocodes. This conclusion is supported by Sener and Varma\cite{157}.

The final model included in the study is the *fib* Model Code 2010\cite{61}. The level of approximation concept in this code is shown to work well with SCS panel design, with the first level producing simple but conservative results, while the 2nd level produces the most accurate approximation. Although iteration is required to establish a resistance, the process is relatively simple. As with the model based on the current Eurocode, an additional adjustment is suggested to account for degree of shear connection, as shown in Equation 5.7.18. This model shows the best consistency with both the test and FE results, meaning it is the model recommended for inclusion in future SCS panel design guidance. However, further testing is recommended for panels with low degrees of shear connection, since the current conclusions are based mainly on finite element models.
5.8 Alternative resistance models for members with shear reinforcement

This section presents comparisons between alternative design models for reinforced SCS panels against tests from the test database (See Table 5.5).

As with the unreinforced study, comparisons are also made with the FE predictions, in order to explore cases with lower degrees of shear connection that are lacking in the test database. Cases are taken from Tables 5.8 to 5.13.

Terms have been translated into Eurocode equivalents where possible.

5.8.1 AISC N690-12

The AISC design manual\cite{182} adopts a relatively simple additive approach for reinforced panel design, with the contribution of the reinforcement being assessed separately to the concrete contribution, which is assessed using the model presented in Section 5.7.1.

The total resistance of an reinforced SCS panel against out-of-plane shear is given in Equation 5.8.1:

$$V_{Rd,ACI} = V_{c,ACI} + V_{s,ACI}$$  \hspace{1cm} (5.8.1)

Where:

$$V_{Rd,ACI} = \min \left\{ A_{sw} f_{ykw} \frac{Z}{S} : 0.67 \sqrt{f_{ck} b_w d} \right\}$$  \hspace{1cm} (5.8.2)

The standard enforces a 0.75$d$ spacing limit for the shear reinforcement, which is identical to the limit proposed in Eurocode 2\cite{30} Section 9.3.2. However, the AISC design guide imposes an additional, more stringent, limit where the force in the steel is large i.e. $V \geq 0.33 \sqrt{f_{ck} b_w d}$, this spacing limit is reduced to 0.425$d$. This limit is particularly stringent, and is rarely met in the tests in the database. However, the force at which this limit is met is larger than the failure load of any test in the database, meaning it is not critical to the performance of any test.

Figure 5.45 shows a comparison between the resistances predicted by the model and test results. Points above the red line show the model is conservative for that case.
The results show a strong correlation between the design model predictions for the test database (See Table 5.5) and the FE predictions (Tables 5.8 to 5.13). The test results are also strongly predicted, with results for lower $a/d$ ratios tending to be slightly more conservative. No adjustments for shear connection appear to be required.

5.8.2 JEAC-4618

The Japanese design code for SC walls (JEAC-4618[87]) provides a set of equations to calculate the out-of-plane shear strength of reinforced SCS members. The contributions of the concrete and steel are again assessed separately, but the maximum is taken rather than the product.

As with the unreinforced model presented in Section 5.7.2, two models are presented, for ‘sustained load’ and ‘temporary load’.
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

Sustained loading

The resistance of an SCS panel against a sustained out-of-plane shear force is given by Equation 5.8.3.

\[
V_{Rd,C,\overline{EAC}} = \max \left\{ 0.16 \sqrt{f_c b_w d} : \min \left\{ Q_w : \frac{1}{2} 0.8 (Q_{arch} + Q_{bond}) \right\} \right\}
\]  
(5.8.3)

Where:

\[
Q_w = \max \left\{ 0.31 \sqrt{f_c b_w d} : 2.8 (\rho_{tf} y_t)^{\frac{3}{2}} b_w d \right\}
\]  
(5.8.4)

With \( \rho_{tf} y_t \) limited to 2MPa i.e. \( \rho_{tf} y_t \leq 2 \)

\[
Q_n = 0.5 A_{sa} \sqrt{f_c E_c}
\]  
(5.8.5)

With \( \sqrt{f_c E_c} \) limited to 880MPa i.e. \( \sqrt{f_c E_c} \leq 880 \)

Figure 5.46 shows a comparison between the resistances predicted by the model and test results. Points above the red line show the model is conservative for that case.

![Figure 5.46: Ratio of test (blue) or FE predictions (green) to resistances predicted by Equation 5.8.3 for members with shear reinforcement](image-url)
The results for this model are much more conservative than those produced by the AISC model presented in 5.8.1. Neglecting the concrete contribution appears to introduce considerable variation in the results, since cases with different relative proportions of steel and concrete strength can reasonably be expected to have different resistances. This conclusion holds against both the test results and the FE cases.

**Temporary loading**

The resistance of an SCS panel against a temporary out-of-plane shear force is given by Equation 5.8.6.

\[ V_{Rd,JE,A} = \max \left\{ 0.31 \sqrt{f_c b_w d} : \min \left\{ Q_w : 0.8 \left( Q_{arch} + Q_{bond} \right) \right\} \right\} \]  

(5.8.6)

Figure 5.47 shows the comparison between the resistances predicted by this model and either the test results or the results from FE results. Points above the red line show the model is conservative for that case.

![Diagram showing comparison between test, FE, and predicted resistances](image)

**Figure 5.47:** Ratio of test (blue) or FE predictions (green) to resistances predicted by Equation 5.8.6 for members with shear reinforcement

The results for the short term model tend to show better correlation with the tests and
FE. However, a number of the cases are predicted unconservatively, particularly for those with low \(a/d\) ratios. For these cases the resistance is dominated by the \(Q_{\text{bond}}\) term, calculated in Equation 5.7.6, which is independent of the amount of steel reinforcement. This term intends to take into account catenary transfer effects, is significant for low \(a/d\) ratios only. For longer shear spans, this action is disrupted by the flexural cracking.

As with the unreinforced model, this model cannot be considered suitable for use in a limit state code, where the characteristic resistance is required.

### 5.8.3 KEPIC-SNG

The final model included in the study is the South Korean code for SC walls KEPIC-SNG\[97].

Similarly to the AISC code, the steel resistance is added to the resistance component from the concrete. Arching action is also considered separately. The steel resistance is relatively simple, as shown in Equation 5.8.8.

\[
V_{n,SNG} = V_{cr,SNG} + V_{s,SNG} + V_{ar,SNG} \quad (5.8.7)
\]

\[
V_{s,SNG} = \rho_f f_y b_w d \quad (5.8.8)
\]

With \(\rho_f f_y\) limited to 2MPa i.e. \(\rho_f f_y \leq 2\)

Figure 5.48 shows the comparison between the resistances predicted by this model and the results of testing and FE. Points above the red line show the model is conservative for that case.
The results in this case show the KEPIC model is unconservative for the majority of cases. These results are consistent with the comparisons produced for the unreinforced model shown in Figure 5.41, which also show results that are unconservative to a large degree. Again, this model should not be used in design in this form.

5.8.4 *fib* Model Code 2010

The *fib* Model Code 2010\[^{[61]}\] also includes a model for beams reinforced in shear. The Model code employs the concept of the inclined compressive stress field, which is also the basis of the Eurocode model, presented in Section 5.1.2. However, the model employs a different approach to the calculation of the effective angle of the stress field, depending on the level of approximation, which can substantially change the resistance.

As with the unreinforced model, discussed in Section 5.7.4 the concept of level of approximations is employed. For the first level of approximation (LoA) I, the approach is essentially the same as the variable inclination method adopted in the Eurocode (See Section 5.1.2), where the angle of the stress field is set between a value...
of $\theta_{\text{min}}$ and 45°, with $\theta_{\text{min}}$ being set by the level of axial force in the beam. For all the cases in the database there are no additional axial forces, meaning $\theta_{\text{min}}$ is set equal to 30°. It can be seen that is more conservative than the limit imposed by the Eurocode, which is 21.8° (See Equation 5.1.7). For the two higher levels of approximation (II and III), more sophisticated (strain based) models are employed to calculate the stress-field angle, both of which involve iteration.

Figure 5.49 shows a comparison between the results that might be achieved for a typical case using the three levels of approximation. Since the test results all fall in the region of $0.0 \leq \rho w y_d / f_{cd} \leq 0.1$, as shown in Figure 5.2, it can be seen the choice of level of approximation can have a substantial effect on the prediction.

![Figure 5.49: Comparison of level I, II and III results for members with $f_{ck} = 50\, \text{MPa}$](taken from Figure 7.3-11 of the fib Model Code 2010[^61])

**Level of approximation (LoA) I**

The resistance for panels according to level of approximation I is given by Equation 5.8.9:

$$V_{Rd} = V_{Rd,s} \leq V_{Rd,max} \quad (5.8.9)$$

Where:

[^61]: fib Model Code 2010
\[ V_{Rd,s} = \frac{A_{sw}}{S_{w}} z f_{yud} \cot \theta \]  
(5.8.10)

\[ V_{Rd,max} = k_c \frac{f_{ck}}{f_c} b_w z \sin \theta \cos \theta \]  
(5.8.11)

\( \theta \) is taken as 30\(^\circ\), as appropriate for members with no significant tensile or compressive axial force.

\[ k_c = k_{c} \times \min \left\{ \left( \frac{30}{f_{ck}} \right)^{1/3} : 1.0 \right\} \]  
(5.8.12)

\[ k_{c} = 0.55 \]  
(5.8.13)

Figure 5.50 shows a comparison between the resistances predicted by the model and test results. Points above the red line show the model is conservative for that case.

Figure 5.50: Ratio of test (blue) or FE predictions (green) to resistances predicted by Equation 5.8.9 for members without shear reinforcement

As can be reasonably expected, Figure 5.50 shows that the lowest level of approximation gives conservative results in all cases. The scatter of cases is relatively
high compared to the more advanced levels of approximation, which again is to be expected. The model includes no specific terms to account for the shear connection, which means the spread of predictions for the FE cases is high; this is especially evident in Group 5 i.e. \( a/d = 2.0 \).

**Level of approximation (LoA) II**

The resistance for panels according to level of approximation II is given by Equation 5.8.14:

\[
V_{Rd} = V_{Rd,s} \leq V_{Rd,max}
\]  
(5.8.14)

Where:

\[
V_{Rd,s} = \frac{A_{sw}}{S_w} z f_{yw} \cot \theta
\]  
(5.8.15)

Where \( \theta \) is the angle of inclined stress field. \( \theta \) may be set anywhere between \( \theta_{min} \) and \( 45^\circ \). \( \theta_{min} \) is calculated as:

\[
\theta_{min} = 20^\circ + 10000 \varepsilon_x
\]  
(5.8.16)

\( \varepsilon_x \) is determined by Equation 5.8.17, in the same way as required for the unreinforced model.

\[
\varepsilon_x = \frac{1}{2E_s A_s} \left( \frac{M_{Ed}}{0.9d} + V_{Ed} \right)
\]  
(5.8.17)

As before, iteration is required to produce a design resistance, which for the cases in the database is achieved by setting a value of \( \varepsilon_x \) such that \( V_{Rd,c} = V_{Ed} \).

\( V_{Rd,max} \) is calculated according to Equation 5.8.11, except:

\[
k_\varepsilon = \frac{1}{1.2 + 55 \varepsilon_1} \leq 0.65
\]  
(5.8.18)

Where:

\[
\varepsilon_1 = \varepsilon_x + (\varepsilon_x + 0.002) \cot^2 \theta
\]  
(5.8.19)

Figure 5.51 shows a comparison between the resistances predicted by the model and test results. Points above the red line show the model is conservative for that case.
The results shown in Figure 5.51 show that there is little difference between the resistances predicted by LOA I, as shown in Figure 5.50. This result can be understood by comparing Figure 5.49 against Figure 5.5, which shows that the level II prediction is not much higher than the level I prediction in the region where the test results are concentrated. If the database included more tests with high ratios of shear reinforcement it is likely that more pronounced differences would be observed.

The predictions for the cases in the test database are almost identical to those predicted by LOA I. However, the level II approximation has improved the prediction of the FE results. This can be attributed to the greater precision in the consideration of strain, through Equation 5.8.17, which has a more significant effect for those cases with a low degree of shear. This is also a feature of LOA III, as presented below.

**Level of approximation (LoA) III**

The resistance for panels according to level of approximation III is given by Equation 5.8.20:
\[ V_{Rd} = V_{Rd,s} + V_{Rd,c} \]  \hspace{1cm} (5.8.20)

The level III approximation is valid when \( V_{Rd} \) is less than \( V_{Rd,max}(\theta_{min}) \). For cases outside of this limit, the resistance is determined using the level II approximation model.

The steel resistance component \( (V_{Rd,s}) \) is again given by Equation 5.8.15. \( \theta \) may be set anywhere between \( \theta_{min} \) and \( 45^\circ \).

The concrete resistance component \( (V_{Rd,c}) \) is given by:

\[ V_{Rd,c} = k_v \frac{\min \left\{ f_{ck}^{(1/2)} : 8 \right\}}{\gamma_c b_w z} \]  \hspace{1cm} (5.8.21)

Where:

\[ k_v = \frac{0.4}{1 + 1500 \varepsilon_x} \left( 1 - \frac{V_{Ed}}{V_{Rd,max}(\theta_{min})} \right) \geq 0 \]  \hspace{1cm} (5.8.22)

Where \( \varepsilon_x \) is calculated in accordance with Equation 5.8.17. \( V_{Rd,max}(\theta_{min}) \) is set by Equation 5.8.11, with \( \theta \) taken as \( \theta_{min} \).

Figure 5.52 shows a comparison between the resistances predicted by the model and test results. Points above the red line show the model is conservative for that case.
CHAPTER 5: OUT-OF-PLANE SHEAR RESISTANCE

It can be seen that the predictions produced by the level III approximation are considerably closer to measured results for the tests. The scatter of the results is low, across all $a/d$ ratios. Two of the tests from the series by Koukkari and Fülöp\cite{98} are slightly over-predicted, though this can be attributed to the fact that these cases are close to bending failure. This results in high values of $\varepsilon_x$, as determined by Equation 5.8.17. $\varepsilon_x$ is limited to 0.001 for LOA I, but no such limit applies to LOA’s II and III. The degree of over-prediction is small for these cases, so no specific adjustment is required for the model.

Figure 5.52 shows that for several of the cases the predicted resistance is less than the resistance predicted by the FE model. As with the unreinforced cases, the worst predictions are for those cases with the lowest degree of shear connection. The reduction in shear connection can be accounted for in the calculation of $\varepsilon_x$, through the use of the reinforcement percentage reduction adjustment presented in Equation 5.7.19, as opposed to Equation 5.8.17. Figure 5.53 shows the prediction when this adjustment is applied.
As expected, the inclusion of the degree of shear connection in the calculation of the strain has improved the consistency of the predictions. In all of the FE cases the resistance prediction is within 10% of the resistance measured in the FE model, which is a very strong correlation for an analytical model. A number of the tests are now predicted more conservatively, though the degree of over-prediction is not large when compared to other design models in the study.

5.8.5 Comparison between models

A number of models for the shear resistance of reinforced SCS panels have been tested against results from testing and FE analysis. Many of the conclusions that can be drawn are the same as those that can be drawn from unreinforced panels i.e. the American model produces similar values to the Eurocode, while the Japanese and South Korean approaches tend to be unconservative in many cases. However, direct comparison is difficult, because these codes are allowable stress codes, as opposed to the limit-state codes used in the USA and Europe. The fib Model Code 2010[61] again appears
to provide the best resistance predictions, which can be attributed to its more precise modelling of the strains induced in the critical cross-section.

Degree of shear connection does not appear to effect the results of the reinforced predictions to the same extent as the unreinforced predictions. Only the fib Model Code 2010 LOA III model needs specific adjustment for the degree of shear connection. Such a conclusion might be expected, since the FE study showed that the effect of changes in degree of shear connection on the resistance of reinforced panels is low (see Section 5.6.5).

5.9 Conclusion

This chapter has explored the resistance of SCS panels to out-of-plane shear forces. The existing Eurocode based design model has been evaluated against tests from the database described in Chapter 3, and has been shown to give acceptable results in the majority of cases. Consistent with conventional reinforced concrete, the model is found to be conservative when the load is placed close to the support (i.e. low $a/d$ ratios), due to the formation of substantial arching action.

Evidence for tests with low degree of shear connection was found to be lacking from the test database. In order to overcome this limitation a relatively large parametric finite element study was conducted. The FE model was verified against a large number of tests from the test database, and showed good consistency with all cases. It can be seen that degree of shear connection does have an effect on shear resistance, with the effects having more consequence for unreinforced panel design. The changes are explored in the context of the expected changes to the mechanical behaviour of each of the numerous shear resisting mechanisms. A simplified adjustment to the Eurocode model, based on a reduced effective reinforcement ratio, is presented that accounts for the cases with low degrees of shear connection (See Equation 5.5.1).

Finally, a number of models for shear resistance of SCS panels from around the world were tested against the database and the FE results. The comparisons appear to suggest that models from the Japanese (JEAC-4618[87]) and the South Korean (KEPIC-SNG[97]) codes give unconservative results in many cases, and therefore should not be used in design (though the large limitation on the allowable stress in the cross-section should ensure that designs using these codes are not unsafe). The American code (ANSIAISC N690-12[91]) tends to give similar results to the Eurocode approach.

The best predictions for both reinforced and unreinforced panels is the fib Model Code 2010[61]. The level of approximation concept implemented in this code is also
attractive for SCS panels, given that many designs are not limited by shear resistance, meaning simple but conservative models are acceptable. It is therefore recommended that this model could be used in future SCS panel design guides. A small amendment is suggested to account for lower degrees of shear connection, which results in accurate approximations for the cases included in the FE study.
Accurate prediction of deflection is essential for good structural design. While excessive deflection is not an ultimate limit state, meaning there is no safety risk to the building occupants, excessive deformation of the structure can significantly impair its function to its users, particularly if the building contains equipment that is sensitive to small movements.

Detailed non-linear finite element analysis, as used in Chapters 3 and 5, is capable of predicting deflection with a high degree of precision. However, such analysis is not practical in day-to-day design office applications, both for the impractical computational cost and the expertise required of the analyst. Designers therefore require more practical guidance, based on either a closed form equation, or a more simple FE approach. In both cases, linear elastic analysis is preferred.

SCS panel behaviour is highly non-linear. The response of panels with similar proportions may also vary considerably, with parameters like the ratio of steel to concrete and the degree of shear connection all leading to changes in stiffness, often depending on the load to which the panel is currently subjected to. In seeking to apply linear analysis, it is inevitable that the non-linear behaviour must be simplified. This chapter explores the simplifications required to develop a model that is both accurate, and still practical for designers. A new model is presented for accounting for the non-linear behaviour that can occur from lower degrees of shear connection.

### 6.1 Methodology for comparative study

Development of an accurate model for deflection requires benchmarking against tests. As described in Chapter 3, a large database of test results has been gathered from the literature. Where possible, load-deflection curves have been extracted from the
relevant publications. Different assumptions for calculating the bending stiffness of the panels are then tested against this data set. Where deviation is found between the prediction and test, refinements to the model are made, until acceptable levels of accuracy are achieved.

Closed-form solutions exist to predict deflections for all of the tests in the test database. These solutions are based on ‘Euler-Bernoulli’ bending theory, which assumes a constant value of stiffness \((EI)\) over the span.

Table 6.1 presents a summary of the these equations for the load arrangements found in the test database.

In most cases deflection is measured at the point of load application. For cases where the load is applied unequally this may not coincide with the point of maximum deflection.

**Table 6.1**: Summary of equations for calculating critical shear forces, bending moments and deflections of cases in the test database presented in Chapter 3

<table>
<thead>
<tr>
<th>One point-load, mid-span</th>
<th>( R )</th>
<th>( V_{\text{max}} )</th>
<th>( M_{\text{max}} )</th>
<th>( \Delta_{\text{max}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \frac{P}{2} )</td>
<td>( R )</td>
<td>( \frac{PL}{4} )</td>
<td>( \frac{PL^3}{48EI_{\text{eff}}} )</td>
</tr>
</tbody>
</table>
Table 6.1: Summary of equations for calculating critical shear forces, bending moments and deflections of cases in the test database presented in Chapter 3

One point-load, offset

\[
R_1 = \frac{P(L-a)}{2L} \\
R_2 = \frac{Pa}{2L} \\
V_{max} = R_1 \\
M_{max} = \frac{Pa(L-a)}{L} \\
\Delta_{(x \leq a)} = \frac{P(L-a)x}{6LEI_{eff}}(L^2 - (L-a)^2 - x^2) \\
\Delta_{(x > a)} = \frac{Pa(L-a)}{6LEI_{eff}}(2Lx - x^2 - a^2) \\
\Delta_{Load} = \frac{Pa^2(L-a)^2}{3LEI_{eff}}
\]

Two point-loads, equal offset and magnitude

\[
R = \frac{P}{2} \\
V_{max} = R \\
M_{max} = \frac{Pa}{2} \\
\Delta_{(x \leq a)} = \frac{Px}{12LEI_{eff}}(3La - 3a^2 - x^2) \\
\Delta_{(x > a)} = \frac{Px}{12LEI_{eff}}(3Lx - 3x^2 - a^2) \\
\Delta_{Load} = \frac{Pa}{12LEI_{eff}}(3La - 4a^2)
\]
Table 6.1: Summary of equations for calculating critical shear forces, bending moments and deflections of cases in the test database presented in Chapter 3

<table>
<thead>
<tr>
<th>Four point-loads, equal spacing and magnitude</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram of four point-loads" /></td>
</tr>
<tr>
<td>$R = \frac{P}{2}$</td>
</tr>
<tr>
<td>$V_{\text{max}} = R$</td>
</tr>
<tr>
<td>$M_{\text{max}} = \frac{3PL}{20}$</td>
</tr>
<tr>
<td>$\Delta_{\text{max}} = \frac{3PL^3}{160EI_{\text{eff}}}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Three supports, two equal point loads</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram of three supports, two equal point loads" /></td>
</tr>
<tr>
<td>$R_1, R_3 = \frac{5P}{16}$</td>
</tr>
<tr>
<td>$R_2 = \frac{22P}{16}$</td>
</tr>
<tr>
<td>$V_{\text{max}} = \frac{11P}{16}$</td>
</tr>
<tr>
<td>$M_{\text{max}, \text{sag}} = \frac{6PL}{32}$</td>
</tr>
<tr>
<td>$M_{\text{max}, \text{hog}} = \frac{5PL}{32}$</td>
</tr>
<tr>
<td>$\Delta_{\text{load}} = \frac{7PL^3}{768EI_{\text{eff}}}$</td>
</tr>
</tbody>
</table>
**Table 6.1**: Summary of equations for calculating critical shear forces, bending moments and deflections of cases in the test database presented in Chapter 3

<table>
<thead>
<tr>
<th>Case Description</th>
<th>Equations</th>
</tr>
</thead>
</table>
| **Beam with two cantilevered ends** | \[ R = \frac{P}{2} \]  
\[ V_{\text{max}} = \frac{P}{3} \]  
\[ M_{\text{max, hog}} = \frac{PL}{30} \]  
\[ M_{\text{max, sag}} = \frac{PL}{30} \]  
\[ \Delta_{\text{max}} = \frac{PL^3}{240EI_{\text{eff}}} \] |
| **Beam with one cantilevered end** | \[ R_1 = \frac{P}{3} \]  
\[ R_2 = \frac{2P}{3} \]  
\[ V_{\text{max}} = \frac{P}{3} \]  
\[ M_{\text{max, hog}} = \frac{PL}{12} \]  
\[ M_{\text{max, sag}} = \frac{PL}{12} \]  
\[ \Delta_{(L/4 \leq x \leq L/2)} = \frac{P}{24LEI_{\text{eff}}} \left( L^2 + 2Lx^2 + 3Lx + 4x^3 \right) \] |
6.2 Models for panel out-of-plane stiffness

This section presents the three prevailing methodologies for calculating bending stiffness, and hence deflection.

Models for out-of-plane stiffness are presented in the Bi-steel manual\textsuperscript{[24]}, the American code\textsuperscript{[182]}, the Japanese code\textsuperscript{[87]} and the South Korean code\textsuperscript{[97]}. In all of the models, the steel plates are assumed to be fully effective, meaning the second moment of area of both plates is added to the final assumed value. The contribution of the concrete varies according to the assumptions made in the model.

The first model assumes that the concrete is un-cracked, i.e. the stiffness of the concrete contributes to the stiffness of the structure in both tension and compression. The second moment of area of the concrete core is added to the second moment of area of the steel plates, to give the final value. This model is recommended by both JEAC\textsuperscript{[87]} and KEPIC-SNG\textsuperscript{[97]}. Section 5.4.5 of the Bi-steel manual\textsuperscript{[24]} also recommends the use of an un-cracked cross-section, though it also recognises that this model gives results that are too stiff. This model can be considered an upper-bound on the potential stiffness.

The stiffness of a composite panel with full concrete contribution can be calculated in accordance with Equation 6.2.1:

\[
EI_{eff, FullConc} = E_s \left( \frac{t_c^3}{12n} + \left\{ \frac{t_b^3}{12} + \frac{t_b}{2} \left( \frac{d - t_b}{2} \right)^2 \right\} + \left\{ \frac{t_{1}^3}{12} + \frac{t_l}{2} \left( \frac{d - t_l}{2} \right)^2 \right\} \right)
\]  

(6.2.1)

Where:

- \(E_s\) is the elastic modulus of the steel plates
- \(E_c\) is the elastic modulus of the concrete
- \(n\) is the modular ratio of steel compared to concrete - Taken as 13 for long term

The second model included in the study represents a lower-bound stiffness model, in that no concrete contribution is included. This approach is not recommended by any design guide, but is useful as a lower bound. The stiffness is calculated in accordance with Equation 6.2.2:

\[
EI_{eff, NoConc} = E_s \left( \left\{ \frac{t_b^3}{12} + \frac{t_b}{2} \left( \frac{d - t_b}{2} \right)^2 \right\} + \left\{ \frac{t_{1}^3}{12} + \frac{t_l}{2} \left( \frac{d - t_l}{2} \right)^2 \right\} \right)
\]  

(6.2.2)
The final model is given by AISC N690 Appendix N9\(^{[182]}\), and includes a partial concrete contribution. This model assumes that cracking of the concrete occurs, but the concrete core still provides a contribution to the stiffness. Appendix N9 refers to work by a number of publications by Varma\(^{[22,183,186]}\).

The stiffness is presented in the form shown in Equation 6.2.3:

\[
EI_{eff,AISC} = (E_s I_s + c_2 E_c I_c) \left( 1 - \frac{\Delta T_{savg}}{150} \right) \geq E_s I_s \tag{6.2.3}
\]

Where:

- \(I_s\) is the second moment of area of the steel plates
- \(I_c\) is the second moment of area of the concrete
- \(\Delta T_{savg}\) is the average of the maximum surface temperature increases for the faceplates due to accident thermal conditions - equal to 0 in the absence of thermal effects
- \(c_2\) is a calibration constant for determining effective flexural stiffness

\[
c_2 = 0.48 \rho_{AISC} n + 0.10 \tag{6.2.4}
\]

\(\rho_{AISC}\) is the reinforcement ratio, given by \((t_b + t_t) / h\)

### 6.2.1 Comparisons between methods

Figure 6.1 shows a histogram of the values of the enhancement in stiffness offered by the two methods over the assumption of no contribution to the stiffness by the concrete, for all of the tests in the database. The enhancement for the AISC model is calculated as follows:

\[
e_{AISC} = \frac{EI_{eff,AISC}}{EI_{eff,NoConc}} \tag{6.2.5}
\]

Where:

- \(EI_{eff,AISC}\) is the effective bending stiffness of the panel given by the AISC method, calculated according to Equation 6.2.3
$E_{I,\text{eff},\text{NoConc}}$ is the effective bending stiffness of the panel assuming no concrete contribution, calculated according to Equation 6.2.2.

The enhancement for the full concrete contribution is calculated as follows:

$$e_{\text{FullConc}} = \frac{E_{I,\text{eff},\text{FullConc}}}{E_{I,\text{eff},\text{NoConc}}}$$  \hspace{1cm} (6.2.6)

Where:

$E_{I,\text{eff},\text{FullConc}}$ is the effective bending stiffness of the panel assuming full concrete contribution, calculated according to Equation 6.2.1.

Equation 6.2.5 and Equation 6.2.6 are calculated for each of the tests in the sample, as described in Table 6.1. The results are then grouped into bands covering 10%. As an example, a value of 1.27 from Equation 6.2.5 is included in the group ‘120% to 130%’.

Figure 6.1: Histogram of concrete stiffness enhancement factors

Figure 6.1 shows that a large increase in stiffness can be gained if the section is assumed to be un-cracked rather than cracked. The concrete contribution depends on the relative proportions of the section, but a median contribution of around 50% is observed. Some
deep panels show a calculated stiffness that is over twice the stiffness calculated when no concrete contribution is assumed.

The AISC model gives a concrete contribution of between 10% and 30%. This is much lower than the contribution obtained by assuming the upper-bound model of a full concrete contribution. This observation suggests that the authors of the AISC method have concluded that the upper-bound assumption is un-conservative, and have therefore sought to quantify the contribution of the compression stress block only.

### 6.3 Results

The predictions of the three models for effective properties described above are compared with the test results. The full and zero concrete contribution models provide upper and lower bound stiffness’s respectively. The AISC model produces results closer to those of the zero contribution model, as explained in Section 6.2.1, and shown in Figure 6.1.

The tests conducted by Koukkari and Fülöp[^98^] can be considered some of the most simple tests in the database to predict, given they are dominated by bending and have relatively high degrees of shear connection.

For test SP1, the model for zero contribution produces the closest prediction. The AISC model also produces reasonable results, though deviation away from the test curve becomes greater at higher load levels. The full concrete contribution model shows a stiffer response than the measured results. Figure 6.2 shows a typical plot from this test series.
CHAPTER 6: PREDICTION OF DEFLECTION

Where: Test measurement

Predicted resistance

Model 1
EI = Equation 6.2.1

Model 2
EI = Equation 6.2.2

Model 3
EI = Equation 6.2.3

Final models from Chapters 4 & 5

Figure 6.2: Comparison of effective bending stiffness models against a test with a stiff shear connection and a bending dominated response (Test SP1 by Koukkari and Fülöp\cite{98}, $a/d = 4.5$, $\mu_b = 194\%$)

The tests by Koukkari and Fülöp\cite{98} are well predicted, since the load-deflection response of these tests tends to be linear up to failure. This is largely due to the relatively stiff shear connection between the steel plates and the concrete ($195\%$ on the bottom plate), which ensures that the assumption that plane sections remain plane during bending is realised. These tests also included relatively large plates in comparison to other tests in the database, ensuring that the behaviour is controlled by the stress-strain response of the steel, which is linear up to failure.

Correlation between the stiffness models and the tests is not strong in many of the other tests in the database. Systematic examination of the results has identified a number of factors that may cause deviation, as discussed below.
6.3.1 Shear deformation dominated tests

A key assumption of the effective stiffness models presented in Section 6.2 is that the out-of-plane deformation is entirely a result of bending. In reality this is unlikely to be true for SCS panels. Shear stiffness has only a marginal effect on the tests with larger span to depth ratios (say $a/d \geq 3.5$), but for smaller spans ($a/d \leq 3.5$) bending deformation will be smaller, meaning the percentage contribution of shear deformation to overall deformation is much higher.

Figure 6.3 shows the results for a typical short span test. In this test the shear span is 1.5m.

![Figure 6.3: Predictions of various bending stiffness models against a test with a relatively short shear span (Test SP1-1 by Varma et al.\cite{Varma184}, $a/d = 3.2$, $\mu_b = 113\%$)](image)

The problem of the engineer’s bending formula under-predicting shear deformation is a well known problem. Timoshenko\cite{Timoshenko1922} proposed an extension to ordinary beam theory in 1922, where an additional degree of rotational degree of freedom is
introduced to allow for shear deformation.

The model presented by Timoshenko is difficult to implement in closed form solutions, and would generally be considered beyond the use of most practising engineers. However, the principles are implemented in many finite elements. Section 6.4 describes how finite element analysis is applied to deflection prediction, and can result in improved accuracy.

6.3.2 Reduced shear connection stiffness

Reduced shear connection stiffness has an effect on the stiffness of the panels. This is manifested as a deviation of the load-deflection curve from the initial stiffness as the load is increased (see Figure 6.4). This is reflective of the curved load-slip relationship of individual shear connectors (as discussed in detail in Section 4.6).

![Figure 6.4: Predictions of various bending stiffness models against a test with a relatively low degree of shear connection (Test SP1-2 by Varma et al.\cite{184}, \(a/d = 3.2, \mu_b = 56\%\)](image-url)

Where:
- **Test measurement**
- **Predicted resistance**
- **Model 1** \(EI = \text{Equation 6.2.1}\)
- **Model 2** \(EI = \text{Equation 6.2.2}\)
- **Model 3** \(EI = \text{Equation 6.2.3}\)

236
Correlating the degree of shear connection with loss of stiffness is difficult, because there are a number of situations that can produce different behaviours. In particular, where loads are applied very close to supports, the calculation method mobilises very few studs. In reality some of the studs that are ignored by design equations will provide some contribution through residual tensile strength, thereby resulting in a stiffer response than predicted.

Since the load-deformation curve of the individual shear connectors is curved, the overall load-deformation of the panels is also curved. Fitting a linear elastic line to a curve with this degree of curvature is inherently difficult. Non-linear models offer a potential route to more accurate predictions of panels with low degrees of shear connection, although non-linear analysis has a number of disadvantages. This is discussed further in Section 6.6.

### 6.3.3 Specimens with no-tie bars

A large number of the specimens tested do not contain tie bars. In practice, tie bars (or alternative means of attaching the plates together like channel sections) are required for handling of the SC units and for the construction stage, so panels without tie-bars are rare in modern tests. Despite this, an understanding of tests without tie-bars is still useful for cases where the tie-bars are relatively far apart.

The load-deflection response of panels with no tie-bars is considerably different from those including tie bars. They exhibit a curved load-deflection relationship, with deviation away from the initial stiffness as the load increases (See Figure 6.5). This curved response cannot be linked to the shear connection, as the degree of shear connection is relatively high ($\mu_b = 148\%$).

This deviation is likely to be at least in part due to delamination stresses in the concrete above the shear studs which, while not causing failure in many cases, are a considerable source of deformation.
Specimens with long studs (taller than half the height of the section), such as those tested by Foundoukos et al. [64], act as reinforcement against this effect, and hence these tests show a linear response until much greater load levels are reached.

Figure 6.6 shows comparisons where the sample containing long studs rather than tie-bars. The presence of long studs would appear to provide considerable stiffness. It can be seen that the response in Figure 6.6 closely resembles the response in Figure 6.2.


6.3.4 Conclusion

The study described above has compared deflection predictions from closed formed solutions (presented in Table 6.1) against each of the tests in the large database where full load-deflection curves were available. The panel stiffness $EI_{eff}$ was calculated using three different models, with including a different level of contribution from the concrete.

For cases with stiff shear connections and relatively long spans, such as the one shown in Figure 6.2, a linear approximation for deflection works well. The results show very clearly that use of the upper-bound full concrete contribution is inappropriate, since the deflections predicted by this model tend to be less than the deflections observed in the tests. The AISC model also appears to predict results that are too stiff in many cases, though the correlation is closer.
For this reason, the final recommendation of this work is that deflection predictions are made assuming that there is no contribution to the stiffness of the panel from the concrete.

The results also suggest that designs where shear deformation dominates over bending deformation, such as the one shown in Figure 6.4, tend to be badly predicted. Where it is important to predict deflections accurately, it is recommended that more advanced techniques are employed, based on finite element analysis. The application of linear FE to the prediction of deflections is explored further in Section 6.4.

### 6.4 Deflection analysis using the finite element method

As discussed above, deflection analysis of individual members of a structure is possible using closed-form expressions. For cases with stiff shear connections and long spans ($a/d \geq 3.5$), these equations can be reasonably accurate. However, for cases where shear deformation is significant, closed formed models tend to underestimate deflection. Finite element analysis presents the designer with a method to undertake a more precise analysis.

In addition to the calculation of absolute values of deflection, finite element models are only used on a whole building basis to understand the distribution of forces and moments in a structure, in response to the design loads. The distribution of forces within a structure depends on the stiffness of each of the elements; a stiffer element will attract more force. As such, overestimating the stiffness of one element of the structure may result in the forces on this element being overestimated, while the forces on other elements may be underestimated. As such, it is essential that the FE model includes the best possible estimate of the true stiffness.

The planar nature of SCS panels means that they are typically modelled in FE using shell elements. Figure 6.7 shows an image of an FE model of a structure that might be found on a nuclear site, constructed with SCS panels.
The use of numerical methods considerably extends the complexity that a designer might reasonably allow for during the design process. Through the selection of the appropriate elements, it is possible to implement Timoshenko’s equations for shear deformation, as discussed further in Section 6.4.1. It is also possible to include material and geometrical non-linearity, as described in Section 6.6.

### 6.4.1 Comparison Study Methodology

As in the study described in Section 6.1, the predictions made by a variety of element formulations are compared against the tests database, presented in Chapter 3. Figure 6.8 shows a typical model.

---

**Figure 6.7:** Finite element model of a nuclear structure, using shell elements. Taken from report by Tuscher [179].
The size and variability of the database means that is preferable to automate the application of the models to the test results. To achieve this efficiently a parametric FE model was constructed to build models for each of the available test results. A system was developed for describing each test in a way that the model could be assembled, solved and post-processed without the intervention of the user.

Finite element modelling in this chapter was performed using ANSYS\textsuperscript{[12]}. However, the choice of software in this case is arbitrary, since the elements used are implemented in nearly all FE software, due to the simplicity of the element technology (as presented in Section 6.4.2). Researchers who implement the results found in alternative software should find the results are identical, except for small changes due to rounding or mesh refinement.

The model is loaded with increments of 20kN. Boundary conditions were such that no axial restraint is introduced into the model.

### 6.4.2 Element formulations

ANSYS includes a number of shell finite elements. However, SHELL181 is described as the best element for most general purpose analysis\textsuperscript{[14]}. SHELL181 implements
‘Mindlin-Reissner’ plate theory\cite{15}, an extension of ‘Kirchhoff-Love’ plate theory\cite{16,109}. ‘Kirchhoff-Love’ plate theory is analogous to ‘Euler-Bernoulli’ or classic beam theory, where plane sections remain plane and no account is taken of shear deformations. ‘Mindlin-Reissner’ extends this to include shear deformations.

The stiffness matrix of a ‘Kirchhoff-Love’ plate element is given by:

\[
\begin{bmatrix}
F_x \\
F_y \\
F_z \\
M_x \\
M_y \\
M_{xy}
\end{bmatrix} =
\begin{bmatrix}
C & Cv \\
Cv & C \\
D & Dv \\
Dv & D
\end{bmatrix}
\begin{bmatrix}
u_x \\
u_y \\
u_z \\
k_x \\
k_y \\
k_{xy}
\end{bmatrix}
\]

Where:

- \(F\) is an axial or in-plane shear force
- \(M\) is a moment
- \(u\) is a displacement
- \(k\) is a curvature
- \(v\) is the effective Poisson’s ratio of the element
- \(C\) is the axial rigidity of the section

\[
C = \frac{Et}{1 - v^2} t
\]  

\[
D = \frac{Et^2}{12 (1 - v^2)} t
\]

- \(E\) is the effective elastic modulus of the element
- \(t\) is the effective thickness of the element

A ‘Kirchhoff-Love’ plate element, such as SHELL181, includes shear deformations through two additional degrees of freedom that are added to the ‘Kirchhoff-Love’ stiffness matrix, hence:
WHERE:

\[
\begin{bmatrix}
F_x \\
F_y \\
F_{xy} \\
M_x \\
M_y \\
M_{xy} \\
S_1 \\
S_2
\end{bmatrix} = \begin{bmatrix}
C & Cv \\
Cv & C \\
C \left\{ \frac{1-v}{2} \right\} & D & Dv \\
Dv & D & D \left\{ \frac{1-v}{2} \right\} & G \\
{G} & \gamma_1 \\
\gamma_2
\end{bmatrix} \begin{bmatrix}
u_x \\
u_y \\
u_{xy} \\
k_x \\
k_y \\
k_{xy} \\
\gamma_1 \\
\gamma_2
\end{bmatrix}
\]

\( (6.4.4) \)

WHERE:

- \( S \) is an out-of-plane shear force
- \( G \) is the shear modulus

\[ G = \frac{E}{2(1+v)} k_s t \]  

\( (6.4.5) \)

\( k_s \) is the shear-correction factor, typically taken as \( 5/6 \)

In the above formulation the out-of-plane shear stiffness \( G \) is taken to be a function of the elastic modulus \( E \). Models for the effective stiffness of the panel generally calculate equivalent values of \( E, t \) and \( v \), for the reasons described in Section 6.4.4. As the formulation shows, this is enough information to give a full representation of the panel stiffness both in-plane and out-of-plane. The models given in Section 6.4.4 give equations for these 3 parameters.

The Poisson’s ratio \( v \) of an SCS panel is difficult to define with certainty. The extent to which the Poisson’s effects will be manifested depend on the failure mechanism, and how the load is applied to the structure. The only explicit guidance in the AISC design manual[9] suggests the use of the Poisson’s ratio of concrete when calculating induced forces due to thermal expansion (as discussed in Section 2.6). Poisson’s ratios used by researchers vary, with some taking the value as high as 0.3[127,200]. Many suggest the use of the concrete Poisson’s ratio, without giving an explicit value. In the absence of better information, a value of 0.2 is taken for this work.

The out-of-plane shear stiffness and out-of-plane bending stiffness are not necessarily coupled in the element stiffness matrix. It is only through the calculation given in
Equation 6.4.5 that the two stiffnesses are related. It is possible to decouple them by using alternative formulations for $G$, such as the one given in Equation 6.5.4. This is done with good reason, as the values implied by 6.4.5 can vary considerably from reasonable values; this is explored further in Section 6.5.

### 6.4.3 Input of stiffness matrix values into ANSYS

ANSYS allows the direct input of the stiffness matrix for a classical shell element. The example below assumes the following values for the effective properties:

\[ E = 40000 \text{ N/mm}^2 \]
\[ t = 450 \text{ mm} \]
\[ v = 0.2 \]

Assuming SI units, the coefficients are calculated as follows:

\[ C = \frac{40000000000 \times 0.45}{1 - 0.2^2} = 1.88 \times 10^{10} \text{ N/mm} \]
\[ D = \frac{40000000000 \times 0.45^3}{12(1 - 0.2^2)} = 3.16 \times 10^8 \text{ Nmm} \]
\[ G = \frac{40000000000}{2(1 + 0.2)} \times \frac{5}{8} \times 0.45 = 6.25 \times 10^9 \text{ N/mm} \]

These values are a direct input into the stiffness matrix using the following commands:

\[
\begin{array}{l}
C = 18750000000 \\
D = 316406250 \\
G = 6250000000 \\
v = 0.2 \\
\text{SECTYPE, 1, GENS} \\
\text{SSPA, C, C*v, 0, C, 0, C*(1-v)/2} \\
! Specifies a preintegrated membrane stiffness for shell sections – 6 components \\
\text{SSPB, 0, 0, 0, 0, 0, 0} \\
! Specifies a preintegrated coupling stiffness for shell sections – 6 components \\
\text{SSPD, D, D*v, 0, D, 0, D*(1-v)/2} \\
! Specifies a preintegrated bending stiffness for shell sections – 6 components
\end{array}
\]
CHAPTER 6: PREDICTION OF DEFLECTION

SSPE,G,0,G
! Specifies a preintegrated transverse shear stiffness for shell sections – 3 components
SSPM,1
! Specifies mass density for a preintegrated shell section.

Alternatively, a single layer shell element can accept values of $E$ and $t$:

! Young’s Modulus
MP,EX,matNo,40000000000
! Poisson’s ratio
MP,PXY,matNo,0.2
SECTYPE,sectNo,SHELL
! Thickness (with 3 additional properties)
SECDATA,0.45,5,0,5
SECOFFSET,MID
SECCONTROL,0,0,0,0,1,1,1

The results of the ANSYS integration are shown below:

LIST SECTION ID SETS 1 TO 1 BY 1
Details = FULL

SECTION ID NUMBER: 1
SHELL SECTION TYPE:
SHELL SECTION NAME IS:
SHELL SECTION DATA SUMMARY:
Number of Layers = 1
Total Thickness = 0.450000

<table>
<thead>
<tr>
<th>Layer</th>
<th>Thickness</th>
<th>MatID</th>
<th>Ori. Angle</th>
<th>Num Intg. Pts</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.4500</td>
<td>1</td>
<td>0.0000</td>
<td>5</td>
</tr>
</tbody>
</table>
### SECTION Membrane & Bending (ABD) Matrix

**Top left**

\[
\begin{bmatrix}
0.187500E+11 & 0.375000E+10 & 0.00000 \\
0.375000E+10 & 0.187500E+11 & 0.00000 \\
0.00000 & 0.00000 & 0.750000E+10
\end{bmatrix}
\]

**Top right**

\[
\begin{bmatrix}
-0.292576E-07 & 0.214176E-07 & 0.00000 \\
0.214176E-07 & -0.292576E-07 & 0.00000 \\
0.00000 & 0.00000 & 0.160131E-07
\end{bmatrix}
\]

**Bottom left**

\[
\begin{bmatrix}
-0.292576E-07 & 0.214176E-07 & 0.00000 \\
0.214176E-07 & -0.292576E-07 & 0.00000 \\
0.00000 & 0.00000 & 0.160131E-07
\end{bmatrix}
\]

**Bottom right**

\[
\begin{bmatrix}
0.316406E+09 & 0.632812E+08 & 0.00000 \\
0.632812E+08 & 0.316406E+09 & 0.00000 \\
0.00000 & 0.00000 & 0.126562E+09
\end{bmatrix}
\]

### SECTION Transverse Shear Correction Factors

\[
\begin{bmatrix}
0.833333 & 0.00000 \\
0.00000 & 0.833333
\end{bmatrix}
\]

### SECTION Transverse Shear Stiffness (E)

\[
\begin{bmatrix}
0.750000E+10 & 0.00000 \\
0.00000 & 0.750000E+10
\end{bmatrix}
\]

Shell Section is offset to MID surface of Shell

Section Solution Controls

User Transverse Shear Stiffness (11) = 0.0000
It can be seen that the results of the ANSYS integration are exactly the same as those manually input. Two small differences can be observed:

1. The ‘Transverse Shear Correction Factor’, $k_s$, is kept separate from the ‘Transverse Shear Stiffness’. Multiplying both parameters together gives the value of $G$ calculated.

2. A number of small terms are included in ‘coupling’ matrices in the top right and bottom left. These coefficients are negligible, and can therefore be ignored.

The simplicity of the alternative method makes it attractive to designers. However, use of this alternative method makes decoupling of the out-of-plane bending stiffness from the out-of-plane shear stiffness more difficult. This is important, as discussed in Section 6.4. The out-of-plane shear stiffness can be decoupled by overriding the Transverse Shear Correction Factors, but it is suggested that this is more easily achieved using the original input method, and is therefore not recommended.

### 6.4.4 Translation of stiffness into an effective height and elastic modulus

A key assumption of the closed form solutions presented in Table 6.1, and the element formulations presented in Section 6.4.2, is that the material throughout the cross-section is homogeneous, with a fixed elastic modulus ($E$). The shape of the section, which determines the efficiency of the cross-section in resisting bending stresses, is described by the second moment of area ($I$). These two quantities multiplied together give the stiffness of the cross section ($EI_{\text{panel}}$).

Most software packages compel the user to enter the elastic modulus and the second moment of area separately, since this is most logical format for users; the user must enter two parameters, rather than the 18 or more required to directly enter a stiffness matrix. Most software packages also contain code to allow for the calculation of

\[
\begin{align*}
(22) &= 0.0000 \\
(12) &= 0.0000 \\
\text{Added Mass Per Unit Area} &= 0.0000 \\
\text{Hourglass Scale Factor; Membrane} &= 1.0000 \\
\text{Bending} &= 1.0000 \\
\text{Drill Stiffness Scale Factor} &= 1.0000
\end{align*}
\]
second moment of area from a given cross-section shape. Assuming a rectangular cross-section, it therefore becomes typical to represent the stiffness of a cross-section using both an effective modulus $E_e$ and an effective cross-section height $h_e$. The relationship between the two properties and the panel stiffness is shown in Equation 6.4.6.

$$EI_{\text{panel}} = E_e I_e = E_e \left( \frac{bh_e^3}{12} \right)$$  \hspace{1cm} (6.4.6)

Equation 6.4.6 contains two unknown parameters. It is therefore possible to adopt any two values of $E_e$ and $h_e$, as long as the product of the two equals $EI_{\text{panel}}$. However, SCS panels usually carry in-plane forces in addition to out-of-plane forces, which means that the in-plane stiffness must also be correct. The in-plane relationship is presented in Equation 6.4.7.

$$EA_{\text{panel}} = E_e A_e = E_e (bh_e)$$  \hspace{1cm} (6.4.7)

Equation 6.4.6 contains the same two unknowns as Equation 6.4.7. These equations may therefore be solved simultaneously, to give fixed values of $E_e$ and $h_e$. Solutions are presented for each of the models for effective stiffness (see Equations 6.4.9 and 6.4.12 for examples).

While the particular values of $E_e$ and $h_e$ are irrelevant to the bending stiffness, these values become important when shear rigidity is introduced into the analysis, through the use of Timoshenko elements (as presented in Section 6.4.2). In classical mechanics, shear stiffness is also a cross-sectional property, which may be derived from the cross-section shape and its elastic modulus. The relationship is shown in Equation 6.4.8:

$$GA_{\text{panel}} = G_e A_e = \frac{E_e}{2(1 + \nu_e)} bh_e$$  \hspace{1cm} (6.4.8)

Where:

$$\nu_e$$  \hspace{1cm} is the effective Poisson’s ratio, taken as 0.2

Unless the user specifically overrides the shear stiffness, the values calculated by the simultaneous solution of Equations 6.4.6 and 6.4.7 can lead to values that vary considerably from values that might be calculated using a more considered model. This is discussed further in Section 6.5.
If **full concrete contribution** is required for the model, substituting, expanding and rearranging Equations 6.4.6 and 6.4.7 leads to the following expressions for the effective elastic modulus:

\[
E_e = \frac{E_s \left( \frac{t^3_c}{12n} + \left\{ \frac{t^3_c}{12} + t_b \left( \frac{d-t_b}{2} \right)^2 \right\} + \left\{ \frac{t^3_c}{12} + t_t \left( \frac{d-t_t}{2} \right)^2 \right\} \right)}{h_e^3} \tag{6.4.9}
\]

\[
E_e = \frac{E_s (t_c/n + t_b + t_t)}{h_e} \tag{6.4.10}
\]

Where:

- \(E_e\) is the effective elastic modulus of the equivalent element
- \(E_s\) is the elastic modulus of steel
- \(I_e\) is the effective second moment of area of the equivalent element
- \(A_e\) is the effective area of the equivalent element
- \(EI_{\text{panel}}\) is the bending stiffness of the panel
- \(EA_{\text{panel}}\) is the axial stiffness of the panel
- \(n\) is the modular ratio of steel compared to concrete - Taken as 13 for long term
- \(t_c\) is the thickness of the concrete layer

\[
t_c = h - t_b - t_t \tag{6.4.11}
\]

Equations 6.4.9 and 6.4.10 can be solved for \(E_e\) and \(h_e\). By equating equations 6.4.9 and 6.4.10 and rearranging, the following expression is obtained for the equivalent height of the element:

\[
h_e = \sqrt{\frac{n \left( 4t^3_b - 6t^2_b d + 3t_b d^2 \right) + n \left( 4t^3_t - 6t^2_t d + 3t_t d^2 \right) + t^3_c}{\sqrt{n (t_b + t_t) + t_c}}} \tag{6.4.12}
\]

\(E_e\) is then calculated by substituting for \(h_e\) in either Equation 6.4.9 and 6.4.10.

If the **zero concrete contribution** model is required, \(t_c\) can be assumed equal to 0 in Equations 6.4.9 and 6.4.10. This leads to the following expressions:

\[
E_e = \frac{E_s \left( \left\{ \frac{t^3_c}{12} + t_b \left( \frac{d-t_b}{2} \right)^2 \right\} + \left\{ \frac{t^3_c}{12} + t_t \left( \frac{d-t_t}{2} \right)^2 \right\} \right)}{h_e^3} \tag{6.4.13}
\]
The effective height is calculated using Equation 6.4.14:

\[ h_e = \frac{\sqrt{n \left( 4t_b^3 - 6t_b^2d + 3t_b d^2 \right) + n \left( 4t_t^3 - 6t_t^2d + 3t_t d^2 \right)}}{\sqrt{n \left( t_b + t_t \right)}} \] (6.4.14)

The stiffness of the element is then entirely contributed by the plates acting at a lever arm of \( d - \left( \frac{t_b}{2} + \frac{t_t}{2} \right) \) (i.e. assuming plane sections remain plane).

The effective height and the effective modulus of elasticity for the AISC model must be found such that the correct in-plane stiffness is also given. The values determined are therefore dependent on whether the section is cracked or un-cracked in in-plane shear.

For a section that is un-cracked in in-plane shear:

\[ h_e = \left( \frac{(G_s A_s + G_c A_c)}{E_{eff}} \right) \] (6.4.15)

Where:

- \( G_s \) is the shear modulus of the steel plates
- \( G_c \) is the shear modulus of concrete
- \( A_s \) is the area of the steel per unit width
- \( A_c \) is the area of the concrete per unit width

Equation 6.4.16 can be rearranged into the following form:

\[ E_{eff} = \frac{(E_s I_s + c_2 E_c I_c)}{\left( \frac{h_e}{12} \right)} \] (6.4.16)

\( E_{eff} \) and \( h_e \) appear in both equations, hence they must be solved simultaneously. This can be achieved analytically or through trial and improvement.

### 6.5 Models for out-of-plane shear stiffness

The shear stiffness of concrete structures is difficult to model, since small changes in the design parameters can lead to significant changes in the response. Key parameters include the concrete tensile strength, the percentage of flexural and shear reinforcement and the distance from the point of load application to the support. SC structures have a number of additional features, such as the plate to concrete shear connection stiffness,
which affects the overall stiffness.

Shear stiffness is not currently explicitly covered by any of the models described in Section 6.2. Instead it is a by-product of the effective properties calculated for bending.

A relatively simple model for shear stiffness is examined. It can be assumed that all of the shear force in the section is resisted by the concrete with no contribution from the steel plates. The shear stiffness is then given by:

\[
G_{t_{\text{eff}}} = G_{t_{\text{conc}}} = \frac{E_c}{2(1 + \nu_c)}t_c
\]  

(6.5.1)

Where:

- \(E_c\) is the elastic modulus of the concrete
- \(t_c\) is the thickness of the concrete layer
- \(\nu_c\) is the Poisson’s ratio of concrete, taken as 0.2

This assumption reflects the fact that when shear dominates, the plates and concrete will separate between the studs. The plates then contribute little to the shear stiffness, since they are acting about their weak axis.

It should be noted that additional adjustment is often applied to this parameter in analysis of conventional reinforced concrete where torsional stiffness is important. Research shows that this parameter tends to overestimate torsional stiffness. The topic is discussed in detail by Broo\textsuperscript{[33]}. This topic is not explored in this thesis, since applications where torsion is significant are not recommended by the design guides; special discussion is presented in the Bi-Steel Manual\textsuperscript{[24]}.

### 6.5.1 Comparisons of shear stiffness predictions with shear stiffness of the concrete layer

As shown in Figure 6.1, the AISC effective properties for bending include a small contribution from the concrete amounting to an increase in stiffness of around 15-30% above the contribution of the steel plates. However, the effective height and modulus of elasticity predictions vary to a greater degree. This is reflected in the shear stiffness that is derived as a result.

Figure 6.9 shows a histogram of the shear stiffness predicted by the stiffness formulations in Section 6.4.4, with no concrete contribution and partial concrete
contribution, normalized by dividing by the stiffness predicted by the concrete only shear stiffness model given in Equation 6.5.1. The values are calculated as follows:

\[ r_1 = \frac{G_{t,\text{eff, NoConc}}}{G_{t,\text{Conc}}} = \frac{E_{e,\text{NoConc}}}{2(1 + \nu_{e,\text{NoConc}})} \frac{h_{e,\text{NoConc}}}{G_{t,\text{Conc}}}/h_{e,\text{NoConc}} \]  
\[ (6.5.2) \]

Where:

- \( G_{t,\text{Conc}} \) is the out-of-plane shear stiffness of the concrete layer, calculated using Equation 6.5.1
- \( E_{e,\text{NoConc}} \) is the effective modulus of elasticity of the element, assuming no concrete contribution, calculated using Equation 6.4.13
- \( h_{e,\text{NoConc}} \) is the effective height of the element, assuming no concrete contribution, calculated using Equation 6.4.14
- \( \nu_{e,\text{NoConc}} \) is the effective Poisson’s ratio, taken as 0.2

\[ r_2 = \frac{G_{t,\text{eff, AISC}}}{G_{t,\text{Conc}}} = \frac{E_{e,\text{AISC}}}{2(1 + \nu_{e,\text{AISC}})} \frac{h_{e,\text{AISC}}}{G_{t,\text{Conc}}} \]  
\[ (6.5.3) \]

Where:

- \( E_{e,AISC} \) is the effective modulus of elasticity of the element, assuming no concrete contribution, calculated using Equation 6.4.16
- \( h_{e,AISC} \) is the effective height of the element, assuming no concrete contribution, calculated using Equation 6.4.15
- \( \nu_{e,AISC} \) is the effective Poisson’s ratio, taken as 0.2

Values using equations 6.5.2 and 6.5.3 are calculated for each of the tests in Table 6.1. The results are then grouped into 10% intervals. The results are shown in Figure 6.9.
The results show there is a vast spread of different predictions, despite the fact the models produce similar stiffnesses for bending (i.e. within 10-30%). The model that assumes no concrete contribution in bending generally under-predicts the shear stiffness when compared with the concrete layer model of Equation 6.5.1. On the other hand, the results produced by the AISC model over-estimate the shear stiffness.

This analysis shows that just because a model has been calibrated to give reasonable predictions of the bending stiffness, it does not follow that the model will also result in reasonable values for the shear stiffness. The latter must be accounted for by using an explicit model.

### 6.5.2 Model by Kim and Mander

Attempts have been made by a number of researchers to derive shear stiffness formulations for reinforced concrete. It is recognised that flexural cracking occurs at very small loads, thus such formulations attempt to model the post cracking shear stiffness. A key conclusion of all approaches is that use of the un-cracked shear stiffness, as used in Section 6.5, leads to overly stiff results.

A relatively simple model is given in Kim and Mander\(^{[93]}\) for beams with shear reinforcement. It is based on the same effective truss concept as the Eurocode 2 model
for calculation of ultimate shear resistance. A method for calculating the angle of the compression members of the effective truss is given. However, in SC panels of typical proportions, an angle of 21.8° is obtained.

The out-of-plane shear stiffness is given by Equation 6.5.4:

\[
G_t = \frac{n\rho_v \cot^2 \theta}{1.3 + n\rho_v csc^4 \theta} \times E_c A_v
\]  

(6.5.4)

Where:

\(E_c\) is the modulus of elasticity of the concrete
\(A_v\) is the shear area of the beam. For a unit area this is taken as \(t\)
\(n\) is the modular ratio
\(\rho_v\) is the shear reinforcement ratio
\(\theta\) is the effective truss angle, taken as 21.8°

Hence, an alternative stiffness formulation can be obtained by using the above result instead of that of Equation 6.5.5 in the element stiffness matrix.

\[
r = \frac{G_{tK&M}}{G_{tConc}}
\]

(6.5.5)

Where:

\(G_{tK&M}\) is the effective out-of-plane shear stiffness, according to Equation 6.5.4
\(G_{tConc}\) is the out-of-plane shear stiffness of the concrete layer only, according to Equation 6.5.1

Figure 6.10 shows a histogram of the results of this model compared to the concrete only shear stiffness of Equation 6.5.1. The histogram is developed using Equation 6.5.5.
Figure 6.10: Predictions of shear stiffness (Gt) using the model by Kim and Mander\textsuperscript{[93]}

As the histogram shows, the shear stiffness is typically much reduced from the un-cracked prediction; nearly all values are less than 10\% of the reference model.

6.5.3 FE comparative study - Results & Validation

Figure 6.11 shows the application of the model described in Section 6.5 to test WS1.5 by Chu et al.\textsuperscript{[46]}, which failed in shear. The bending and in-plane stiffness matrix terms are calculated using Equations 6.4.16 and 6.4.15, i.e. the AISC bending stiffness model.
Where: Test measurement
Predicted resistance
Final models from Chapters 4 & 5
Model 1
Model 2

Figure 6.11: Predictions of load deflection response for two models of shear stiffness (Gt) for a shear deformation dominated test (Test WS1.5 by Chu et al.\textsuperscript{[46]}, \(a/d = 1.5, \mu_b = 85\%\))

The un-cracked model for Gt (Equation 6.5.1) shows no significant reduction from the original prediction. However, the model for Gt proposed by Kim and Mander\textsuperscript{[93]} (Equation 6.5.4) shows much better correlation. The model predicts the stiffness well at most load levels, with only the final non-linear portion of the curve showing deviation. This is acceptable, since it would not be possible for a linear elastic approximation to reproduce this portion of the curve.

Tests where the stiffness is dominated by shear response tend to show a better fit with the Kim and Mander model. Figure 6.12 shows the results of a continuous beam test by Takeuchi et al.\textsuperscript{[174]}. 

257
CHAPTER 6: PREDICTION OF DEFLECTION

Where: Test measurement
Predicted resistance
Final models from Chapters 4 & 5
Model 1
Model 2

![Figure 6.12](image)

**Figure 6.12:** Predictions of load deflection response for two models of shear stiffness (Gt) to a continuous test (Test 10 by Takeuchi et al.\(^{[174]}\), \(a/d = 2\), \(\mu_b = 44\%\))

The results of this formulation are also accurate where shear deformation is small or negligible, such as SCIENCE test SP1, which failed in bending (Figure 6.13).
Where: Test measurement
Predicted resistance
Final models from Chapters 4 & 5
Model 1
Model 2

Figure 6.13: Predictions of load deflection response for two models of shear stiffness (Gt) to a bending deformation dominated test (Test SP1 by Koukkari and Fülöp\textsuperscript{[98]}, \(a/d = 4.5, \mu_b = 194\%\))

Overall, the results of this analysis have shown that it is important to include specific provision for shear stiffness when calculating effective properties, since calculating shear stiffness as a by-product of the effective bending stiffness results in values that can vary considerably from more refined predictions.

The assumption that the concrete provides all of the shear stiffness to the section produces results that are too stiff. However, the model by Kim and Mander produces very good results in most cases, despite its simplicity. It is therefore recommended that the Kim and Mander model provides the best representation of shear stiffness for analysis of SC structures.
6.6 Non-linear finite element analysis

Comparisons presented in the previous sections have shown that linear-elastic analysis is capable of predicting the deflection of most SCS panels accurately, as long as the correct concrete contribution is used. However, this only holds true when the shear connection between the steel plates and concrete is sufficiently stiff to allow the full stiffness of the plates to be mobilised. For panels with lower degrees of shear connection, such as the one shown in Figure 6.4, deformation of the shear interface dominates the overall response.

The computational power that has become available in recent years means that non-linear analysis techniques have become viable for everyday design\[85\]. Computational power is harnessed to perform iterative calculations, allowing the stiffness of the element to change as the forces within the element develop.

Most problems in structural engineering are inherently non-linear, particularly at the onset of failure\[13\]. Non-linear analysis allows a more accurate representation of true behaviour of the structure, and is therefore preferable. However, non-linear analysis techniques also have a number of disadvantages:

1. **No principle of superposition** - Linear elastic structural analysis may employ the principle of superposition, wherein the load effects (forces, moments, deflections) generated by a particular load profile may be added together without considering any effect from loads working in combination. The load effects are also proportional to the forces, meaning the load effects can be proportionately scaled.

   It is typical to perform an analysis of the effects of the self-weight of the structure separately, after which applied loads may be added in various combinations. This principle no longer applies for non-linear analysis.

   Judicious use of the principle of superposition can considerably reduce the number of analysis that need to be performed to calculate all of the load combinations that may arise in a typical structure. For the example design performed as part of the SCIENCE project\[179\], 28 combinations were considered before rationalisation. However, only 6 FE runs were required to produce results, as the design effects could be added and scaled as required. For a non-linear analysis, all the combinations must be considered separately. For more complex nuclear structures, the number of combinations may exceed 100, as discussed by Gallitre\[70\].
2. **Calculation time** - Non-linear analysis is inherently more time consuming than an equivalent linear analysis. A linear problem is solved with one inversion of the stiffness matrix, while a non-linear problem requires incremental-iterative convergence toward a solution. Assuming an iterative solver based on the Newton-Raphson method\cite{85}, at least three solves are required; two solutions to find the deformation to out-of-balance force gradient, and one solve to check the convergence of the final solution. In practice, many more runs than three are usually required.

Given more analysis required (as discussed in point 1), and that each analysis might require considerably more solves, the computational cost of non-linear analysis is considerably more than for linear analysis.

Although computational power per pound spent continues to improve over time, availability of computer resource is still a major concern for designers.

3. **Expertise of analyst** - Introduction of additional complexity into the computation also places additional requirements on the analyst, often including management of a largely increased set of modelling and material model parameters. In particular, convergence problems can be difficult to diagnose for those who are not experienced in finite element analysis.

4. **Software cost** - The inherent need for iteration in a non-linear analysis introduces additional requirements for the software. In addition to the iteration, the program must also be capable of keeping track of convergence, decide appropriate load increments and provide diagnostics for convergence, each of which might be considered more difficult to code than the matrix decomposition. This extra development effort is reflected in the cost of the software to the designers. For more general purpose software, specific modifications may be required to make the analysis work correctly, as explored in Section 6.6.1.

Although these disadvantages are significant, non-linear analysis is justified if it allows a greatly improved level of accuracy. For tests with lower degrees of shear connection, like the one shown in Figure 6.4, it appears reasonably accuracy can only be achieved with a non-linear model. The remainder of this section explores the level of improvement that can be achieved.
6.6.1 Implementation of non-linear model in ANSYS

The majority of commercial finite element packages include some form of non-linear analysis capability. However, this non-linear capability is generally implemented through specification of non-linear material properties, which are then used to perform iterative integrations of the cross-section stresses and strains. Material based non-linearity is not suitable for modelling SCS panels, as the correct modelling of the in-plane, out-of-plane and shear stiffnesses must occur at different rates in response to the same stress state. As an example, a panel subject to a moment close to causing failure may suffer a 50% degradation of out-of-plane stiffness, while the in-plane and shear stiffnesses may degrade to a lesser or greater extent. Without the ability to define different material laws in different directions, at least one of the stiffnesses will be inaccurate.

Decoupling of the 3 stiffnesses can be achieved by direct entry of the stiffness coefficients into the program. This can be achieved in ANSYS using the process described in Section 6.4.3. However, this method is only valid for linear analysis. No capability is available to couple any of the quantities in the stiffness matrix to any other parameter, such as the extreme fibre stress or the applied moment.

After trying a number of options, it was found that there was no option to produce the correct uncoupled element coefficients in the non-linear solver. Instead, the implementation used in this work makes use of the linear solver. Shell element properties are entered into the program using the general section, as described in Section 6.4.3, based on the initial stiffness. The analysis is then run. From there, the moments extracted are used to update the element coefficients. In order to simplify the code, ten analysis runs were carried out for each case, rather than programming any convergence criteria. Once deflection exceeded 100mm the analysis was stopped, since all of the tests in the test database were judged to have failed before this deflection was measured.

6.6.2 Empirical model for loss of flexural stiffness with increased moment

Accounting for the change in stiffness resulting from deformation of the shear connection is difficult. As discussed in detail in Section 4.8, overall deformation and resistance of cases with low degrees of shear connection is heavily driven by slip of the tension plate. Tensile resistance is also provided by inherent tensile resistance in the concrete, with the formation of cracks leading to discontinuities in the tension plate slip and the overall deflection.
Characterisation of the load-deflection relationship relies on detailed knowledge of the crack locations. Detailed non-linear finite element models, such as the one discussed in Section 4.4, offer a route for developing this understanding. However, models like this are beyond the use of most ordinary designers, as the model is extremely computer resource intensive, and require specific expertise to build and operate.

An attempt was made to produce a simplified model, on the basis of the critical cross-section definition presented in Section 4.12. This model was more accurate in cases with larger structural heights and thicker plates, as the cracking on the tensile side occurs early in the loading process, leaving the critical span well defined. This was found not to be the case for smaller panels, where the critical crack for the ULS model might only appear shortly before the failure. In these cases, deflection at lower loads is governed more by the tensile resistance of the concrete, which can vary considerably from case to case.

An empirical model for stiffness reduction was developed. The model correlates stiffness change with the moment that causes slip failure of the tension plate, which will be higher than the failure moment if the degree of shear connection is greater than 100%. If failure occurs by some other means, such as in shear, stiffness will still be reduced, but not to the same extent as if failing in bending.

The stiffness reduction equation is:

\[
EI_{\text{eff,Non-Linear}} = EI_{\text{eff}} \left\{ \min \left( \mu_{\text{avg}} : 1 \right) \left( 1 - 0.4 \left( \frac{M_{\text{Ed}}}{M_{\text{Rd,\mu}}} \right) - 0.55 \left( \frac{M_{\text{Ed}}}{M_{\text{Rd,\mu}}} \right)^2 \right) \right\}
\]

(6.6.1)

Where:

- \( \mu_{\text{avg}} \) is the average degree of shear connection on the top and bottom plates
- \( M_{\text{Ed}} \) is the applied moment on the element
- \( M_{\text{Rd,\mu}} \) is the moment resistance of a section that fully utilises the shear connection

\( M_{\text{Rd,\mu}} \) is calculated as follows:

\[
M_{\text{Rd,\mu}} = M_{\text{Rd,\mu}} \mu_{\text{avg}}
\]

(6.6.2)
\( M_{Rd} \) is the moment resistance of a section, given by either Equation 4.1.1 or Equation 4.1.2

It should be noted that Equation 6.6.2 will give values of \( M_{rd, \mu} \) greater than \( M_{Rd} \) when the degree of shear connection is greater than unity. \( M_{rd, \mu} \) is a reference value, and does not have any physical significance.

### 6.6.3 Results

Figure 6.14 shows the result of applying this model to Test SP2c-2 by Varma et al.\[184\]. This test has a particularly low stiffness shear connection, and therefore suffers a considerable loss of stiffness with load, until it eventually fails in the shear connection. The new model predicts this loss of stiffness well.

![Figure 6.14: Comparison between a linear and non-linear prediction of stiffness (Test SP1-2 by Varma et al.\[184\])](image-url)

Where:
- Test measurement
- Predicted resistance
- Final models from Chapters 4 & 5
- Model 1: \( EI = \text{Equation 6.2.3} \)
- Model 2: \( EI = \text{Equation 6.6.1, with Equation 6.2.3} \)
Figure 6.15 shows the results of applying the model to Test SP2c-2 by Varma et al.\cite{184}, which has a stiffer shear connection. The prediction in this case is still reasonable, though the loss of stiffness predicted at higher moments is greater than observed in the test.

Where: Test measurement Predicted resistance Final models from Chapters 4 & 5
Model 1 EI = Equation 6.2.3
Model 2 EI = Equation 6.6.1, with Equation 6.2.3

**Figure 6.15:** Comparison between a linear and non-linear prediction of stiffness (Test SP2c-2 by Varma et al.\cite{184}, $a/d = 2, \mu_b = 78\%$)

Application of the model to cases with stiff shear connections does not result in a considerable drop in stiffness. Figure 6.16 shows the results for test SP1 by Koukkari and Fülöp\cite{98}. As with the previous case, the loss of stiffness at the higher moments is slightly over-predicted, but not to the considerable detriment of the analysis.
Where: Test measurement
Predicted resistance
Model 1
Model 2

Final models from Chapters 4 & 5
EI = Equation 6.2.3
EI = Equation 6.6.1, with Equation 6.2.3

Figure 6.16: Comparison between a linear and non-linear prediction of stiffness (Test SP1 by Koukkari and Fülöp\textsuperscript{[98]}, $a/d = 4.5, \mu_b = 194\%$)

6.6.4 Conclusions

As shown in Section 6.6.3, non-linear analysis is a possible route to improved accuracy in the prediction of deflection of SCS panels. In cases like Test SP1-2 by Varma et al.\textsuperscript{[184]}, the low degree of shear connection considerably reduces the stiffness of the panel. This loss of stiffness is captured through an empirical formula, calibrated against tests in the database. Although the accuracy of the prediction varies, the results are generally better than those predicted through linear elastic analysis.

Although non-linear analysis may offer improved accuracy, the improvement has come at considerable cost to the ease of use of the model. Non-linear analysis in general has a number of disadvantages, as explained at the start of 6.6. In addition, capturing the behaviour of SCS panels requires an element technology that is capable of uncoupled non-linear evolution of the bending and axial stiffnesses, which is not readily implemented in the majority of finite element software, including ANSYS.
Although ANSYS was used to produce the comparisons presented, this was only achieved through the use of programming techniques. The implementation presented is also aimed at reproducing the deflections recorded in the test database, which tend to be for relatively simple arrangements of beam test. Applying the same techniques to a whole building model is likely to lead to a number of difficulties, including the possibility of run-away deflections and convergence difficulties.

Given the considerable difficulty that is likely to be encountered by designers, and the large number of dis-advantages, it is not recommended that buildings are analysed using non-linear models at this time. However, results have shown that introducing non-linearity into deflection predictions can result in improved accuracy. Should this accuracy prove to be necessary, this section presents a model for including this behaviour, through the use of a non-linear moment-stiffness reduction relationship. It may be possible to develop a bespoke analysis software that is capable of analysing buildings using this concept, though this is beyond the scope of this thesis.

6.7 Overall conclusion

This chapter has examined and proposed a number of different ways of predicting the deflection of SCS panels. The key to accurate prediction of deflection is the development of an understanding of the contribution of the concrete core to the overall stiffness of the panel. To achieve this understanding, three models were assumed for the contribution, including full contribution (uncracked), no contribution, and partial contribution, as included in the AISC code for SCS panels. These models are compared against the database of test results described in Chapter 3. The following conclusions are reached:

1. Linear models perform well for tests with a high degree of shear connection that fail in bending. Results appear to suggest that concrete contribution is relatively low.

2. The AISC model, which includes some concrete contribution, gives reasonable results.

3. Many tests exhibit non-linear response from an early stage. When separated into groups, this can be attributed to:
   
   - Low degree of shear connection – Shear connection slip is highly non-linear. A lower bound limit on the degree of shear connection in design will reduce
or eliminate such behaviour.

- Shear deformation – The shear deformation of members is not taken into account in published codes. Application of a new model for shear stiffness produced good results, and is recommended for future use.

- Lack of shear reinforcement – Many of the tests included in the sample do not include tie-bars or vertical shear reinforcement. These tests are not reflective of likely construction practice, since the construction stage requires some form of tying between the plates. The lack of correlation with these tests can be ignored.

The results in this chapter suggest that shear deformation was underestimated in the analysis. It was recognised that improvements could be made to the predictions by using a finite element analysis that incorporates the classic shear deformation model presented by Timoshenko\textsuperscript{[178]}. For shell elements, use of ‘Mindlin-Reissner’ plate theory has been shown to produce better results than ‘Kirchhoff-Love’ plate theory. A model for shear stiffness presented by Kim and Mander\textsuperscript{[93]} was found to improve predictions.

Finally, non-linear analysis was investigated. An empirical formula that accounts for stiffness reduction with increased moment was developed, and was found to improve predictions in a number of cases, especially when the degree of shear connection was low. However, non-linear analysis has a significant number of disadvantages, which make it unattractive to designers. It is suggested that non-linear analysis is not suitable for use by designers, unless extreme accuracy is required.
Chapter 7

Calibration of Eurocode partial resistance factors

Partial resistance factors are required with Eurocode design to ensure the probability of a component having insufficient strength is minimised to acceptable levels. Partial factors are calibrated on the basis of ‘long experience of building tradition’ or ‘statistical evaluation of experimental data and field observations’. In the case of SCS panels, sufficient structures do not exist to use tradition as a justification, so a full statistical assessment is required.

A method for calculating partial factors can be developed from Eurocode 0 Annex D [31]. A number of papers describe the application of this process to various structures and materials, as reviewed in Section 7.1. However, this method has not been applied to SCS structures. The method is applied to the models for out-of-plane shear resistance. The method as presented in Annex D is not readily applicable to more complex failure modes, such as the bending resistance of SCS panels. An extension to the method is proposed that allows for the limitations found. This new method is termed ‘the matrix method’, since the process is most efficiently represented when the input parameters are stored in either one or two-dimensional matrices. It is shown that the result produced by this new method is the same as the one produced by the Annex D procedure for linear resistance functions, such as out-of-plane shear failure of unreinforced panels. The new method is demonstrated on the case of out-of-plane bending resistance of SCS panels.
7.1 History of Partial Factors

There is always uncertainty in structural design, which can come from variation in loads, material strengths, dimensional variances and design model consistency. Accounting for this uncertainty in design is possible by making conservative assumptions about the design parameters. However, application of too many conservative assumptions in tandem can significantly increase the cost of the structure. A balance needs to be found between cost and safety considerations, which occurs when the probability of the structure failing over its lifetime is low enough to be acceptable to society.

Probabilistic analysis is a feature of all modern design codes, although the form in which the results are represented varies. Historically, codes were presented in terms of an ‘allowable stress’, where design models are presented in terms of a material stress. Uncertainty was allowed for by applying a factor to the maximum stress to which the material can be subject.

Since the 1970’s, allowable stress codes have been superseded in favour of ‘Limit state’ design codes. In structural engineering, the most important criteria for design is ensuring that the building does not collapse, which is referred to as the ultimate limit state (ULS). The other main criteria are serviceability limit states (SLS). SLS failure is defined as a point at which the building will not collapse, but still no longer fulfils its function to its occupants. These limit states include criteria like excessive deflection (which may affect equipment, cause cracking of finishes etc.) and excessive vibration.

ULS limit states are considered more critical than SLS limit states, since ULS failure is a safety issue for users, so the probability of occurrence of a ULS failure that is allowed by design codes is typically more onerous than the allowable probability of occurrence for an SLS limit state.

Loads and resistances in real projects can not be precisely defined. Instead, the values tend to follow statistical distributions. Figure 7.1 shows a typical arrangement of probability densities for a construction product resisting an applied load effect.
CHAPTER 7: CALIBRATION OF EUROCODE PARTIAL RESISTANCE FACTORS

Figure 7.1: Typical arrangement of probability densities for a construction product resisting an applied load effect e.g. shear force, bending moment. Lines show ratio of actual applied load to nominal resistance (blue) or actual member resistance (green) to nominal resistance.

Failure occurs when the load effect exceeds the resistance. This failure condition can be met in a number of combinations, meaning a single point of failure does not exist. This is shown in Figure 7.1 by the overlapping region. Once the load and resistance distributions are known, the designer must ensure that the probability of these limit states ever being realised is sufficiently low.

A full probabilistic assessment of a structure is usually not possible for most structures, given both time constraints and the expertise required. Instead, a sufficiently low probability of failure is achieved in limit state codes through an increase in characteristic load or a decrease in characteristic resistance. The magnitude to which these values need to be adjusted are given in the code as partial factors, which are calibrated by researchers and code writers. The factors are called partial because they are usually applied in combination i.e. one factor on load, one on resistance.

The methods by which partial factors are calibrated have been the subject of considerable research. Probabilistic quantification of failure is relevant to many other topics outside of structural engineering, including, but not limited to, marine structures, aerospace and electrical engineering. Techniques developed in...
these areas have been applied to civil and structural engineering, with varying levels of complexity and precision.

One method for developing partial resistance factors for use in Eurocode design is given by EN 1990 Annex D[^31] ‘Design assisted by testing’. The method is based on the first order reliability method (FORM), and accounts for variations in both the testing and variations in parameters that make up the resistance function, such as material properties. The remaining sections of this chapter explain the theoretical background to Annex D. Application of the Annex to out-of-plane shear and bending resistance is described in Section 7.4. Thereafter, the proposed matrix method is presented, and subsequently applied to both the shear and bending models.

## 7.2 Eurocode partial factor derivations in the literature

As discussed above, a method for development of partial factors on resistance in accordance with the Eurocode is given by EN 1990 Annex D[^31]. Annex D has been applied by a number of researchers to various structures and materials. Presented below are a selection of the most relevant applications of the method to SCS panels.

### 7.2.1 Concrete structures

A number of researchers have developed partial factors for concrete design for use in the Eurocode.

Vrouwenvelder and Siemes[^189] describes some of the early work to develop partial factors for Dutch standards, which includes concrete. The results presented are largely consistent with the material partial factors presented in the current codes.

More recently, Holicky et al.[^81] describe a study that follows the Eurocode philosophy, though the study is aimed at the South African codes. The authors find material partial factors that are slightly smaller than those included in EN1992-1-1[^30]; 1.10 for steel reinforcement, compared to 1.15 in EC2, and 1.40 for concrete, compared to 1.50 in EC2. However, this derivation is based on a reduced reliability index of $\beta = 3.0$, compared to the reliability index of $\beta = 3.8$ required by Eurocode 0 (see Table B.2). It is likely that an increase in reliability index of this magnitude would increase the final values.

The paper recognises the variation of the partial factors with the values of the basic variables. Considerable attention is devoted to exploring the variation of the partial factors with the reinforcement ratio. The partial factors that are recommended as a final conclusion are based on a set of representative values, including a concrete strength of

[^189]: Vrouwenvelder and Siemes
[^30]: EN1992-1-1
[^81]: Holicky et al.
20MPa and a steel strength of 500MPa.

Recent work by Beeby and Jackson\cite{20} examines the partial factor associated with steel reinforcement. While the material of interest is steel, the analysis examines concrete beam tests, failing in both flexure and shear, since the correct partial factor must consider the context within which the material is used. Different partial factors are developed for different grades of reinforcement, reflecting the differing coefficients of variation and over-strength. However, the results are found to vary by a magnitude that is small enough to be ignored, meaning detailed argument about which value is representative of the entire population is not required.

Finally, a recent thesis by Herbrand\cite{77} includes a chapter on derivation of partial factors, for a number of new models for concrete beams in shear. The work follows the Annex D method closely, including the derivation of weighting factors for variables, as discussed in Section 7.3.4. The work captures the fact that the partial factor changes with the values of the design variables, with the variation of the partial factor with concrete strength being the main focus of discussion.

### 7.2.2 Steel-concrete composite structures

Annex D has been applied to concrete and steel-concrete composite structures by a number of researchers.

The earliest work on calibration of partial factors for composite structures for the Eurocode was performed by Johnson and Huang\cite{90}. The work applies the methods described by Vrouwenvelder and Siemes for concrete structures. The analysis is comprehensive, taking into account the variability of $V_{rt}$ with each individual design through weighting factors (as per Section 7.3.4). However, the expressions for doing so are described as too complex to include in the final paper. Subsequent criticism in a discussion paper\cite{19} suggests that the weighting factors were not compatible with CIRIA 63\cite{50} (discussed in Section 7.2.4).

Recent work by Hicks and Pennington\cite{78} updates the work of Johnson and Huang. Hicks and Pennington suggest that their work is based on a wider range of data than the work by Johnson and Huang. The authors use a numerical model with Monte-Carlo analysis to assess $V_{rt}$, rather than using the weighting factor expression used in Annex D, given as Equation 7.3.28.

The authors give a value of $V_{rt}$ that covers the whole population, including beams that have relatively low and high degrees of shear connection. Figure 3 of the paper, reproduced below as Figure 7.2 then shows the results of the simulations, which
suggest that the mean value increases and $V_{ht}$ gets smaller at higher degrees of shear connection. This is reflected in the values of the partial factor, as shown in Figure 7.3. The work finds that the current partial factors given in Eurocode 4\cite{29} remain acceptable for non-high-strength grades of steel.

![Figure 7.2](image1.png)

**Figure 7.2**: Bending resistance against degree of shear connection from a Monte Carlo simulation, taken from Figure 3 of paper by Hicks and Pennington\cite{78}

![Figure 7.3](image2.png)

**Figure 7.3**: Relationship between partial factor $\gamma_{Rd}$ and degree of shear connection, taken from Figure 4 of paper by Hicks and Pennington\cite{78}

### 7.2.3 Steel structures

From 2000 to 2003, a European project concerning probabilistic quantification of the safety of steel structures was undertaken\cite{37}. This was followed by SAFEBRITCLE, another pan-European collaborative research project to re-evaluate partial factors for steel\cite{58}. The project has generated a number of papers, in addition to the project
Stainless steel is a relatively recent addition to the Eurocodes, and is still evolving as a standard. Since there is minimal historic precedent for use of stainless steel in structures, a greater emphasis is put on probabilistic calibration. The author of this thesis was involved in a project to derive partial factors for this standard. The work is described by Afshan et al. A project report was also prepared. Subsequent analysis of stainless steel structures has used the same approach for calibration of partial factors.

Both the SAFEBRITCLE papers and the stainless steel work present extensions to the Annex D method. A particular problem is found in applying the method to column buckling, since a single representative value of each of the design parameters must be selected. This does not reflect well the changing influence of the material and geometrical uncertainties as the slenderness of the column changes. A question then arises about what is a representative sample of slendernesses in real structures. The works by Afshan et al. and Francis and Baddoo both implement the Matrix method to overcome this issue, as presented in Section 7.5, though the matrix terminology is not used. The works make use of the test results to define the distribution of basic variables across the domain being assessed, which is the argument also followed in Section 7.6.

### 7.2.4 Alternative structural reliability methods

The Eurocode 0 Annex D method presents a defined procedure for a probabilistic analysis that is compatible with the Eurocodes. It should be noted that the procedure presented is not the only one possible route to establishing the reliability of a particular structure.

One of the first comprehensive treatments of structural reliability was published by the Construction Industry Research and Information Association (CIRIA) Report 63, published in 1977. CIRIA 63 is a general purpose guide to derivation of partial factors, including reviews and references of previous academic literature up to the point of publication. The first-order reliability method (FORM), which is the basis of Eurocode 0 Annex D, is discussed, along with other methods, such as first-order second-moment methods (FOSM).

Reliability of structures is a large topic, and may be approached in a number of ways. Melchers and Beck provide a comprehensive description of a number of the techniques available. The text includes a discussion on the implementation of
7.3 Calibration of partial factor using Eurocode 0 Annex D - Design assisted by testing

Based on the work of the researchers presented in Section 7.2, the procedure for deriving a partial factor in accordance with Eurocode 0 Annex D\[^{[31]}\], from a set of test results, is presented.

This section starts with basic theory (as first presented in the Annex), and then progresses to assumptions that must be made to ensure the partial factor is applicable to its intended design domain.

7.3.1 Example resistance equations

In presenting the Annex D method, a number of the expressions and arguments are best illustrated through the use of examples. The chosen examples in this case are the expressions for out-of-plane shear resistance.

A full discussion of the model is described in Chapter 5. The key expressions are given below. Equation 7.3.1 is the equation for resistance of unreinforced panels, originally presented as Equation 5.1.1.

\[
V_{R,C} = \frac{C_{R,c}}{\gamma_c} \left[ k (100 \rho_l f_c) \right]^{1/3} b_w d \tag{7.3.1}
\]

Equations 7.3.2, 7.3.3 and 7.3.4 in combination give the resistance of a reinforced panel. The model is originally presented in Section 5.1.2.

\[
V_{R,s} = \frac{A_{sw}}{s} z \frac{f_y}{\gamma_s} \cot \theta \tag{7.3.2}
\]

\[
V_{R,max} = a_{cw} b_w z v_3 \frac{a_{cc} f_c / \gamma_c}{(\cot \theta + \tan \theta)} \tag{7.3.3}
\]

\[
V_R = \min (V_{R,max} : V_{R,s}) \tag{7.3.4}
\]
The theoretical mean resistance is given if the mean values of the basic variables are assumed and the partial factors $\gamma_c$ and $\gamma_s$ are set to 1.0. The design resistance (as usually required by users of the code) is given by taking the values of the basic variables to be their characteristic or nominal values (where appropriate), while also taking $\gamma_c = 1.5$ and $\gamma_s = 1.15$.

### 7.3.2 Theoretical basis

Test results can be analysed using EN 1990 Annex D[^31] ‘Design assisted by testing’ to calculate partial resistance factors. The method is based on the first order reliability method (FORM), and accounts for variations in both the testing and variations in parameters that make up the resistance function, such as material properties.

There are an infinite number of potential combinations under which failure can occur, as long as the following combination is satisfied:

$$E_d > R_d$$  \hspace{1cm} (7.3.5)

Where:

- $E_d$ is the design value of the effect of actions
- $R_d$ is the design value of the resistance (also given as $r_d$ in EN 1990 Annex D)

A ‘failure surface’ can therefore be defined, as shown in Figure 7.4:
The location of the point (P) on the failure line is critical, since it is the most likely combination to occur. Its location is defined by the following equation:

\[ \alpha_E^2 + \alpha_R^2 = 1 \]  \hspace{1cm} (7.3.6)

Where:

\[ \alpha_E = \frac{-\sigma_E}{\sqrt{\sigma_E^2 + \sigma_R^2}} \]  \hspace{1cm} (7.3.7)
\[ \alpha_R = \frac{-\sigma_R}{\sqrt{\sigma_E^2 + \sigma_R^2}} \]  \hspace{1cm} (7.3.8)

Where:

\( \sigma_E \) \hspace{0.5cm} \text{is the standard deviation of the effect of actions}
\( \sigma_R \) \hspace{0.5cm} \text{is the standard deviation of the resistance}

These factors are known as the FORM sensitivity factors. In order to simplify the analysis, EN 1990 makes the following assumption, which allows the load effects and the resistance effects to be separated:
\[ \alpha_E = 0.7 \quad (7.3.9) \]
\[ \alpha_R = 0.8 \quad (7.3.10) \]

The target reliability index \( \beta \) expresses the probability of failure that is deemed acceptable to society. Probability is converted to the reliability index by means of the standard normal cumulative distribution function, as shown in Equation 7.3.11:

\[ P( R < R_d ) = \Phi_E(-\alpha_R\beta) \quad (7.3.11) \]

Where:

\[ \Phi_E \] is the cumulative distribution function of the standard Normal distribution.

The value for the target reliability index \( \beta \) can be obtained from EN 1990 Annex C, Section C6. In this thesis, and most applications in civil engineering, \( \beta \) is taken equal to 3.8, suitable for Ultimate Limit States and a design life of 50 years. While this is typical for industrial buildings, nuclear applications will typically demand a higher consequence class, and hence a large \( \beta \).

Annex D combines the reliability index and the FORM sensitivity factor into \( k_d \) which is equal to \(-\alpha_R\beta\). For a large population of tests results \( k_d,inf = 0.8 \times 3.8 = 3.04 \).

The partial factor on resistance is found by rearranging Equation 6.6c of EN1990, presented below in 7.3.12:

\[ \gamma_M = \frac{r_k}{r_d} \quad (7.3.12) \]

Where:

\[ r_k \] is the characteristic resistance
\[ r_d \] is the design resistance, usually taken as 3.04 standard deviations below the mean resistance

However, this form of the expression is difficult for designers to apply, since they will
not have access to the test results that would enable them to define the characteristic resistance and the standard deviation.

The code offers an alternative form for $\gamma_M$ that relates the factor to the nominal resistance $r_n$, which can be calculated and used by the designer. This factor is designated with an asterisk (i.e. $\gamma_M^*$), as it relates to $r_n$ rather than $r_k$.

The value of $\gamma_M^*$ is calibrated by researchers by comparing the resistance of typical structures against the nominal resistance predicted by the design model, as shown in Equation 7.3.13:

$$\gamma_M^* = \frac{r_n}{r_d}$$ (7.3.13)

Where:

$r_n$ is the nominal resistance, calculated from a resistance function using the nominal values of the basic variables

$r_d$ is evaluated according to Equation 7.3.14, given as equation D.22 in EN 1990, assuming a large number of tests:

$$r_d = b g_{rt} \{X_m\} \exp (-k_{d,inf} Q - 0.5 Q^2)$$ (7.3.14)

Where:

$$Q = \sqrt{\ln \left( V_r^2 + 1 \right)}$$ (7.3.15)

$$V_r^2 = V_{\delta}^2 + V_{rt}^2$$ (7.3.16)

$V_r$ is coefficient of variation of the resistance

$V_{rt}$ is the total coefficient of variation of the basic variables

$V_{\delta}$ is the coefficient of variation of the error of the resistance prediction, estimated by linear regression; see Annex D Equation D.13

$b$ is the bias of the resistance prediction, estimated by linear regression; see Annex D Equation D.7

$k_{d,inf}$ is the characteristic fractile factor for a large number of tests, from Annex D Table D2
CHAPTER 7: CALIBRATION OF EUROCODE PARTIAL RESISTANCE FACTORS

When the number of tests in the test population used to calculate \( b \) and \( V_\delta \) is less than 100 an additional adjustment must be made, to reflect the presence of statistical (sampling) uncertainty. \( r_d \) is then calculated as:

\[
    r_d = b g_{rt} \{ X_m \} \exp \left( -k_d,n \alpha_{rt} Q_{rt} - k_d,n \alpha_\delta Q_\delta - 0.5 Q^2 \right) \tag{7.3.17}
\]

Where:

\[
Q_\delta = \sqrt{\ln \left( V_\delta^2 + 1 \right)} \tag{7.3.18}
\]

\[
Q_{rt} = \sqrt{\ln \left( V_{rt}^2 + 1 \right)} \tag{7.3.19}
\]

\[
\alpha_{rt} = \frac{Q_{rt}}{Q} \tag{7.3.20}
\]

\[
\alpha_\delta = \frac{Q_\delta}{Q} \tag{7.3.21}
\]

\( k_{d,n} \) is the characteristic fractile factor, from Annex D Table D2

\( V_\delta \) and \( b \) are calculated using comparisons between theoretical and measured resistances for a set of tests. Each test in the set produces two values; \( r_e \) for the experimental resistance and \( r_t \) for the theoretical resistance. The process required to produce these values is presented in Section D8.2.2.3 of Annex D.

Resistance functions themselves typically contain a conservative bias, equivalent to an over-strength. Code writers will tend to make conservative assumptions regarding the behaviour of the structure, in order to ensure their design is safe even if their worst possible assessment of the resistance of the structure proves to be true, which is rarely the case. This is captured by the parameter \( b \), in a mean value sense i.e. an average.

Some resistance functions are more precise than others in predicting the behaviour of the structure. Resistance functions that are less precise in their predictions should have a larger partial factor (with all other things being equal). This is reflected in the parameter \( V_\delta \).

The parameters \( V_\delta \) and \( b \) are theoretically independent of the basic variables. In reality this assumption is often incorrect e.g. the model may work better for lower strength
concrete than higher strength concrete. To overcome this issue, the design domain can be sub-divided into sub-sets, such that a value of $V_j$ is calculated for each sub-set. This is described in EN 1990 Clauses D.8.2.2.5.

### 7.3.3 Allocation of uncertainty to material or resistance model

A key consideration for the analysis is the allocation of partial factors to either the material or the resistance model. EN 1990\cite{EN1990} includes two different safety checking formats in Section 6.3.5, which are presented below as Equation 7.3.22 and Equation 7.3.23:

\[
R_d = \frac{1}{\gamma_{Rd}} R \left\{ \eta_i \frac{X_{k,i}}{\gamma_{M,i}} ; a_d \right\} 
\]  

(7.3.22)

\[
R_d = R \left\{ \eta_i \frac{X_{k,i}}{\gamma_{M,i}} ; a_d \right\} 
\]  

(7.3.23)

Where:

- $\gamma_{Rd}$ is the partial factor associated with the uncertainty of the resistance model
- $\gamma_{m,i}$ is the partial factor for the variable $i$, for a material property
- $\gamma_{M,i}$ is the partial factor for variable $i$, a material property, also accounting for model uncertainties and dimensional variations
- $R_d$ is the design value of the resistance
- $R$ is the resistance function
- $a_d$ is the design values of the geometrical data
- $\eta_i$ is the mean value of the conversion factor taking into account volume and scale effects, effects of moisture and temperature, and any other relevant parameters
- $X_{k,i}$ is the characteristic value of a material property

In Equation 7.3.22 the design resistance is obtained through the introduction of two partial factors - $\gamma_{m,i}$ for the material and $\gamma_{Rd}$ for the resistance model uncertainty. In Equation 7.3.23 the same uncertainties are allowed for, but the approximation $\gamma_{M,i} = \gamma_{m,i} \times \gamma_{Rd}$ is adopted.

The expression for out of plane shear resistance, Equation 7.3.1, includes $\gamma_c$ as a partial
Chapter 7: Calibration of Eurocode partial resistance factors

factor relating to the strength of concrete. It is clear from works describing its derivation (see work by König and Fisher[96]) that in this context $\gamma_c$ is both a material factor and a model factor, as it intended to account for the uncertainty with which the model is able to predict the resistance of the structure given the actual strength of the concrete. However, $\gamma_c$ is applied to the resistance function in Equation 7.3.1, rather than the basic variable. The implied resistance checking format is therefore:

$$R_d = \frac{1}{\gamma_M^*}R \{ \eta_iX_{k,i}; a_d \}$$  \hspace{1cm} (7.3.24)

The $\gamma_M^*$ factors output from the methods described in Sections 7.3 and 7.5 are in this form i.e. they take into account material and model uncertainty, and are applied the resistance function rather than the basic variables.

The partial factor reflects uncertainty in the overall resistance prediction, which comes from uncertainty in the accuracy of the resistance function and uncertainty in the basic variables. It is possible to allocate this uncertainty within the resistance function by adding partial factors to the basic variables.

The partial factors on the variables can conceivably be set equal to any value, so long as the relationship in Equation 7.3.25 is respected. $\gamma_M^*$ is determined by the methods described in Sections 7.3 and 7.5, while $\gamma_{Rd}$ and $\gamma_{m,i}$ are unknowns.

$$\frac{1}{\gamma_M^*}R \{ \eta_iX_{k,i}; a_d \} = \frac{1}{\gamma_{Rd}}R \{ \eta_i \frac{X_{k,i}}{\gamma_{m,i}}; a_d \}$$  \hspace{1cm} (7.3.25)

For linear resistance functions, where $R_d$ is directly proportional to $X_{k,i}$, Equation 7.3.25 may be reduced to this form:

$$\gamma_M^* = \gamma_{Rd} \times \gamma_{m,i}$$  \hspace{1cm} (7.3.26)

However, this formulation does not hold for all resistance functions. Some equations, like Equation 7.3.1, are either more complex or are dimensionally inconsistent, meaning applying the partial factors as $\gamma_m$ and $\gamma_{Rd}$ in Equation 7.3.22 yields a different design resistance to the resistance given by Equation 7.3.23, with the partial factor applied as $\gamma_{M,i}$.

As an example, Equation 7.3.1 raises the concrete strength to the power of 1/3, which results in a dimensional inconsistency. Equation 7.3.25 reduces to the form shown in Equation 7.3.27 in this case.

$$\gamma_M^* = \gamma_{Rd} \times \gamma_{f_{c,i}}^{1/3}$$  \hspace{1cm} (7.3.27)
For more complex resistance models, such as the bending resistance of SCS panels, such a simple relationship between the partial factors cannot be developed. Instead, the procedures presented in Sections 7.3 and 7.5 are repeated with the partial factors applied to the basic variables incorporated when calculating the nominal resistance.

Misunderstandings about the allocation of uncertainty can lead to misleading conclusions when comparing partial factors between different parts of the code or different materials. This is discussed further in Section 7.7.4.

### 7.3.4 Weighting factors

When computing $V_{rt}$ as per Equation 7.3.28, each variable is assigned a ‘weighting factor’, which weights the effect that small changes in the variables have on the resistance. Taking the resistance equation for out-of-plane shear (originally given as Equation 5.1.1, reproduced as Equation 7.3.1) as an example (the resistance equation for shear), a 1% change in concrete strength results in only a 0.33% change in resistance, since the concrete strength is raised to the power of 1/3. It is therefore appropriate to weigh the coefficient of variation of concrete strength by 1/3 when calculating $V_{rt}$.

Annex D gives the following equations for the calculation of $V_{rt}$, from which an expression for $w_{X,i}$ may be derived:

$$V_{rt}^2 = \frac{\text{VAR} [g_{rt}(X)]}{g_{rt}^2(X_m)} \approx \frac{1}{g_{rt}^2(X_m)} \times \sum_{i=1}^{j} \left( \left. \frac{\partial g_{rt}}{\partial X_i} \right|_{X_m} \sigma_{X,i} \right)^2 \quad (7.3.28)$$

To apply the method given in Equation 7.3.28, the derivative of the resistance function with respect to the variable $i$ is divided by the resistance function, and then multiplied by the standard deviation of the basic variable $i$. Using Equation 7.3.1 as an example:

$$\frac{\partial g_{rt}}{\partial f_c} = \frac{C_{Rd,c} \left[ k \left( 100 \rho_1 \right)^{1/3} \right] b_w d}{3 f_c^{2/3}} \quad (7.3.29)$$

This expression can then be evaluated at the mean point, $X_m$: 284
\[
\frac{\partial g_{rt}}{\partial X_i} \bigg|_{X_m} = g_{rt} \left[ \frac{C_{Rd,c} \left[ k (100 \rho_1)^{1/3} \right] b_{ud}}{3 f_{c,m}^{2/3}} \right] \bigg|_{X_m} = \frac{1}{3 f_c} \bigg|_{X_m} = \frac{1}{3 f_{c,m}} \quad (7.3.30)
\]

It is usually advantageous to express the variation of the variable in terms of the coefficient of variation rather than the standard deviation. When substituted into Equation 7.3.28, the mean value and the standard deviation of the variable \(i\) may be replaced with the coefficient of variation, accompanied by a weighting factor. This is shown in Equation 7.3.31.

\[
V_{rt}^2 = \sum_{i=1}^{j} \left\{ w_{X,i} V_{X,i} \right\}^2
\]

(7.3.31)

Where:

\(V_{X,i}\) is coefficient of variation of the variable \(i\)

\(w_{X,i}\) is the variable weighting factor, as defined in Equation 7.3.33

The definition of the variable weighting factor presented in Equation 7.3.33 is developed by rearranging the bracketed terms in Equation 7.3.28, as follows:

\[
\left( \frac{\partial g_{rt}}{\partial X_i} \right)_{X_m} \sigma_{X,i} = \left( \frac{\partial g_{rt}}{\partial X_i} \right)_{X_m} V_{X,i} = \left( \frac{\partial g_{rt}}{\partial X_i} \right)_{X_m} V_{X,i} = w_{X,i} V_{X,i}
\]

(7.3.32)

Therefore:

\[
w_{X,i} = \left( \frac{\partial g_{rt}}{\partial X_i} \right)_{X_m} \bigg|_{X_m}
\]

(7.3.33)

Using Equation 7.3.1 as an example, \(w_{f_c}\) can be calculated as follows:

\[
w_{f_c} = \left[ \frac{C_{Rd,c} \left[ k (100 \rho_1)^{1/3} \right] b_{ud}}{3 f_{c,m}^{2/3}} \right] \bigg|_{X_m} = \frac{1}{3} \times f_{c,m} = \frac{1}{3}
\]

(7.3.34)

The interrelation between the two forms of the expression can be expressed simply as follows:
For linear resistance functions $w_{X,i}$ evaluated according to Equation 7.3.33 is constant for all values of $X_i$.

The expressions presented above are strictly valid if the resistance function is linear and the expansion of the variables is carried out at the mean values of the variables i.e. $X = X_m$. Considerable discussion on this topic is presented in Section 4.5 of the textbook by Melchers and Beck\textsuperscript{114}.

### 7.3.5 Variability associated with the basic variables

The term $V_{rl}$ (calculated in Equation 7.3.28) should include the variation of all of the variables in the resistance function, which includes both material and dimensional variation.

This work includes consideration of two materials; steel and concrete. Values of $V_{X,i}$ (i.e. $V_{f_y}$ and $V_{f_c}$ respectively) are widely reported in the literature for various materials (and geometric variations if required).

A number of studies have measured the properties of steel. It has generally been found that the COV of steel has decreased over time, as technology has improved\textsuperscript{114}. The final values used for this study are taken from Melchers and Beck\textsuperscript{114}. For reinforcing steel:

$$V_{f_y} = 5\%$$  \hspace{1cm} (7.3.36)

$$\frac{f_{ym}}{f_{yn}} = \frac{560\text{MPa}}{500\text{MPa}} = 1.12$$  \hspace{1cm} (7.3.37)

For hot-rolled steel:

$$V_{f_y} = 5\%$$  \hspace{1cm} (7.3.38)

$$\frac{f_{ym}}{f_{yn}} = 1.15$$  \hspace{1cm} (7.3.39)

The relationship between the characteristic strength of concrete and $V_{f_c}$ is more complicated than for steel, since tests have shown that concrete of different grades tend to have a fixed standard deviation rather than a fixed coefficient of variation.
Chapter 7: Calibration of Eurocode Partial Resistance Factors

EN 1992-1-1 Table 3\[30\] shows that the relationship between the nominal strength of concrete \(f_{c,k}\) and the mean strength \(f_{c,m}\). Since the nominal strength is also the 5% characteristic strength, it can be assumed that the difference between the mean strength and the nominal strength is equal to 1.64 standard deviations. Thus, assuming a normal distribution for \(f_c\):

\[
f_{c,m} = f_{c,k} + 8 = f_{c,k} + 1.64\sigma_f\tag{7.3.40}
\]

Therefore:

\[
\sigma_f = \frac{8}{1.64} = 4.87\text{MPa}\tag{7.3.41}
\]

\[
V_{f_c} = \frac{\sigma}{f_{c,m}} = \frac{4.87}{f_{c,m}}\tag{7.3.42}
\]

\[
f_{c,m}/f_{c,n} = (f_{c,k} + 8)/f_{c,k}\tag{7.3.43}
\]

Application of this relationship leads to different partial factors for different concrete grades. This is shown in Section 7.4.

SCS panels contain many components that can dimensionally vary, including the plate thicknesses, spacing of shear connectors and the overall depth of the section. It is possible to include a COV for each of these sources of variation in Equation 7.3.31. However, statistical data presented by Melchers and Hough\[115\] for reinforced concrete structures shows that dimensional variation is much smaller than variation in material properties, meaning dimensional variation can be ignored without significant changes to \(V_r\). This proved to be the case in preliminary analysis of SCS panels; when dimensional variability was included, no significant changes in \(V_r\) were observed. For completeness, the combined effect of all dimensional variation is assumed to have a COV of 2%.

7.3.6 \(V_{X,\text{known}}\) and \(V_{X,\text{unknown}}\)

Annex D includes increased values for \(k_n\) when the sample size is small. This reflects the increased uncertainty that the small population does not cover all of the potential design permutations that could occur in real design. The level of adjustment required depends on the level of knowledge about the potential size of the coefficient of variation. If there is ‘full knowledge’, then the adjustments for \(V_{X,\text{known}}\) may applied.
If there is ‘no prior knowledge about the coefficient of variation’ the adjustments for $V_{X,\text{unknown}}$ must be applied.

Little guidance on the level of knowledge required to justify the use of $V_{X,\text{known}}$ is given. It is the opinion of the author that use of $V_{X,\text{known}}$ adjustment is appropriate for the analysis presented in this chapter. The behaviour in testing has shown consistency with tests on conventional reinforced concrete panels, and as such it can be anticipated that no further failure mechanisms could occur that would increase the coefficient of variation observed.

### 7.4 Application of the procedure to the resistance model for Out-of-plane shear

This section shows the results of applying the procedure given in Section 7.3 to out-of-plane shear resistance.

The results are presented in Table 7.10 and Table 7.11. Assessment is based on the tests described in Chapter 3, split into ‘reinforced’ and ‘unreinforced’. $n$ is taken as the total number of tests in the sub-population.
7.4.1 Unreinforced panels in shear

Table 7.10: Calculation of $\gamma_M^*$ for unreinforced SCS panels failing in shear, using the Annex D approach

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Test</th>
<th>$r_{ti}$</th>
<th>$r_{ei}$</th>
<th>$r_{ei}/r_{ti}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>131</td>
<td>E1</td>
<td>33</td>
<td>38</td>
<td>1.17</td>
</tr>
<tr>
<td></td>
<td>E2</td>
<td>33</td>
<td>52</td>
<td>1.61</td>
</tr>
<tr>
<td>184</td>
<td>SP1-1</td>
<td>160</td>
<td>206</td>
<td>1.29</td>
</tr>
<tr>
<td></td>
<td>SP1-3</td>
<td>179</td>
<td>224</td>
<td>1.25</td>
</tr>
<tr>
<td></td>
<td>SP1-4</td>
<td>160</td>
<td>230</td>
<td>1.43</td>
</tr>
<tr>
<td>105</td>
<td>JZ3.0-1</td>
<td>124</td>
<td>200</td>
<td>1.61</td>
</tr>
<tr>
<td></td>
<td>JZ3.5-1</td>
<td>124</td>
<td>140</td>
<td>1.13</td>
</tr>
<tr>
<td></td>
<td>JZ3.0-N</td>
<td>107</td>
<td>135</td>
<td>1.26</td>
</tr>
<tr>
<td></td>
<td>JZ2.5-2</td>
<td>136</td>
<td>233</td>
<td>1.72</td>
</tr>
<tr>
<td></td>
<td>JZ3.0-2</td>
<td>136</td>
<td>220</td>
<td>1.62</td>
</tr>
</tbody>
</table>

The following values represent the model uncertainty:

$V_\delta = 15.43\%$

$b = 1.4$

As Equation 7.3.42 shows, the coefficient of variation of concrete varies with the strength, which the implication being that the final value of $\gamma_M^*$ also varies with strength. To demonstrate this, two concrete strengths are chosen. For $f_{c,k} = 30N/mm^2$:

$$V_{f_c} = \frac{4.87}{f_{c,m}} = \frac{4.87}{(30 + 8)} = 12.84\%$$ (7.4.1)

Therefore:

$$V_r = \sqrt{V_\delta^2 + V_{f_c}^2}$$

$$= \sqrt{V_\delta^2 + \frac{w_{f_c}V_{f_c}^2 + V_{dim}^2}{f_{c,m}} + V_{dim}^2}$$

$$= \sqrt{15.43^2 + \left(\frac{1}{3} \times 12.84\right)^2 + 2^2}$$

$$= 16.12\%$$ (7.4.2)
When this result is entered into Equation 7.5.9 the final value of $\gamma_M^*$ derived, assuming $V_{X,\text{known}}$ is 1.11. The equivalent value assuming $V_{X,\text{unknown}}$ is 1.35.

For $f_{c,k} = 40 N/mm^2$:

$$V_{f_c} = \frac{4.87}{f_{c,m}} = \frac{4.87}{(40 + 8)} = 10.16\% \quad (7.4.3)$$

Therefore:

$$V_r = \sqrt{V_{\delta}^2 + V_{ri}^2}$$

$$= \sqrt{V_{\delta}^2 + w_f c_f V_{f_c}^2 + V_{dim}^2}$$

$$= \sqrt{15.43^2 + \left(\frac{1}{3} \times 10.16\right)^2 + 2^2}$$

$$= 15.91\% \quad (7.4.4)$$

When this result is entered into Equation 7.5.9 the final value of $\gamma_M^*$ derived, assuming $V_{X,\text{known}}$ is 1.13. The equivalent value assuming $V_{X,\text{unknown}}$ is 1.37.

### 7.4.2 Reinforced panels in shear

The test results for the reinforced cases are presented in Table 7.11.

**Table 7.11:** Calculation of $\gamma_M^*$ for reinforced SCS panels failing in shear, using the Annex D approach

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Test</th>
<th>$r_{ti}$ kN</th>
<th>$r_{ei}$ kN</th>
<th>$r_{ei}/r_{ti}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>82</td>
<td>NR-0R-3S400-4ST*</td>
<td>583</td>
<td>1056</td>
<td>1.81</td>
</tr>
<tr>
<td></td>
<td>NRC-0R-4S400-4ST*</td>
<td>777</td>
<td>1236</td>
<td>1.59</td>
</tr>
<tr>
<td>98</td>
<td>SP3_E2</td>
<td>926</td>
<td>1287</td>
<td>1.39</td>
</tr>
<tr>
<td></td>
<td>SP4_E2</td>
<td>1093</td>
<td>1165</td>
<td>1.07</td>
</tr>
<tr>
<td></td>
<td>SP6_E2</td>
<td>931</td>
<td>919</td>
<td>0.99</td>
</tr>
<tr>
<td>105</td>
<td>JZ2.5-1</td>
<td>137</td>
<td>197</td>
<td>1.44</td>
</tr>
<tr>
<td></td>
<td>JZ3.0-3</td>
<td>165</td>
<td>275</td>
<td>1.67</td>
</tr>
<tr>
<td></td>
<td>JZ3.0-4</td>
<td>174</td>
<td>258</td>
<td>1.49</td>
</tr>
<tr>
<td></td>
<td>JZ3.5-2</td>
<td>165</td>
<td>270</td>
<td>1.64</td>
</tr>
</tbody>
</table>

The following values represent the model uncertainty:
CHAPTER 7: CALIBRATION OF EUROCODE PARTIAL RESISTANCE FACTORS

\[ V_\delta = 20.74\% \]
\[ b = 1.28 \]

Given all of the tests in population are in design region 2 (as defined in Section 5.1.2), the resistance is controlled entirely by the strength of the steel reinforcement, calculated using Equation 7.3.2. It can be shown that the weighting factor for steel (calculated in accordance with Equation 7.3.3) in this case is equal to 1.0, and is independent of the strength of the material.

\[
V_r = \sqrt{V_\delta^2 + V_{f_5}^2 + V_{dim}^2}
\]
\[ = \sqrt{20.74^2 + 5^2 + 2^2} \]
\[ = 21.98\% \]  
(7.4.5)

When this result is entered into Equation 7.5.9 the final value of \( \gamma_M^* \) derived, assuming \( V_{X,known} \), is 1.45. The equivalent value assuming \( V_{X,unknown} \) is 1.95.

### 7.4.3 Discussion

For unreinforced panels, the partial factor calculated from this analysis is considerably smaller than the 1.5 used found in Eurocode design for conventional reinforced concrete structures. Given the behaviour in shear is very similar to reinforced concrete, as discussed in Section 5.2, it should be expected that a similar partial factor should be applied. It is possible that the inclusion of panels with a small amount of shear reinforcement in this sample may not be justified, given these cases have produced the three highest resistance ratios (shown in the final column in Table 7.10). It is therefore recommended that the partial factor from Eurocode 2\textsuperscript{[30]} be adopted for SCS design. Should further tests become available, a revised value may be calculated. The \( b \) value for the unreinforced panels (1.40) is also higher than the \( b \) for reinforced panels (1.28), which suggests that the unreinforced model has a greater degree of overall conservatism.

For reinforced panels, it can be seen that the partial factors produced from this analysis are much larger than the equivalent partial factors for a conventional reinforced concrete beam or panel, as found in EN1992-1-1\textsuperscript{[30]}. Since the design model generally calculates conservative results (as shown in Table 7.11), these high values can be attributed mostly to the high scatter of the resistances calculated.

The results also highlight the large adjustments that are required to account for small
population sizes, particularly where \( V_{X,\text{unknown}} \) is assumed. However, where \( V_{X,\text{known}} \) is assumed and the sample contains at least 20 points it can be seen that the adjustment does not result in large changes in \( \gamma_M^* \). This suggests that a sample size of at least 20 points may be enough to obtain a reasonable value from an Annex D analysis.

### 7.4.4 Limitations of existing method

The results presented above show that the Annex D methodology for calculating partial factors is readily applied to the resistance of panels to out-of-plane shear forces. However, the method is not as applicable to all design equations in the Eurocodes. In seeking to apply the method to the bending resistance of SCS panels, two key limitations were exposed; (1) The design equation must be analytically differentiable, which is not always possible; and (2) The definition of a single set of representative variables for which the partial factor is evaluated may not adequately represent the whole population.

This work proposes a new procedure for calculation of partial factors that overcomes the limitations. The new method applies the existing Annex D procedure, but extends the calculation to a number of representative cases, from which the average is taken to give the final partial factor. A novel terminology is used, which represents the input data required and the numerical processes in matrix form. This method is presented in Section 7.5.

### 7.5 The matrix method

As shown in Section 7.3, Eurocode 0\[^{[31]}\] includes a method for developing partial factors that can be applied to many resistance functions. However, the method is not easily applied to complex or implicit resistance functions. This section explores a new extension to the method that allows calculation of partial factors to be calculated in these situations.

The key issues that the new method addresses are:

- \( V_M \) and \( w_{X,i} \) can vary across a population of designs - When evaluating Equation 7.3.33, it is usually the case for linear resistance functions that a constant value of \( w_{X,i} \) is achieved. For more complex resistance functions this is not necessarily the case. In many cases the weighting factor for a particular variable can vary with respect to other variables in the resistance equations.
The net effect is that $\gamma_M^*$ can change in response to the point at which the evaluation is performed. This can be seen in Section 7.4, where the value of $\gamma_M^*$ changes with the concrete grade. In this case the two values of $\gamma_M^*$ are roughly the same, so there is no issue with interpretation of the result. However, this is not always true. Section 7.7.3 shows an example where $\gamma_M^*$ changes in response to the degree of shear connection.

To overcome this issue, an averaging process is proposed, based on the results for a set of representative cases. The sample included in the representative cases is designed to represent the likelihood of a particular combination of variables appear in real designs.

- Resistance equation must be differentiable in closed form - The calculation of $w_{X,i}$ according to Equation 7.3.33 relies on the resistance equation being differentiable by analysis. In many cases this is not possible, since the equation may include discontinuities due to minimum and maximum functions. Shear stud resistance is an example (see Equation 4.1.9).

Numerical differentiation presents a solution to this problem. A new equation for calculating the weighting factors that does not require closed form differentiation of the resistance function is described in Section 7.5.3.

The remainder of this section describes the theoretical basis and the application of the method to the problem of bending resistance of SCS panels.

### 7.5.1 Theoretical basis

For many of the equations presented, an exact functional equivalent may be found in Section 7.3.2. However, interoperation of the functions in the matrix presentation demands revised terminology.

The final value of $\gamma_M^*$ is calculated in accordance with Equation 7.5.1:

$$
\gamma_M^* = \frac{\sum_{q=1}^{n_q} \gamma_{M,q}^*}{n_q} \\
= \frac{\sum_{q=1}^{n_q} \left( \frac{r_{n,q}}{r_{d,q}} \right)}{n_q} \\
= \frac{\sum_{q=1}^{n_q} \left( \frac{g_{rt} \left\{ X_{n,i,j,q} \right\}}{r_{d,q}} \right)}{n_q}
$$

(7.5.1)
In this context \( n_q \) is the number of a representative cases that will be included in the averaging process, defined in accordance with the guidance presented in Section 7.5.2. Each case has a separate value of \( q \).

\( r_{d,q} \) is evaluated according to Equation 7.5.2, given as Equation D.21 in EN 1990, assuming a large number of tests:

\[
r_{d,q} = b_{grt} \left\{ X_{m,i,q} \right\} \exp \left\{ -k_{d,inf} Q_q - 0.5 Q_q^2 \right\}
\]

(7.5.2)

Where:

\[
Q_q = \sqrt{\ln \left( V_{r,q}^2 + 1 \right)}
\]

(7.5.3)

\[
V_{r,q}^2 = V_o^2 + V_{rt,q}^2
\]

(7.5.4)

\[
V_{rt,q}^2 = \sum_{i=1}^{j} \left( w_{X,i,q} V_{X,i,q} \right)^2
\]

(7.5.5)

Where:

\( j \)  

is the number of basic variables in the resistance function

Equation 7.5.1 may therefore be presented in the following form:

\[
\gamma_M^* = \sum_{q=1}^{n_q} \left( \frac{g_{rt} \left\{ X_{n,i,q} \right\}}{b_{grt} \left\{ X_{m,i,q} \right\} \times b \times \exp \left\{ -k_{d,inf} Q_q - 0.5 Q_q^2 \right\}} \right) / n_q
\]

(7.5.6)

For convenience, the parameter \( b_{rt,q} \) is defined as follows:

\[
b_{rt,q} = \frac{g_{rt} \left\{ X_{m,i,q} \right\}}{g_{rt} \left\{ X_{n,i,q} \right\}}
\]

(7.5.7)

The final form of the partial factor calculation is therefore:

\[
\gamma_M^* = \sum_{q=1}^{n_q} \left( \frac{1}{b_{rt,q} \times b \times \exp \left\{ -k_{d,inf} Q_q - 0.5 Q_q^2 \right\}} \right) / n_q
\]

(7.5.8)
When the number of tests in the test population used to calculate $b$ and $V_\delta$ is less than 100 an additional adjustment must be made, to reflect the fact the result is less certain. The partial factor in this case is calculated as:

$$
\gamma_M^* = \frac{\sum_{q=1}^{n_q} \left( \frac{1}{b_{rt,q} \times b \times \exp \left\{ -k_{d,inf} \alpha_{rt,q} Q_{rt,q} - k_{d,n} \alpha_{\delta,q} Q_\delta - 0.5 Q_q^2 \right\}} \right) / n_q}{(7.5.9)}
$$

Where:

$$
Q_\delta = \sqrt{\ln \left( V_\delta^2 + 1 \right)}
$$

$$
Q_{rt,q} = \sqrt{\ln \left( V_{rt,q}^2 + 1 \right)}
$$

$$
\alpha_{rt,q} = \frac{Q_{rt,q}}{Q_q}
$$

$$
\alpha_{\delta,q} = \frac{Q_\delta}{Q_q}
$$

$k_{d,n}$ is the characteristic fractile factor, from Annex D Table D2

$V_\delta$ and $b$ are calculated by comparison of model predictions against tests, using the exact same approach used in Section 7.3.2.

The statistical properties of the basic variables are captured in the following three two-dimensional arrays:

$$
V_{X,i,q} = \begin{pmatrix}
V_{X,1,1} & V_{X,2,1} & \cdots & V_{X,i,1} \\
V_{X,1,2} & V_{X,2,2} & \cdots & V_{X,i,2} \\
\vdots & \vdots & \ddots & \vdots \\
V_{X,1,q} & V_{X,2,q} & \cdots & V_{X,i,q}
\end{pmatrix}
$$

(7.5.14)

$$
X_{n,i,q} = \begin{pmatrix}
X_{n,1,1} & X_{n,2,1} & \cdots & X_{n,i,1} \\
X_{n,1,2} & X_{n,2,2} & \cdots & X_{n,i,2} \\
\vdots & \vdots & \ddots & \vdots \\
X_{n,1,n_q} & X_{n,2,n_q} & \cdots & X_{n,i,n_q}
\end{pmatrix}
$$

(7.5.15)
The representative structures will be selected in terms of the nominal values of the basic variables. \( X_{m,i,q} \) is then filled using the over-strength ratios of each of the variables, defined as \( b_X = \frac{X_m}{X_n} \). Therefore, an alternative form for Equation 7.5.15 is:

\[
X_{m,i,q} = \begin{pmatrix}
X_{m,1,1} & X_{m,2,1} & \cdots & X_{m,j,1} \\
X_{m,1,2} & X_{m,2,2} & \cdots & X_{m,j,2} \\
\vdots & \vdots & \ddots & \vdots \\
X_{m,1,n_q} & X_{m,2,n_q} & \cdots & X_{m,j,n_q}
\end{pmatrix}
\] (7.5.16)

The probability of a designer selecting a particular value for a design variable in structural engineering is not usually adequately described by a continuous statistical distribution. As an example, only two steel strengths are commonly used in the UK for bridges and buildings; S275 and S355. The chance of selecting one of these values is not random, but instead depends on design criteria and current availability. The same principle is applied to concrete strengths (though a greater variety of grades are available) and plate thicknesses, which are only available in discrete sizes (8mm, 10mm, 12mm etc.).

In addition, the selection of a particular nominal value of a variable for use in a structure is likely to be linked to the selection of other variables in the design. For example, in SCS panels it would be expected that designs with greater depths would also incorporate the thickest plates, since a thick panel will typically be subject to higher forces than a thin panel.

The issue of obtaining a representative sample of typical structures is important in
more refined statistical analysis of structures, and has therefore been explored in other publications. CIRIA 63\textsuperscript{[50]} includes a detailed discussion of this topic. The approach taken by CIRIA was to survey a number of existing buildings, assuming these are representative of the choices designers would typically make (see section 7.6.2 CIRIA 63\textsuperscript{[50]}).

The approach followed by CIRIA is not possible in this instance, since typical buildings constructed using SCS panels do not currently exist. It may be possible to obtain typical designs based on existing design rules, but part of reason for the RFCS project SCIENCE (See Section 2.2) is to allow larger and more slender panels in future designs, which will mean previous designs are no longer typical\textsuperscript{[4]}.

In the absence of better information, a reasonable assumption might be that tests given in the literature are representative of the population. This approach is used to define representative samples in works by Afshan et al.\textsuperscript{[1]} and Cajot et al.\textsuperscript{[37]}.

This assumption is somewhat contentious, since test results may not be representative for a number of reasons. Firstly, tests are often conducted at smaller scale, especially for large structures like SCS panels. Tests are also often designed to exaggerate a single failure mode; for example, panels may be designed with large connector spacings to ensure compression plate buckling occurs (see test SP2 by Koukkari and Fülöp\textsuperscript{[98]}). In reality these spacings are specifically excluded by the prevailing design codes (the SCIENCE Design Manual\textsuperscript{[4]} in Europe), meaning a panel is unlikely to be built in such a consideration.

It is expected that as SCS panel technology matures the specification of a representative sample may be revisited.

7.5.3 Non-linear resistance functions

For non-linear resistance functions, the weighting factor calculated in accordance with Equation 7.3.35 may not be a fixed value.

For more complex design expressions, analytical differentiation becomes impossible. Taking bending as an example, a discontinuity appears as the degree of shear connection changes from less than 100% to greater than 100%, since the resistance is now governed by a different failure mechanism. A discontinuity also appear in the stud resistance formula (Equation 4.1.9), as the resistance changes from being governed by steel failure to concrete failure. Numerical differentiation presents a solution to this problem.

For more complex resistance functions it is also possible that the value of the derivative
with respect to the \( i_{th} \) variable may also be dependent on the values of other variables in \( \bar{X}_{m,q} \). The implication of this is that the weighting factor on each variable changes with each value of \( q \) i.e. with each representative design.

In accordance with the proof given in Section 6.2.3.2 of CIRIA 63\[^{50}\], the values of the variables must be taken as their mean values when the numerical evaluation of the derivatives is performed.

Consider the value of the variable \( i \) increased by a small amount, \( \delta \). The resistance also then increases or decreases in proportion to the weighting factor for the variable. The weighting factor is then given as below:

\[
\frac{\partial g_{rt}}{\partial X_i} = \frac{g_{rt}(X_i (1 + \delta)) - g_{rt}(X_i)}{\delta X_i}
\]  
(7.5.18)

\[w_i = \left( \frac{\partial g_{rt}}{\partial X_i} \right) X_i \bigg|_{\bar{X}_m} = \lim_{\delta \to 0} \frac{g_{rt}(X_i (1 + \delta)) - g_{rt}(X_i)}{g_{rt}(X) \cdot \delta X_i} \bigg|_{\bar{X}_m} = \lim_{\delta \to 0} \frac{g_{rt}(X_i (1 + \delta)) - g_{rt}(X_i)}{g_{rt}(X) \cdot \delta} \bigg|_{\bar{X}_m}
\]  
(7.5.19)

It can be demonstrated the weighting factor calculated according to Equation 7.5.19 is the same as the one produced by Equation 7.3.33, provided \( \delta \) is sufficiently small.

For Equation 7.3.1 (the resistance of an unreinforced SCS panel), the calculation is performed as follows. The concrete is assumed to have a characteristic strength of 30 N/mm\(^2\). The evaluation is therefore performed at the mean value of the variable associated with this characteristic strength i.e 38 N/mm\(^2\) (See Equation 7.3.43).

Although in this case the remaining variables are arbitrary, the resistance is determined for a 500x500mm panels, with 2% shear reinforcement. \( d \) is assumed to equal 0.9\( h \).

For the purposes of demonstration \( \delta \) is assumed to equal 0.02, though a much smaller value is usually required for non-linear evaluations. If the resistance function contains a discontinuity between \( g_{rt}(X_i (1 + \delta)) \) and \( g_{rt}(X_i) \) the value of \( w_i \) may be affected by the choice of \( \delta \). If this is suspected, choosing a smaller value of \( \delta \) will usually correct the issue.

The mean resistance of the panel \( g_{rt}(X_i) \) is given by Equation 7.5.20:

\[g_{rt}(X_i) = g_{rt}(f_{c,m} : b : d : \rho_t) = g_{rt}(38 : 500 : 500 : 2\%) = 499.55 \text{ kN}
\]  
(7.5.20)
The resistance of the panel assuming a concrete strength increased by $\delta (g_{rt} (X_i (1 + \delta)))$ is given by Equation 7.5.21:

\[
g_{rt} (X_i (1 + \delta)) = g_{rt} (f_{c,m} (1 + \delta) : b : d : \rho_t) \\
= g_{rt} (38 \times 1.02 : 500 : 500 : 2\%) \\
= 502.85 \text{ kN}
\] (7.5.21)

The weighting factor is then calculated as follows:

\[
w_i = \frac{g_{rt} (X_i (1 + \delta)) - g_{rt} (X_i)}{g_{rt} (X_i) \cdot \delta} = \frac{(502.85 - 499.55)}{(499.55 \times 0.02)} = 0.33
\] (7.5.22)

For out-of-plane shear design of SCS panels, the design is governed by three equations, with the weighting factor depending on the relative resistance of the concrete to crushing to the resistance of the reinforcing bars to yielding.

Equation 7.3.4 is an example of a complex resistance function. Through the minimisation, and the variable angle of $\theta$, the function includes a number of design regions, each of which has varying contributions from either the steel or concrete. This is discussed further in Section 5.1.2.

Evaluating Equation 7.5.19 at various relative reinforcement levels captures the changing influence of the materials on the resistance. This is shown in Figure 7.5, for an example case. In region 2, the resistance is controlled entirely by the reinforcing steel, which is reflected in $w_{\text{steel}} = 1$ and $w_{\text{conc}} = 0$. Conversely, region 4 is controlled by the concrete, which is again reflected in the weighting factors. Region 3 shows a combination of the two.
Similar relationships can be observed for other complex resistance functions. Afshan et al.\cite{1} describes the derivation of weighting factors for steel columns. In these examples, the weighting factor on steel strength is shown to be higher than the weighting factor for cross-sectional area variability for non-slender columns. As the slenderness increases, with resistance increasingly controlled by the geometrical properties of the column rather than the strength, the weighting factor on strength decreases while the weighting on area increases.

It should be noted that a plot of $b_{rt,q}$ (as defined by Equation 7.5.7) also shows useful information about the regions of design and the relative effects of changes to the variables. An example of a plot of this type can be seen in Figure 7.6. The value of this evaluation shows how the variable over-strength is reflected in the net over-strength of resistance function. In region 2, the resistance is proportional to the steel strength, meaning the over-strength of the resistance function is proportional to the over-strength of the steel, taken as 1.22. In region 4 the resistance is governed by the concrete, so the net over-strength is proportional to over-strength of concrete,
evaluated as \((f_{c,k} + 8)/f_{c,k}\). Region 3 shows a transition between the two, as the contribution of each material changes. The ‘kink’ at the start is due to the mean resistance and the design resistance being in different regions.

![Graph showing variation of parameter \(b_{rt}\) for an example SCS structure with varying reinforcement ratio \((f_c = 35\, \text{MPa}, f_y = 355\, \text{MPa})\)]

Figure 7.6: Variation of the parameter \(b_{rt}\) for an example SCS structure with varying reinforcement ratio \((f_c = 35\, \text{MPa}, f_y = 355\, \text{MPa})\)

### 7.6 Application of the matrix method to panels unreinforced in shear

This section describes the application of the matrix method to SCS panels unreinforced in shear. The aim of the section is to demonstrate the results of the analysis are the same as the resistance given by the Annex D procedure (as presented in Section 7.4) when the resistance function is linear, while also demonstrating the new matrix terminology.

To begin, an array of nominal values of the basic variables for a set of representative values is defined. This is shown below:

\[
X_{n,i} = \begin{pmatrix} f_{c,k} & \rho_l & b & T \end{pmatrix} \quad (7.6.1)
\]
CHAPTER 7: CALIBRATION OF EUROCODE PARTIAL RESISTANCE FACTORS

The array is filled with a set of typical designs, which in this case correspond to the test geometries presented in Table 7.10. While it can be proved mathematically that only the concrete strength affects the results, the remaining variables are presented for clarity.

\[
X_{n,i,q} = \begin{pmatrix}
38.96 - 8 & 4\% & 150 & 150 \\
37.68 - 8 & 4\% & 150 & 150 \\
42.1 - 8 & 1.4\% & 306.19 & 457 \\
42.1 - 8 & 2.08\% & 306.19 & 457 \\
42.1 - 8 & 1.4\% & 306.19 & 457 \\
40 - 8 & 2\% & 300 & 300 \\
40 - 8 & 2\% & 300 & 300 \\
25.6 - 8 & 2\% & 300 & 300 \\
30.6 - 8 & 2.26\% & 300 & 380 \\
30.6 - 8 & 2.26\% & 300 & 380 \\
\end{pmatrix} \quad (7.6.2)
\]

Once the nominal values of the variables are defined, a second array is filled with the equivalent mean values of basic variables. The relationship between the mean value and the nominal value is typically taken from standard data, as described in Section 7.3.5. For many variables, especially geometric variables, the mean value is equal to the nominal value.

\[
X_{m,i} = \left( f_{c,k} \times \frac{f_{k+8}}{f_{j,k}} \rho \times 1 \quad b \times 1 \quad T \times 1 \right) \quad (7.6.3)
\]

\[
X_{m,i,q} = \begin{pmatrix}
38.96 & 4\% & 150 & 150 \\
37.68 & 4\% & 150 & 150 \\
\vdots & \vdots & \vdots & \vdots \\
\end{pmatrix} \quad (7.6.4)
\]

Finally, a third array containing the coefficients of variation of the basic variables is constructed:

\[
\overline{V}_{X,i,q} = \begin{pmatrix}
4.87 & 0\% & 0\% & 0\% \\
4.87 & 0\% & 0\% & 0\% \\
\vdots & \vdots & \vdots & \vdots \\
\end{pmatrix} \quad (7.6.5)
\]

It should be noted that, in this case, the individual values of the coefficients of variation of the geometrical properties are not used. Instead, a 2% allowance for geometrical variation is assumed when calculating \( V_{rt} \) according to Equation 7.5.9. This assumption
is only valid if geometrical variability is significantly less than material variability, as discussed in 7.4.

The weighting factors for each variable are calculated:

\[
\mathbf{w}_{X,i,q} = \begin{pmatrix}
\frac{1}{3} & 1 \\
\frac{1}{3} & 1 \\
\vdots & \vdots 
\end{pmatrix}
\]  
\[
\mathbf{w}_{X,i,q} = \begin{pmatrix}
\frac{1}{3} & 1 \\
\frac{1}{3} & 1 \\
\vdots & \vdots 
\end{pmatrix}
\]  
\[
(7.6.6)
\]

The \(V_{X,i,q}\) used in the calculation is therefore:

\[
V_{X,i,q} = \begin{pmatrix}
1 & 3 \\
3 & 4.87 \\
2\% & (30 + 8) \\
1 & 3 \\
3 & 4.87 \\
2\% & (40 + 8) \\
\vdots & \vdots 
\end{pmatrix}
\]  
\[
(7.6.7)
\]

\(b_{rt,q}\) is calculated for each case according to Equation 7.5.7:

\[
b_{rt,q} = \frac{g_{rt} \{ X_{m,q} \}}{g_{rt} \{ X_{n,q} \}} = \frac{C_{Rd,c} k (100\rho_lf_{cm})^{1/3}}{C_{Rd,c} k (100\rho lf_{ck})^{1/3}} b_{wd}
\]  
\[
(7.6.8)
\]

After cancelling, \(b_{rt,q}\) is calculated for each case as:

\[
b_{rt,q} = \frac{f_{cm}^{1/3}}{f_{ck}^{1/3}} = \left(\frac{f_{ck} + 8}{f_{ck}}\right)^{1/3}
\]  
\[
(7.6.9)
\]

Applying Equation 7.5.9 to each row in the matrices i.e. \(q = 1\) to \(n_q\), leads to:

\[
\mathbf{\Gamma}_{M,q}^* = \begin{pmatrix}
1.11 \\
1.11 \\
1.12 \\
1.12 \\
1.12 \\
1.12 \\
1.08 \\
1.10 \\
1.10 
\end{pmatrix}
\]  
\[
(7.6.10)
\]

The final value of \(\mathbf{\Gamma}_M^*\) may be taken as the average:

\[
\mathbf{\Gamma}_M^* = 1.11
\]  
\[
(7.6.11)
\]
Alternatively, the final value of $\gamma_{M}^{*}$ may be taken as the maximum. This would imply that all of the designs considered would meet the target reliability specified. In this case there is no practical difference between the average and the maximum, but this will not always be the case. Either choice is valid, depending on the preference of the calibrator.

### 7.7 Application of the matrix method to panels in bending

This section describes the application of the matrix method to the bending resistance of SCS panels.

The formulation for bending resistance is more complex than the formulation for shear. As presented in Section 4.1, the cross-sectional resistance is established through plastic analysis, which may include contributions from the steel plates and concrete core. For cases with degrees of shear connection lower than 100% the force that may develop in the bottom plate is limited to the force that can be developed in the shear connection, equal to the sum of the shear connector resistances. Shear connector resistance may be governed by the resistance of the stud itself or the surrounding concrete. It is this interdependency of the variables that makes the resistance function impossible to differentiate analytically.

#### 7.7.1 Calculation of $V_{\delta}$ and $b$

As per the procedure presented in Section 7.5, the first step in the calculation is to establish $V_{\delta}$ and $b$, based on comparison with test results.

Similarly to the shear resistance, bending is also the subject to enhancements in resistance due to arching action. Figure 7.7 shows a plot of the test resistance against the predicted resistance for all of the cases predicted to fail by bending in the test database.
CHAPTER 7: CALIBRATION OF EUROCODE PARTIAL RESISTANCE FACTORS

Figure 7.7: Ratio of test to model resistances for different shear span to effective depth ratios for members failing in bending

For the same reasons presented in Section 7.4, panels loaded with an $a/d$ ratio less than 2.5 are removed from the population. This leaves the cases presented in Table 7.14.

Table 7.14: Calculation of $V_s$ and $b$ for SCS panels failing in bending

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Test</th>
<th>$r_{ti}$ kN</th>
<th>$r_{ei}$ kN</th>
<th>$r_{ei}/r_{ti}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>131</td>
<td>A1</td>
<td>17</td>
<td>16</td>
<td>0.95</td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>17</td>
<td>20</td>
<td>1.19</td>
</tr>
<tr>
<td></td>
<td>B1</td>
<td>11</td>
<td>14</td>
<td>1.3</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>18</td>
<td>21</td>
<td>1.19</td>
</tr>
<tr>
<td></td>
<td>B3</td>
<td>20</td>
<td>26</td>
<td>1.31</td>
</tr>
<tr>
<td></td>
<td>B4</td>
<td>20</td>
<td>34</td>
<td>1.71</td>
</tr>
<tr>
<td></td>
<td>C1</td>
<td>17</td>
<td>19</td>
<td>1.12</td>
</tr>
<tr>
<td></td>
<td>C2</td>
<td>17</td>
<td>18</td>
<td>1.1</td>
</tr>
<tr>
<td></td>
<td>D1</td>
<td>13</td>
<td>13</td>
<td>0.94</td>
</tr>
<tr>
<td></td>
<td>D2</td>
<td>10</td>
<td>10</td>
<td>1.04</td>
</tr>
<tr>
<td></td>
<td>F1</td>
<td>16</td>
<td>15</td>
<td>0.94</td>
</tr>
<tr>
<td></td>
<td>F2</td>
<td>17</td>
<td>18</td>
<td>1.08</td>
</tr>
<tr>
<td></td>
<td>F3</td>
<td>16</td>
<td>23</td>
<td>1.38</td>
</tr>
<tr>
<td></td>
<td>F4</td>
<td>16</td>
<td>29</td>
<td>1.8</td>
</tr>
</tbody>
</table>
### Table 7.14: Calculation of $V_\delta$ and $b$ for SCS panels failing in bending

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Test</th>
<th>$r_{ti}$</th>
<th>$r_{ei}$</th>
<th>$r_{ei}/r_{ti}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>184</td>
<td>F5</td>
<td>17</td>
<td>19</td>
<td>1.15</td>
</tr>
<tr>
<td></td>
<td>SP1-2</td>
<td>210</td>
<td>187</td>
<td>0.89</td>
</tr>
<tr>
<td></td>
<td>SP1-5</td>
<td>2540</td>
<td>2585</td>
<td>1.02</td>
</tr>
<tr>
<td></td>
<td>SP2a-1</td>
<td>5300</td>
<td>5460</td>
<td>1.03</td>
</tr>
<tr>
<td></td>
<td>SP2a-2</td>
<td>5300</td>
<td>5703</td>
<td>1.08</td>
</tr>
<tr>
<td></td>
<td>SP2a-3</td>
<td>3786</td>
<td>4394</td>
<td>1.16</td>
</tr>
<tr>
<td></td>
<td>SP2c-3*</td>
<td>2266</td>
<td>2464</td>
<td>1.09</td>
</tr>
<tr>
<td></td>
<td>SP2c-4*</td>
<td>2266</td>
<td>2403</td>
<td>1.06</td>
</tr>
<tr>
<td>82</td>
<td>B-4R-2S400-4ST*</td>
<td>718</td>
<td>998</td>
<td>1.39</td>
</tr>
<tr>
<td></td>
<td>S-4R-2S400-4ST*</td>
<td>718</td>
<td>822</td>
<td>1.14</td>
</tr>
<tr>
<td>98</td>
<td>SP1</td>
<td>2940</td>
<td>2737</td>
<td>0.93</td>
</tr>
<tr>
<td></td>
<td>SP2</td>
<td>2903</td>
<td>3027</td>
<td>1.04</td>
</tr>
<tr>
<td>151</td>
<td>B9</td>
<td>35</td>
<td>68</td>
<td>1.95</td>
</tr>
<tr>
<td></td>
<td>B10</td>
<td>84</td>
<td>95</td>
<td>1.13</td>
</tr>
<tr>
<td></td>
<td>B11</td>
<td>47</td>
<td>64</td>
<td>1.35</td>
</tr>
<tr>
<td>64</td>
<td>BS4*</td>
<td>200</td>
<td>214</td>
<td>1.07</td>
</tr>
<tr>
<td></td>
<td>BS7*</td>
<td>203</td>
<td>242</td>
<td>1.19</td>
</tr>
<tr>
<td></td>
<td>BS10*</td>
<td>168</td>
<td>246</td>
<td>1.46</td>
</tr>
<tr>
<td></td>
<td>BS13*</td>
<td>550</td>
<td>601</td>
<td>1.09</td>
</tr>
</tbody>
</table>

The following values represent the model uncertainty:

$V_\delta = 18.85\%$

$b = 1.06$

#### 7.7.2 Effect of variability of the basic variables

The net variability of the basic variables is defined by comparison to a set of representative cases. In comparison to the out-of-plane shear case, more variables influence the bending behaviour. Through examination of the resistance model, the following variables are used to characterise a bending case:
CHAPTER 7: CALIBRATION OF EUROCODE PARTIAL RESISTANCE FACTORS

\[ X_n = \left( f_c \, f_y \, DOSC \, P_r \, h \, b \right) \]  

(7.7.1)

Where:

- \( f_c \) is the strength of concrete
- \( f_y \) is the strength of steel
- \( DOSC \) is the degree of shear connection
- \( P_r \) is the resistance of an individual shear connector
- \( h \) is the height of the section
- \( b \) is the width of the section

A key decision in this case is the classification of the stud resistance as a single variable. As the resistance equations (4.1.10 and 4.1.11) show, stud resistance can be described as a function of the material properties \( f_y,stud \) and \( f_c \), both of which could be included in \( X_n \).

However, the results of testing show that the variability of the stud resistance is much higher than the variability of either the stud material or the concrete. This is because the behaviour in push tests shows that the failure mechanisms for shear connectors depends on interactions between the two materials, even if the design model suggests failure is governed by a single material. For this reason, the stud resistance is treated as a separate variable, with its own separate COV and over-strength. In accordance with the work by Smith and Couchman\[168]\, these values are taken as 15% and 1.3 respectively.

Once the basic variables are defined, the array may be filled in accordance with the guidelines presented as in Section 7.5.2. As discussed, existing SCS structures are not widely available, so the array cannot be filled by a survey of existing structures.

In the absence of better information, the representative sample for calibration of the bending model was designed using a reasonable projection of modelling trends, based on experience and recent testing. All cases included tie-bars, at spacings lower than the limits presented in the SCIENCE design guide\[4\]. 30% of the cases were assumed to have degrees of shear connection less than 100%. 10% of designs were assumed to have asymmetric plate thicknesses.

The array was filled with 40 representative cases i.e. \( n_q = 40 \). For ease of presentation, only the first 5 rows of each array are presented.
Chapter 7: Calibration of Eurocode partial resistance factors

\[ X_{n,i,q} = \begin{pmatrix} 30 & 355 & 60\% & 170 & 500 & 500 \\ 30 & 355 & 70\% & 170 & 500 & 800 \\ 35 & 420 & 90\% & 170 & 500 & 600 \\ 30 & 420 & 110\% & 170 & 500 & 500 \\ 40 & 355 & 120\% & 170 & 700 & 1000 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \] (7.7.2)

The mean value array is defined as follows:

\[ X_{m,i,q} = \begin{pmatrix} 38 & 408 & 60\% & 196 & 500 & 500 \\ 38 & 408 & 70\% & 196 & 500 & 800 \\ 43 & 483 & 90\% & 196 & 500 & 600 \\ 38 & 483 & 110\% & 196 & 500 & 500 \\ 48 & 408 & 120\% & 196 & 700 & 1000 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \] (7.7.3)

As per the example presented in Section 7.4, geometrical variability is included as a single coefficient of variation of 2%, with the individual variabilities reduced to 0% in the array. Therefore:

\[ V_{X,i,q} = \begin{pmatrix} 4.87 & \frac{f_{c,k}}{f_{c,k} + 8} & 5\% & 0\% & 15\% & 0\% & 0\% \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \] (7.7.4)

Using the arrays defined above, a value of \( V_{rl} \) may be defined for each row. A weighting factor for each variable is calculated in accordance with Equation 7.5.19. For bending, this leaves the following equation:
\[
V_{rc,q}^2 = (w_{f,c} \times V_{f,c})^2 \\
+ (w_{f,bottom,c} \times V_{f,bottom,c})^2 \\
+ (w_{f,top,c} \times V_{f,top,c})^2 \\
+ (w_{DOSC,c} \times V_{DOSC,c})^2 \\
+ (w_{P,bottom,c} \times V_{P,bottom,c})^2 \\
+ (w_{P,top,c} \times V_{P,top,c})^2 \\
+ (w_{t, P} \times V_{t, P})^2 \\
+ (w_{b,c} \times V_{b,c})^2 \\
+ (w_{w,c} \times V_{w,c})^2
\]

(7.7.5)

A plot of \(V_{rt}\) for each of the cases shows a somewhat surprising trend; despite the fact that the equations contain many variables, the final value of \(V_{rt}\) shows only two fixed values, with a discontinuity appearing at 100% shear connection. This is shown in Figure 7.8.
This result can be explained by analysis of the bending design model. As first discussed in Section 4.1, panels that are specified with equal plate thicknesses will tend to fail in the tension plate, since the compression stress block can be shared between the tension plate and the concrete. The only conceivable design that would produce concrete failure, and therefore be affected by the variability of concrete, is one where the compression plate is neglected due to an large shear connector spacing producing compression plate buckling. However, this failure mode is specifically excluded by the latest version of the SCIENCE Design Manual\cite{4}.

Assuming tension plate failure is critical for the majority of designs, it is reasonable that two distinct regions exist. For panels with a shear connection ratio less than 100% the variability of the resistance is governed almost entirely by the variability of the failure load of the shear connectors, with geometrical variation adding the remaining uncertainty. The yield strength of the steel plate is irrelevant, since the plate will not reach yield at failure. Conversely, cases with a degree of shear connection greater than 100% are subject only to the variability of the steel plate, with the variability of the shear connector resistances proving to be irrelevant.

The plot of $b_{rl,q}$ also shows this trend. When the shear connection becomes critical the over-strength of the shear connectors determines the overall over-strength, while the steel over-strength governs when the shear connection is greater than 100%. Since the over-strength of the stud resistance (1.3) is assumed higher than the over-strength of
steel (1.15), the over-strength drops at degrees of shear connection greater than 100%. This is shown in Figure 7.9.

![Figure 7.9: Values of $b_{r,l,q}$ calculated for representative cases of SCS panels failing in bending](image)

**7.7.3 Calculation of $\gamma_M^*$**

To develop the final value of $\gamma_M^*$, a value of $\gamma_M^*$ is calculated for each of the representative cases.

$$X_{n,i,q} = \begin{pmatrix} 30 & 355 & 60\% & 170 & 500 & 500 \\ 30 & 355 & 70\% & 170 & 500 & 800 \\ 35 & 420 & 90\% & 170 & 500 & 600 \\ 30 & 420 & 110\% & 170 & 500 & 500 \\ 40 & 275 & 120\% & 170 & 700 & 1000 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

\[ E_{q.7.5.9} \]

$$\gamma_M^* = \sum \begin{pmatrix} 1.58 \\ 1.58 \\ 1.58 \\ 1.45 \\ 1.45 \\ \vdots \end{pmatrix} / n_q = 1.49$$

311
The values found in the array of $\gamma_{M}^{*}$ reflect the underlying value of $V_{rt}$, as shown in Figure 7.8. Those designs (i.e. lines of the array) that have a degree of shear connection less than 100% have a $\gamma_{M}^{*} = 1.58$, while those with greater than 100% shear connection have a $\gamma_{M}^{*} = 1.45$. Reflecting the proportions of the representative sample, 30% of the values are equal to 1.58, and 70% are equal to 1.45.

7.7.4 Comparison between $\gamma_{M}^{*}$ for SCS construction and the $\gamma_{M}^{*}$ for conventional composite construction

The value calculated is relatively large for the design of a composite system. In conventional composite design to Eurocode 4\[29\] there is no additional partial factor on bending resistance. However, the results are not directly comparable.

In simplified form, the resistance of composite structures (SCS and conventional) can be expressed as a function of concrete strength, steel strength, shear connector resistance and other dimensional parameters. This relationship is shown in Equation 7.7.8, in the form originally presented in Equation 7.3.24.

$$R_{d} = \frac{1}{\gamma_{M}^{*}}R\{f_{y,n};f_{c,k};P_{Rk};a_{d}\}$$ (7.7.8)

The $\gamma_{M}^{*}$ presented in Equation 7.7.7 is developed for use in an expression in this form. Eurocode 4 recognises that the shear connector resistance is a considerable source of variation in the resistance. Uncertainty is therefore allocated to shear connectors through a partial factor.

$$R_{d} = \frac{1}{\gamma_{M}^{*}}R\left\{f_{y,n};f_{c,k};\frac{P_{Rk}}{\gamma_{V}};a_{d}\right\}$$ (7.7.9)

Where:

$\gamma_{V}$ is the partial factor for design shear resistance of a headed stud, taken as 1.25\[29\]

A partial factor that is directly comparable to the one required by Eurocode 4 for conventional composite construction can be developed by the same method presented in Section 7.7. The only change required is to apply the partial factor $\gamma_{V}$ to the stud resistances when calculating the nominal resistance $g_{n} \left\{X_{n,q}\right\}$, as required in
Equation 7.5.7.

The result of this second analysis is $\gamma_{M^*} = 1.33$. This can be compared to the implied value of $\gamma_{M^*} = 1.0$ used in conventional composite construction.

It can reasonably be expected that the resistance of SCS panels is less certain than conventional composite construction, and therefore requires a greater partial factor. Conventional composite beams develop their composite action only in compression, which the results presented in Chapter 4 suggest is a relatively predictable phenomenon. SCS panels develop composite action in tension, which is affected by the presence of cracking and discontinuities in the force distribution.

The variation of the overall resistance function, represented through the parameters $b$ and $V_\delta$, appears to be more critical to final value of the partial factor than the variation of the basic variables, represented through the factors $b_{rt,\lambda}$ and $V_{rt,\lambda}$. This suggests there may be scope to improve the partial factor through improvements to the resistance function that more precisely predict the behaviour.

Another possible route to improvement of the partial factor is further testing, perhaps combined with additional filtering of the test results. As shown in Table 7.14, the test population includes both old tests (e.g. Oduyemi and Wright\textsuperscript{[131]} and Roberts et al.\textsuperscript{[151]}) and new tests (Koukkari and Fülöp\textsuperscript{[98]}), and combines model scale and full scale tests. It may be possible to improve $b$ and $V_\delta$ through exclusion of any test that does not conform with more modern designs, such as those tests that do not contain tie-bars. However, the test population should not be filtered too much, since $\gamma_{M^*}$ is affected by the number of tests included in the population, through the $k_{d,n}$ factor included in Equation 7.5.9.

### 7.8 Conclusions

This chapter has presented a Eurocode based method for developing partial factors for use in design from a set of test results. A review of relevant literature, as described in Section 7.1, has shown the method has been applied numerous times to steel structures, but only sparingly to concrete or composite. In applying the methods to the SCS system a number of complexities are exposed, as discussed in Section 7.5. To aid with presentation, a new way of presenting the method that utilises matrices was developed, as described in Section 7.3.

The method as described is applied to the out-of-plane shear resistance of SCS panels. The results of this analysis are shown in Section 7.4. The results show relatively high
partial factors, compared to those presented in the Eurocode for reinforced concrete design. This is particularly the case for the bending resistance, which relies entirely on the shear connection to develop resistance.

The results highlight the need for sub-division of the population of test results, since inclusion of tests that fail via alternative failure mechanisms tend to lead to high scatter. The small number of tests available for SCS panels means that this sub-division is not always possible, highlighting a need for increased testing.
Conclusion & further work

This thesis has examined the behaviour and failure mechanisms of steel-concrete-steel (SCS) sandwich panels when subject to out-of-plane loads. The key focus of the investigation has been changes in behaviour that can occur due to the provision of lower degrees of shear connection between the plates and the concrete core, and how these changes might affect design. This chapter presents a summary of this investigation, its novel contributions, and highlights areas where further research would be beneficial.

8.1 Summary

Aside from the background work described in Chapter 2, the first key process towards improving understanding of the behaviour of SCS panels subject to out-of-plane loads was a detailed review of the available test evidence. Chapter 3 describes the construction of a detailed database, containing all of the numerous parameters required to describe a test. In addition to the data gathering exercise, programming techniques were developed to allow the cases to be parametrically constructed in a number of FE programs, increasing the speed and accuracy of the modelling process. Many of the cases found in the literature were incompletely described, meaning the test had to be discarded from use in further study.

Once the detailed database was assembled, existing models for the resistance of the panels to out-of-plane failure could be tested against the test database. The three key design checks for panels subject to out-of-plane loads are bending failure, shear failure and excessive deflection. A separate chapter is dedicated to each check.

Chapter 4 begins by examining existing models for bending resistance. Existing models for bending resistance of SCS panels are based on plastic analysis of a cross-section,
and are relatively straight-forward to apply to beams with degrees of shear connection greater than 100%. However, cases where the degree of shear connection is less than 100% present more of a challenge, as the designer must decide the portion of the span over which the shear connection is mobilised.

The test database was found to be lacking in sufficient cases where the degree of shear connection is low. A detailed FE model was developed to supplement the database, allowing for parametric study of cases that had not been investigated to be numerically calculated. The verification exercise suggested that strong correlation was achieved between the model and the test results.

A considerable advantage of the FE model over traditional testing is the possibility to measure stresses and strains in more detail and at a greater number of locations than would be practically or financially achievable in a test. A key example of this is the ability of the FE model to directly output the force in an individual shear connector, which is not possible in a test. As a result of this, profiles of stud force distribution could be constructed. Figure 4.42 shows an example of a case where abrupt discontinues in the stud force profile are found, due to the presence of flexural cracks.

Existing predictions of a sinusoidal distribution for the connector force distributions, as given by Newmark et al. \textsuperscript{[126]}, were found not to be consistent with the distributions obtained from the FE model, for the plate subject to tension. A further mechanical interpretation based on these results suggested that the misunderstanding of the force distribution on the tension face of the panel could lead to unconservative predictions of the bending resistance, when the panel is subject to a UDL and has a low degree of shear connection. No test showed this behaviour, because no test had been conducted for a panel with a low degree of shear connection, subject to a UDL. A new design rule for this situation is suggested. A test by Roberts et al. \textsuperscript{[151]}, where a panel is subject to four point loads, shows the expected behaviour (as presented in Section 4.9.2). This is supplemented with additional FE evidence. Based on these models, a new design model is presented for panels subject to a UDL, which allows prediction of an appropriately conservative resistance.

Chapter 5 presents the investigation of shear resistance. Shear resistance was investigated in a similar manner to the bending resistance. However, shear failure is generally understood to be a much more complex phenomena than bending failure, meaning there is considerably more variation between design models from different countries. To understand this variation, a large number of design models were applied to cases in the test database that failed in shear.

As with bending resistance, the test database included insufficient tests on panels with
low degrees of shear connection. The test database was therefore supplemented with a large parametric FE study. This FE model was exactly the same one used for the bending study, without additional modifications. Comparisons between the model and the available test data are very strong, as presented in Section 5.4.1.

The study encompassed panels both unreinforced and reinforced in shear. The critical effect of the $a/d$ ratio was also investigated.

The results of the FE study show that models for failure of conventional reinforced beams work well for SCS panels. Enhancements due to arching action were found to be large, though no method in any of the existing design codes was found to accurately account for it in all cases.

The best comparisons between the tests and design model were found for the *fib* Model Code 2010\[61\]. This model is based on international research from a number of groups, and includes iterative models to properly assess the strain present in the cross-section. This added complexity appears to improve predictions. The *fib* Model Code 2010 is recommended as the basis of future design guidance for SCS panels.

The degree of shear connection was found to influence the resistance, for both panels reinforced and unreinforced in shear. Resistance is not necessarily proportional to degree of shear connection, since increased shear connection can prompt a shift in ‘Kani’s valley’, which can reduce resistance, as explored in 5.5.3. Both the Eurocode and *fib* models were found to produce unconservative resistances for panels with very low degrees of shear connection. An amendment, based on a reduced effective reinforcement ratio, is suggested for both models. In both cases the predictions improve.

Design models from South Korea and Japan for shear failure were found to give unconservative results in many cases. In some cases the over-prediction of the resistance was found to be significant. This over-prediction is unlikely to be a safety issue, since both codes have considerably limited the allowable stresses of the materials. It is recognised that the comparisons in these cases are based on translated versions of the design codes, which may be incomplete. It is recommended that these results require further research.

**Chapter 6** looks at deflections. Prediction of deflection/stiffness is important both for serviceability, and due to its implications on the distribution of forces within the structure. Prediction of deflection is normally achieved by assuming linear elastic properties, with either the steel contributing stiffness only or the concrete being additionally effective. A model developed by Varma\[22,183,186\] for the ASIC design
guide allows for a partial contribution of the concrete. This model was found to be most suitable for the majority of cases in the database.

Examination of some specific cases in the test database suggested that use of shell finite elements that only incorporate bending deformation (i.e. are based on the Bernoulli bending equation) can overestimate stiffness when shear deformations are significant. This issue can be overcome through the use of more sophisticated elements that contain a separate term for shear deformation. These elements implement the well known work of Timoshenko\(^{[178]}\). A method presented by Kim and Mander for the prediction of shear stiffness of cracked reinforced concrete structures is shown to be a good predictor of the shear stiffness of an SCS panel, when incorporated in a Timoshenko element.

Shear connection stiffness was found to significantly affect the deflection response of panels to load. Linear elastic analysis of panels with low degrees of shear connection was found to be not feasible, since the overall panel deflection reflects the load-slip response of connectors, which becomes non-linear early in its load-deformation relationship (See Figure 4.22). A new empirical equation is presented to allow for degradation in stiffness with load, with the degree of shear connection being an input parameter. This new model is shown to predict the stiffness of a number of test cases well.

Chapter 7 presents the derivation of Eurocode compatible partial factors for the shear and bending resistance models. These factors are essential for adoption of the models in design in Europe. The partial factors are calculated to take into account uncertainty in the resistance formulation and the materials from which the system is constructed.

A procedure for calculating these partial factors is presented in Eurocode 0 Annex D\(^{[31]}\). This method is readily applied to relatively simple resistance functions, where the resistance is proportional to the strength of the underlying material. However, difficulty is found when the method is extended to cases where there is an interaction of materials, such as the bending resistance model, which includes contributions from the steel and the concrete. A new method for calculating the partial factors is proposed, called the ‘matrix method’. The new method can be readily applied to any resistance function.

The partial factors derived for bending and shear failure of members with shear reinforcement are higher than those currently included in Eurocode 2\(^{[30]}\), for conventional reinforced concrete design. A number of reasons are suggested for this, though it is most likely that the method is sensitive to ‘out-lying’ predictions in the existing test database, especially since the number of tests is limited. It is suggested
that these partial factors may be reduced when more testing is conducted.

In summary, this thesis has presented a comprehensive exploration of the effects of out-of-plane loads on the resistance of SCS panels. Supported by FE modelling, the degree of shear connection was found to effect all of the potential failure modes, with varying degrees of significance.

It is proposed that designs that utilise low degrees of shear connection may now be readily specified by designers. However, it should be understood that changes in the behaviour of the structure can be expected, which will have to be accounted for in the design process. New design rules are presented for each of the failure modes, that take into account the degree of shear connection.

The economy of the SCS system is linked to the number of shear connectors that must be fixed to the plates, as discussed in Section 2.1. In some cases more shear connectors will be needed to satisfy the new design rules than would have been the case before this thesis, but for most designs this will not be the case. No significant decrease in the economy of the system is therefore expected as a result of the application of these new rules.

8.2 Research significance and contributions

Through the process of developing the content for this thesis, a number of areas where existing understanding is lacking were found. Increased clarity was bought to these areas through an extensive literature search, understanding of the mechanical behaviour leading up to failure, detailed modelling using finite element analysis, and application of engineering and statistical modelling techniques.

The following list of topics are proposed as novel and significant contributions to the understanding of SCS panel behaviour:

1. Finite element analysis has been used to extend knowledge of panel behaviour to designs that have not been tested, particularly with regards to low degrees of shear connection. The proposed novel modelling technique of using non-linear springs to represent the load-deformation response of individual shear connectors was found to give accurate predictions of the behaviour, but particularly with regard to predicting end-slip.

2. The suitability of 2D modelling has been demonstrated for a number of the mechanisms, including out-of-plane shear and bending. This is unique among researchers in SCS panel technology, who have exclusively employed 3D
modelling. It is expected that the techniques developed will be equally as applicable to other 2D failure mechanisms, such as member compression or connection failure.

3. Existing design rules for calculating bending resistance of an SCS panel requires the assessment of it’s bending resistance at the point of mid-span. This has been demonstrated to give un-safe predictions resistance for beams subject to a UDL with partial shear connection, through application of both FE and a first principles analysis to establish cross-section equilibrium. New rules are presented for assessing the critical cross-section when the beam is subject to a UDL, which is a plausible design case when considering wind and blast actions.

4. Eurocode design rules for calculating resistance of concrete beams to shear are demonstrated to work for SCS panels. However, unconservative results are found when the degree of shear connection is low (usually less than 60%). Analysis of FE results suggest that this effect can be attributed to the greater propensity of cracks to open when not prevented by the plate. A new adjustment factor for the Eurocode model is presented, such that conservative results are calculated for all designs.

5. Current design equations for predicting deflection of SCS panels tend to be based on Euler–Bernoulli beam theory, which only takes into account bending deformation. In many cases, but particularly those with a low shear-span to depth ratio, shear deformation is more critical, meaning stiffness is often predicted unconservatively. New equations are presented for accurately assessing stiffness. A new model is presented to allow for degradation of stiffness with load in non-linear analysis. This is particularly critical for cases with low degrees of shear connection, since the response of these structures is non-linear even at low load levels.

6. In spite of the ubiquity of composite construction in construction in the UK, few publications have demonstrated the application Eurocode method for calibrating partial factors (Eurocode 0 Annex D[33]) to composite structures. Application of the existing equations proved to be problematic for failure modes where the steel and concrete each provided a component of resistance, with the proportions of resistance coming from each being a function of the panel geometry. It is eventually demonstrated that expressing the functions of the method though a new ‘matrix representation’ allows for the non-linear nature of the resistance function. The method also allows for the definition of multiple
representative structures, which can represent the likelihood of designs being realised in practice.

The ‘matrix method’ is of general interest to researchers and code writers, even outside of composite structures.

8.3 Published work and project reports

Work on this thesis has generated a number of research outcomes and results that are suitable for further publication.

The work presented in this thesis was undertaken in parallel with the SCIENCE project, funded by the ‘Research Fund for Coal and Steel (RFCS)’[^4]. As part of this project a number of work packages were defined, each of which required a deliverable. The author of this thesis either wholly prepared or contributed to a number of these deliverables, as presented in Section 8.3.2. These deliverables were reviewed and commented on by the project team.

Work produced as part of this project has also generated content considered suitable for publication in journals. Three possible publications are suggested below in Section 8.3.1. Of those three, one has been accepted for publication, one is expected to be submitted imminently, and one is currently being prepared.

8.3.1 Peer reviewed papers

The following papers have been prepared in relation to the work presented in this thesis:

Sagaseta and Francis[^152] - Out-of-plane shear strength of steel-concrete sandwich panels

A paper was submitted for the ‘fib International Workshop on Beam Shear’ in Zurich (Sept 2016). The author of this thesis contributed data collection, comparison between the design models and the test data, and interpretation of the behaviour of tests.

Chapter 5 of this thesis elaborates and extends the contributions made to this paper. The FE study presented in Sections 5.5 and 5.6 supports some of the conclusions that were drawn in the paper[^152] only from the available test data. The FE study supports the conclusions of this paper in all respects. Of most note is the detection of a shift in the position of ‘Kani’s valley’ with changes in the degree of shear connection, which can
lead to the counter-intuitive loss of resistance with increased shear connection stiffness; this is discussed in Section 5.5.3 of this thesis. The paper also explores the classification of the test results by shear-span to depth ratio \((a/d)\), which impacts the selection of cases for the reliability work, as presented in Chapter 7.

**Francis et al. - Development of Eurocode compatible models for shear resistance of steel-concrete sandwich panels**

A paper has been prepared concerning the behaviour of SCS panels in shear. The paper includes work from a number of areas of this thesis, including the test database (Chapter 3), out-of-plane shear behaviour (Chapter 5) and the calculation of partial factors (Chapter 7).

Submission of this paper has been delayed by the need for a final version of the design rules developed as part of the SCIENCE project. This time has also allowed refinement of the cases included in the test database (Chapter 3). Submission is expected imminently.

**Francis et al. - Design of steel-concrete-steel sandwich panel connections subject to out-of-plane actions**

The connection testing and design rule development undertaken as part of the SCIENCE project is expected to published in the near future. A paper has been planned that describes both the testing and development of design rules, as described in SCIENCE Deliverables D7.2/7.6\(^{[68]}\) and D4.2\(^{[120]}\).

The parametric study performed by the author of this thesis is expected to be included as supporting evidence for the new design rules. The FE model used for this study is exactly the same one used in this thesis, as described in Section 4.4, and presented in Appendix A. The paper will also include significant discussion on the bending and shear models presented in Chapters 4 and 5 of this thesis, since back analysis of the connections must ensure that the member failure is not observed before connection failure.

The co-authors of this paper will include researchers from a number of the SCIENCE partners, including the authors of deliverables D4.2\(^{[120]}\) and D4.4\(^{[122]}\). These authors will contribute a description of the testing and additional, corroborating, FE analysis.
8.3.2 SCIENCE project deliverables

This thesis was prepared in parallel with the SCIENCE RFCS project. The following deliverables overlap with the contents of this thesis, and were wholly or partially contributed by the author:

- Deliverable D7.2/7.6 - Parametric study and design methods for connections\textsuperscript{[68]}
- Deliverable D2.5/2.6 - Derivation of partial safety factors for limit states of SC structures based on the Eurocode Annex D procedure\textsuperscript{[45]}
- Deliverable D7.9 - Effective properties for use in the analysis of SC structures\textsuperscript{[67]}
- Deliverable D8.1b - Stresses in the plate due to fresh concrete pressures\textsuperscript{[65]}
- Deliverable D9.1\textsuperscript{[2]} / P414\textsuperscript{[4]} - Design of steel concrete composite (SC) structures

8.4 Further research

Throughout the course of this research a number of areas for further research. In a number of cases these could not be pursued, due to limitations on time and lack of a budget or facilities for testing. This section discusses these areas of possible research, with the aim of inspiring work by other researchers in this area.

8.4.1 Verification through beam testing

Changes to the degree of shear connection has been found to produce changes to all of the relevant performance criteria for an SCS panel subject to out-of-plane forces. In most cases the evidence for these conclusions is based mostly (but not entirely) on the results of finite element modelling, since the test database lack sufficient tests on panels with low degrees of shear connection.

While the finite element model shows very good predictions for the cases that are available for verification, the results for any further cases will remain theoretical. In particular, no test has ever been conducted on a genuine uniformly distributed load (UDL). The results presented in Section 4.9 of this work suggest a four-point bending test may have a significantly higher bending resistance than a test subject to a UDL, due to differences in the location of the critical cross-section.
8.4.2 Additional testing

In addition to the UDL testing, any additional testing will be helpful to the analysis. The partial factors calculated as in Chapter 7 have shown sensitivity to the sample, which for the out-of-plane shear design check is relatively small (see Section 7.4). Additional testing may help in the calculation of less conservative partial factors.

Additional measurements in the test program may further enhance the partial factor analysis. A key assumption of this work is that all geometric variation is covered by a 2% coefficient of variation. This assumption was required because detailed coefficients of variation for the geometrical variation of each of the component parts of the system was not available. Detailed measurement of geometrical parameters of a constructed SCS system, including plate thicknesses and as-built tolerances, would aid the reliability analysis.

8.4.3 Push testing of tie-bars

As first presented in Section 4.5, a simplified model has been developed for the shear connectors, relying on the use of a non-linear spring to model the load-slip behaviour. This spring stiffness has been shown to work well for studs embedded in a solid concrete slab, as shown by a number of researchers (See Section 4.6). However, it cannot be known with certainty that the same load-slip relationship also applies for tie-bars, based on the current test evidence.

Parametric studies in this work and by others have shown that the shape of the load-slip curve at the refined scale does not significantly affect the results from a large scale model, in terms of overall bending resistance and end-slips. The limited test evidence that is available (by Clubley et al.[48], as discussed in Section 4.6.4) supports the hypothesis that load-slip curves measured for tie bars are consistent with those measured for shear studs.

While this conclusion is adequate for this work, extra push-test evidence would provide welcome additional certitude to the overall conclusions of this study.

8.4.4 Co-existent tension and shear in tie-bars

While push testing of tie-bars would provide helpful clarity on the performance of tie-bars in shear, it can also be recognised that the standard push-test may not accurately represent the true stress state in a tie-bar. When tie-bars are utilised in SCS panels they are usually expected to carry tensile forces, which may be developed through resisting
out-of-plane shear or may be locked-in at the construction stage. Locked-in stresses are discussed in more detail in Section 2.7.2.

Given the above, testing of tie-bars in a combined state of shear and tension would provide useful clarity for SCS panel design. However, this form of testing is not easily achieved in practice, as there is no simple means to apply additional pressure to the inside of the plates at the time of casting.

A possible alternative to testing is the use of a 3D finite element model. FE modelling has been shown to produce results with high levels of precision in the modelling of shear studs used in conventional composite construction. Examples of recent studies where FE analysis has been successfully employed to model push-tests include works by Qureshi et al. [141], Guezouli and Lachal [75] and Bouchair et al. [23]. The key advantage of FE in this context is that an internal pressure can be readily applied to the internal face of the plates, to represent the concrete pressure.

8.4.5 Application of the same techniques to in-plane shear

The final area where work is recommended to be undertaken concerns the resistance of panels to in-plane forces, as opposed to the out-of-plane forces explored as part of this thesis.

In-plane forces arise in many load cases for buildings utilising SCS panels. In-plane forces are particularly large in design for earthquakes, which is critical design case for all nuclear power plants, including those in non-seismic areas [159].

In-plane resistance is considerably harder to test than out-of-plane resistance. The load arrangement to obtain a state of pure shear is considerably more complicated to implement than the arrangement for an out-of-plane test. The forces required to produce failure for a reasonably sized panel are also much larger than the forces that can fail a panel out-of-plane, and are beyond the capability of most laboratories in the world.

The numerical modelling techniques described in this thesis can be applied to SCS panels subject to in-plane forces, including the novel technique of modelling shear connectors using springs (as described in Section 4.5). However, a 3D analysis is required. Researchers should be aware that the computation time associated with a 3D model may make meshes as fine as the ones described in Section 4.4.2 infeasible. Attention should be paid to the artificial strain energy for less fine meshes.

It is expected that changes in behaviour as a result of reduced degrees of shear connection are likely to be found for in-plane forces, as they are for out-of-plane
forces. The combined effects of in-plane and out-of-plane forces are an area worthy of further study.
Bibliography


[9] American National Standards Institute and American Institute of Steel


[43] B. J. Choi, K. Kim, C.-h. Kim, and T. Y. Kim. Experimental compression behaviour of Stiffened Steel-Plate Concrete (SSC) structures under compressive loading. In
BIBLIOGRAPHY

20th International Conference on Structural Mechanics in Reactor Technology (SMiRT 20), pages 94–96, 2008.


[112] B. Mckinley. Large Deformation Structural Performance of Double Skin Composite

Journal of Constructional Steel Research, 58(10):1347–1359, 2002. ISSN 0143974X.
science/article/pii/S0143974X02000159.

[114] R. E. Melchers and A. T. Beck. Structural reliability analysis and prediction. 3rd


[116] E. Miller and J. Strain. Introduction to the ANSYS Parametric Design Language

Investigation on Impact Resistance of Steel Plate Reinforced Concrete Barriers
Against Aircraft Impact Part 1 : Test Program and Results. 18th International
Conference on Structure Mechanics in Reactor Technology (SMiRT 18), (SMiRT 18):

1680/iicep.1991.17485. URL http://wrap.warwick.ac.uk/22262/\{\}5c

design of SSC wall in fire. In Transactions of the 20th International Conference
on Structural Mechanics in Reactor Technology (SMiRT 20), pages 177–179, Espoo,
Finland, 2009.

[120] M. M. Müller, N. Herrmann, and H. S. Müller. SC for Industrial, Energy and
Nuclear Construction Efficiency (SCIENCE): D4.2 - Testing of SC connections.


Out-of-plane behavior and design of steel-plate composite (sc) walls for 


Steel-plate composite walls: Experimental 
database and design for out-of-plane shear. 


Behaviour of Double Skin Composite Slabs - An Experimental Study. International Journal of Steel Structures, 2005.


[163] Simulia.  


[164] Simulia.  


[165] Simulia.  

6.3.3 Computational cost Explicit dynamic analysis. In ABAQUS 6.14 Theory Guide. 2015.

[166] Simulia.  


Appendices
Python code for modelling an SCS beam using ABAQUS

This appendix presents the python code used to generate the models used for the parametric studies presented in this thesis. The model implements the approach described in Section 4.4.

```python
import math
class AssemblyClass():
    from abaqus import *
    from abaqusConstants import *
    from part import *
    from material import *
    from section import *
    from assembly import *
    from step import *
    from interaction import *
    from load import *
    from mesh import *
    from job import *
    from sketch import *
    from visualization import *
    from connectorBehavior import *
    import regionToolset
    import connectorBehavior
    import time
    import math

def BuildModel(self, runModeBoolean):
    import sys
    self.version = 'NA'
    if sys.argv[0].find('6.14-3') > -1:
        self.version = '6.14-3'

    self.ModelNo = 83
    # self.Case_Name = 'BeamTest-' + str(self.ModelNo)
    self.Case_Name = 'Varma_SPI-1'

    self.Case_Settings = ''
    # self.Case_Settings = self.Case_Settings + '_NoConfinement'

    self.Model_Name = 'SCIENCE_

    print 'Building Model'
    self.LoadParams()
```
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
self.CreateModel()
self.CreateMaterial()
self.CreateBeamPart()
self.CreateStudPart()
self.CreateStudAssembly()
self.CreatePlatesPart()
self.CreateSupportPart()
self.CreateAssembly()
self.CreateLoadsAndResults()
self.saveVariablesToODB()
selSelf.CreateJob(runModeBoolean)

def LoadParams(self):
    global regionToolset
    import regionToolset
    global connectorBehavior
    import connectorBehavior
    global mesh
    import mesh

    import math

    self.load1 = -100.0
    self.load1a = -100.0
    self.load1b = -100.0
    self.load1c = -100.0

    self.loadFactor1 = 1.0
    self.loadFactor1a = 1.0
    self.loadFactor1b = 1.0
    self.loadFactor1c = 1.0

    self.additionalSlipFactor = 1.0
    self.DeflectionControl = True
    self.appliedDeflection = 0.1
    #Default values – Don’t change
    
    #x=0.0 is centre of beam
    self.supportWidth = 400.0 / 1000.0
    self.supporth = 50.0 / 1000.0 * 3.0

    self.concreteModel = 2
    self.DeflectionControl = True

    self.connectorModel = 3.0 #3 rounded, 2 quad linear 3,4,5,6
    self.additionalSlipFactor = 1.0

    self.studPositions = []
    self.studPositionDescriptions = []

    self.IncludeGravity = False
    self.GravityTime = 0.01
    self.LoadTime = 10.0 #0.0 #3 mins per cycle
    self.Cycles = 0
    self.EndCycles = 1
    self.CycleIntervals = 2
    self.FinalLoadTime = 1500.0

    # if self.Case_Name[find('Varma')]=1:
    #
```

350
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
# self.studResistance = -1000.0
self.studNo = 0.0

# self.studSurround = self.studMeshScale*2.0 #0.05
# self.studDepth = self.studMeshScale #20.0/1000.0+3.0

self.studPositions.sort()
print self.studPositions

self.Explicit = True
self.widthScale = 1.0/self.beamWidth
self.finalStrength = 0.0

# FIB model code 2010 5.1–9
self.GF = 73.0+math.pow(self.concretefc,0.18) #N/m, fc in MPa
self.crackRatio = 0.2
self.ecu = 0.0025
self.tensionCapacityAdjustment = math.pow(self.concretefc,(2.0/3.0))*0.3/self.concretefc

# FIB model code 2010 5.1.5 – Tensile capacity upper bound
# self.tensionCapacityAdjustment = self.tensionCapacityAdjustment + 1.3
# self.GF = self.GF+2.0

if self.Case_Settings.find('NoConfinement') > -1:
    self.concreteCompressionEnhancement = 1.0 #1.3
    self.includeCrushing = True
elif self.Case_Settings.find('ExtraStrong1') > -1:
    self.concreteCompressionEnhancement = 1.5 #1.3
    self.GF = self.GF+1.2
    self.includeCrushing = False
elif self.Case_Settings.find('ExtraStrong2') > -1:
    self.concreteCompressionEnhancement = 1.3
    self.GF = self.GF+1.3
    self.tensionCapacityAdjustment = self.tensionCapacityAdjustment*1.3
    self.includeCrushing = False
elif self.Case_Settings.find('HalfTension') > -1:
    self.tensionCapacityAdjustment = self.tensionCapacityAdjustment*0.5
    self.concreteCompressionEnhancement = 1.3 #1.3
    self.includeCrushing = False
else:
    #Enhancement required for confinement
    self.concreteCompressionEnhancement = 1.3 #1.3
    self.includeCrush = False

self.conMassScale = 1.0 #1000.0
self.steelMassScale = 1.0 #1000.0
self.springMass = 1.0*2500.0+self.studMeshScale*#self.studTipW=1.0 #*10.0
self.springStudMass = self.springMass+1.0
self.springStudRotation = 1.0
self.springStudBaseMass = 1.0 #1000.0 #self.springMass+1.0
self.interfaceMassScale = 10.0 #1000.0 #1000.0
self.LoadMass = 0.001*#self.springStudBaseMass #*100.0
self.materialDampingAlpha = 0.05 #0.1
self.materialDampingBeta = 0.0

self.linearBulkViscosityStep = 0.02

def loadCase(self, caseName):
    Lines = []
    Areas = []
```

APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
CurrentDir = r"\UWS30089\Users\pf00062.SURREY\OneDrive\Surrey"
self.numCores = 4
if os.path.isdir(CurrentDir + '/') == False:
    CurrentDir = r"\pcv-pxf.sci.local\Users\PXF\OneDrive\Surrey"
if os.path.isdir(CurrentDir + '/') == False:
    CurrentDir = r"C:\\Users\\Phil\\OneDrive\\Surrey"
if os.path.isdir(CurrentDir + '/') == False:
    CurrentDir = r"C:\\Users\\pfran\\OneDrive\\Surrey"
self.CurrentDir = CurrentDir

if self.Case_Name.find('KIT') > -1:
    FilePath = self.CurrentDir + r"\\ABAQUS\\abaqus_plugins\\Case_Data\\Tests\\KIT1.py"
    if os.path.isfile(FilePath):
        execfile(FilePath)
    else:
        FilePath = r'Z:\\\ABAQUS\\abaqus_plugins\\Case_Data\\Tests\\KIT1.py'
        print FilePath
        if os.path.isfile(FilePath):
            execfile(FilePath)
        else:
            raise ValueError('Cant Load KIT builder')

elif self.Case_Name.find('Connections Parametric Study') > -1:
    FilePath = self.CurrentDir + r"\\ABAQUS\\abaqus_plugins\\Case_Data\\Connections Parametric Study\\Builder.py"
    if os.path.isfile(FilePath):
        execfile(FilePath)
    else:
        FilePath = r'Z:\\\ABAQUS\\abaqus_plugins\\Case_Data\\Connections Parametric Study\\Builder.py'
        print FilePath
        if os.path.isfile(FilePath):
            execfile(FilePath)
        else:
            raise ValueError('Cant Load CPS builder')
else:
    # Beam tests
    FilePath = self.CurrentDir + r"\\ABAQUS\\abaqus_plugins\\Case_Data\\Cases.py"
    FilePath2 = self.CurrentDir + r"\\ABAQUS\\abaqus_plugins\\Case_Data\\Cases_Builder.py"
    print FilePath
    if os.path.isfile(FilePath):
        execfile(FilePath)
        execfile(FilePath2)
    else:
        FilePath = r'Z:\\\ABAQUS\\abaqus_plugins\\Case_Data\\Cases.py'
        FilePath2 = r'Z:\\\ABAQUS\\abaqus_plugins\\Case_Data\\Cases_Builder.py'
        print FilePath
        if os.path.isfile(FilePath):
            execfile(FilePath)
            execfile(FilePath2)
        else:
            raise ValueError('Cant Load case builder')
self.beamWidth = self.beam
self.appliedDeflection = 0.1 # 0.1
self.plateMeshScale = 0.0035
self.GravityTime = 200.0
self.FinalLoadTime = 1000.0
self.IncludeGravity = True

def meshSizes(self):

```

352
if self.meshScale == 0:
    meshSizeOption = 6
if meshSizeOption == 1:
    self.meshScale = self.beamh/25.0
    self.studMeshScale = self.meshScale/5.0
    self.studLine = self.studMeshScale*7.0
elif meshSizeOption == 2:
    self.meshScale = self.beamh/30.0
    self.studMeshScale = self.meshScale/5.0
    self.studLine = self.studMeshScale*10.0
elif meshSizeOption == 3:
    self.meshScale = self.beamh/40.0
    self.studMeshScale = self.meshScale/2.0
elif meshSizeOption == 4:
    self.meshScale = self.beamh/50.0
    self.studMeshScale = self.meshScale/1.5
elif meshSizeOption == 5:
    self.meshScale = self.beamh/60.0
    self.studMeshScale = self.meshScale/1.3
elif meshSizeOption == 6:
    self.meshScale = self.beamh/40.0
    self.studMeshScale = self.meshScale
    self.studLine = self.studMeshScale*9.0
elif meshSizeOption == 7:
    self.meshScale = self.beamh/20.0
    self.studMeshScale = self.meshScale
elif meshSizeOption == 8:
    self.meshScale = self.beamh/60.0
    self.studMeshScale = self.meshScale
elif meshSizeOption == 9:
    self.meshScale = self.beamh/25.0
    self.studMeshScale = self.meshScale/1.0
    self.studLine = self.studMeshScale*5.0
elif meshSizeOption == 10:
    self.meshScale = 0.025
    self.studMeshScale = self.meshScale/2.0
    self.studLine = self.studMeshScale*5.0

def CreateModel(self):
    # session.journalOptions.setValues(replayGeometry=COORDINATE, recoverGeometry=COORDINATE)
    session.journalOptions.setValues(replayGeometry=INDEX, recoverGeometry=INDEX)
    # Create Job
    ClearAll = 'No'
    try:
        execfile('\UWS30089\Users\pf00062\SURREY\ABAQUS\Scripts\Standard_Routines\Clear.py')
    except:
        clearFail = 1

    # MyM = My Model
    # Declare Globals that will change here
    global mM
    global myPart
    mM = mdb.Model(name=self.Model_Name)
    self.mM = mM

    session.viewports[ 'Viewport : 1' ].partDisplay.meshOptions.setValues(meshTechnique=ON)

def CreateMaterial(self):
    import math

    #Materials
    for ConcreteNo in range(0, 2):
        if ConcreteNo == 0:
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
Mat = mjm.Material(description='Concrete', name='Concrete')
concfc = self.concretefc

if ConcreteNo == 1:
    Mat = mjm.Material(description='Strong Concrete', name='Strong_Concrete')
    concfc = self.concretefc * 1.0

Mat.Damping(alpha=self.materialDampingAlpha, beta=self.materialDampingBeta)

self.ConcreteStressStrainCompression = []
self.ConcreteStressStrainCompressionActual = []
self.ConcreteDamage = []
self.ConcreteStressStrainTensionActual = []
self.ConcreteStressStrainCrackActual = []
self.ConcreteStressCrackTension = []
self.TensionDamageCrack = []

if self.concreteModel == 1:
    Mat.ConcreteDamagedPlasticity(table=((38, 0.025, 1.16, 0.66667, 0.19),))
    Mat.Elastic(table=((33.0*1E9, 0.18),)) #GPa
    Mat.Density(table=((2500.0, ),))

elif self.concreteModel == 2:
    tensilestrength = 1E6 * concfc * self.tensionCapacityAdjustment #N/m^2

    #FIB model code 2010
    AlphaE = min(0.8 + 0.2 * concfc / 88.0, 1.0) #5.1−24
    #print AlphaE
    Ect = ((2.15 * math.pow(10.0, 4.0)) * (math.pow(concfc / 10.0, (1.0 / 3.0)))) * 1E6 * AlphaE #N/m^2
    Ect15 = tensilestrength / 0.00015
    #Straight line up to 0.00015
    print Ect
    print Ect15
    Ect = Ect15
    ecr = tensilestrength / Ect
    Mat.Elastic(table=((Ect, 0.18),)) #GPa
    Mat.Density(table=((2500.0 * self.concMassScale, ),))
    Mat.ConcreteDamagedPlasticity(table=((38, 0.1),))

    FibModel = True
    if FibModel == True:
        cstart = 0.0 #1.0* self.GF/tensilestrength*0.3

        #self.GF = 350.0 #self.GF*1.3
        #Just for plot
        self.ConcreteStressStrainTension = []
        self.ConcreteStressStrainTension.append((0.0, 0.0))
        self.ConcreteStressStrainTension.append((ecr, tensilestrength))

        self.ConcreteStressCrackTension.append((tensilestrength * 1.0, 0.0))
        self.ConcreteStressCrackTension.append((tensilestrength * 0.2, 1.0 * self.GF/tensilestrength))
        self.ConcreteStressCrackTension.append((tensilestrength * 0.01, 5.0 * self.GF/tensilestrength))
        self.ConcreteStressCrackTension.append((tensilestrength * 0.01, 10.0 * self.GF/tensilestrength))
        self.ConcreteStressCrackTension.append((tensilestrength * 0.01, 30.0 * self.GF/tensilestrength))

        #Temporary for interpolation
        self.ConcreteStressCrackTension2 = []

        self.TensionDamageCrack.append((0.0, 0.0))
        if cstart > 0.0:
            self.ConcreteStressCrackTension2.append((tensilestrength * 1.0, 0.0))
            self.TensionDamageCrack.append((0.0, cstart))

        #Strain is crack width in this context
```

354
#CBA to change code

\[ \text{Strain} = 0.0 \times \text{self GF/tensile strength} \]
\[ \text{crackw} = 0.0 \times \text{self GF/tensile strength} \]

while crackw < 20.0 \times \text{self GF/tensile strength}:

\[ \text{crackw} = \text{Strain} \]
\[ \text{print ‘ ‘} \]
\[ \text{for } j \text{ in range(0, len(self.ConcreteStressCrackTension) - 1)}: \]
\[ \text{line1 = self.ConcreteStressCrackTension[} j \text{]} \]
\[ \text{strain1 = line1[} 1 \text{]} \]
\[ \text{sigma1 = line1[} 0 \text{]} \]

\[ \text{line2 = self.ConcreteStressCrackTension[} j + 1 \text{]} \]
\[ \text{strain2 = line2[} 1 \text{]} \]
\[ \text{sigma2 = line2[} 0 \text{]} \]

\[ \text{print ‘%012e ::%012e :%012e ‘} \]
\[ \text{if (strain1 <= Strain) and (strain2 > Strain):} \]
\[ \text{ratio = (Strain - strain1) / (strain2 - strain1)} \]
\[ \text{sigma = ratio * (sigma2 - sigma1) + sigma1} \]
\[ \text{self.ConcreteStressCrackTension2.append((sigma, Strain + cstart))} \]
\[ \text{break} \]

\[ \text{if crackw <= 0.0:} \]
\[ \text{print Strain} \]
\[ \text{print ‘The d’} \]
\[ \text{d=0.0} \]

\[ \text{else:} \]
\[ \text{#Assume 1m effective length for the element} \]
\[ \text{crackoffset = -(tensile strength \times 1.0 / Ect)} \]

\[ \text{if tensile strength - sigma} < 0.0000000000001:} \]
\[ \text{sigmar = 1/0.0000000000001} \]
\[ \text{else:} \]
\[ \text{sigmar = sigma / (tensile strength - sigma)} \]

\[ \text{c1 = Strain} \]
\[ \text{c2 = (c1 + (1.0 - self.crackRatio) * crackoffset) / (sigmar + self.crackRatio)} \]
\[ \text{c3 = (c1 + c2) + self.crackRatio} \]

\[ \text{#Undamaged} \]
\[ \text{c2u = (-crackoffset) / (sigmar + 1.0)} \]
\[ \text{c3u = (c1 + c2u)} \]

\[ \text{utpl = crackoffset + c3} \]
\[ \text{utck = crackoffset + c3u} \]

\[ \text{gppw = Ect * (utck - utpl)} \]
\[ \text{fl = sigma} \]
\[ \text{d = gppw / (fl + gppw)} \]
\[ \text{d = max(0.0, d)} \]
\[ \text{d = min(0.99, d)} \]

\[ \text{utpl_ = utck - ((d / (1.0 - d)) + (sigma / Ect}} \]

\[ \text{if 1 == 2:} \]
\[ \text{print ‘ ‘} \]
\[ \text{print ‘Strain’} \t \text{’ + str(Strain)} \]
\[ \text{print ‘sigmar’} \t \text{’ + str(sigmar)} \]
\[ \text{print ‘c1’} \t \text{’ + str(c1)} \]
\[ \text{print ‘c2’} \t \text{’ + str(c2)} \]
\[ \text{print ‘c3’} \t \text{’ + str(c3)} \]
\[ \text{print ‘c2u’} \t \text{’ + str(c2u)} \]
\[ \text{print ‘c3u’} \t \text{’ + str(c3u)} \]
\[ \text{print ‘utck’} \t \text{’ + str(utck)} \]
\[ \text{print ‘utpl’} \t \text{’ + str(utpl)} \]
\[ \text{print ‘d’} \t \text{’ + str(d)} \]
\[ \text{print ‘utpl_’} \t \text{’ + str(utpl_)} \]

355
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
#self. ConcreteStressCrackTension. append((sigma, c3u))
self. TensionDamageCrack. append((d, c3u+cstart))

Strain = Strain + 0.2 * self.GF / tensilestrength

#Mat. concreteDamagedPlasticity. ConcreteTensionStiffening(self. ConcreteStressStrainTension2)
Mat. concreteDamagedPlasticity. ConcreteTensionStiffening(self. ConcreteStressCrackTension2, type=DISPLACEMENT)

Mat. concreteDamagedPlasticity. ConcreteTensionDamage(self. TensionDamageCrack, type=DISPLACEMENT)
Mat. concreteDamagedPlasticity. concreteTensionDamage. setValues(compressionRecovery=0.0)

self. ConcreteStressStrainTensionActual = self. ConcreteStressCrackTension2

#Compression
Strain = 0.0
while Strain < 0.02:
    #Means that the SS curve doesn’t start at zero, as required by damaged plasticity model
    strainAdjustment = self.ecu / 3.0

    if 1 == 2:
        #EN 1993−1−2
        Stress = 3 * ((Strain+strainAdjustment)/self.ecu)/(2*math.pow((Strain+strainAdjustment)/self.ecu,3))
    else:
        #Plateu for confined concrete − No crushing
        if Strain+strainAdjustment <= self.ecu or self.includeCrushing == True:
            Stress = 3 * ((Strain+strainAdjustment)/self.ecu)/(2*math.pow((Strain+strainAdjustment)/self.ecu,3))
        else:
            Stress = 1.0

    Stress = Stress * 1E6 * concfc

    if Stress < 0.001:
        Stress = 0.001

    self. ConcreteStressStrainCompression. append((Stress, Strain))
    self. ConcreteStressStrainCompressionActual. append((Stress, Strain+strainAdjustment))

    if (Strain+strainAdjustment) > self.ecu:
        self. ConcreteDamage. append((0.0, Strain))
    else:
        self. ConcreteDamage. append((0.0, Strain))

    Strain = Strain + 0.0001 / 20.0

Mat. concreteDamagedPlasticity. ConcreteCompressionHardening(self. ConcreteStressStrainCompression)
Mat. concreteDamagedPlasticity. ConcreteCompressionDamage(self. ConcreteDamage)
Mat. concreteDamagedPlasticity. concreteCompressionDamage. setValues(tensionRecovery=1.0)

if ConcreteNo == 0:
    session. XYData(name='ConcreteA1Compression', data=self.swap1(self. ConcreteStressStrainCompression), sourceDescription='')
    session. XYData(name='ConcreteA1Tension', data=self.swap1(self. ConcreteStressStrainTension), sourceDescription='')
else:
    session. XYData(name='ConcreteA4Compression', data=self.swap1(self. ConcreteStressStrainCompression), sourceDescription='')
    session. XYData(name='ConcreteA4Tension', data=self.swap1(self. ConcreteStressStrainTension), sourceDescription='')

self. EctA1 = Ect
```
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

self.ConcreteA4Compression = self.ConcreteStressStrainCompressionActual
self.ConcreteA4Tension = self.ConcreteStressStrainTensionActual

Mat = myM.Material(description='Steel - Plate Grade 1', name='Steel_Plate_1')
Mat.Elastic(table=((210.0+1E9, 0.3), ))
y = self.plateFy1
fu = self.plateFu1

# In python math.log with no argument is equal to the natural log
# Peak reached at 10% normal strain
Mat.Plastic(table=(
    (fy*1E6, 0.0),
    #(fu*1E6, math.log(1.002/1.0)),
    #(fu*1E6, math.log(1.005/1.0)),
    #(fu*1E6, math.log(1.01/1.0)),
    #(fu*1E6, math.log(1.02/1.0)),
    #(fu*1E6, math.log(1.05/1.0)),
    (fu*1E6, math.log(1.1/1.0)),
    (fu*1E6, math.log(1.2/1.0)),
    (fu*1E6, math.log(1.5/1.0)),
))
Mat.Density(table=((7850*self.steelMassScale, ), ))
Mat.Damping(alpha=self.materialDampingAlpha, beta=self.materialDampingBeta)

Mat = myM.Material(description='Steel - Plate Grade 2', name='Steel_Plate_2')
Mat.Elastic(table=((self.plateE2, 0.3), ))
y = self.plateFy2
fu = self.plateFu2
Mat.Plastic(table=(
    (fy*1E6, 0.0),
    #(fu*1E6, math.log(1.002/1.0)),
    #(fu*1E6, math.log(1.005/1.0)),
    #(fu*1E6, math.log(1.01/1.0)),
    #(fu*1E6, math.log(1.02/1.0)),
    #(fu*1E6, math.log(1.05/1.0)),
    (fu*1E6, math.log(1.1/1.0)),
    (fu*1E6, math.log(1.2/1.0)),
    (fu*1E6, math.log(1.5/1.0)),
))
Mat.Density(table=((7850*self.steelMassScale, ), ))
Mat.Damping(alpha=self.materialDampingAlpha, beta=self.materialDampingBeta)

Mat = myM.Material(description='Steel - Plate Grade 3', name='Steel_Plate_3')
Mat.Elastic(table=((self.plateE3, 0.3), ))
y = self.plateFy3
fu = self.plateFu3
Mat.Plastic(table=(
    (fy*1E6, 0.0),
    #(fu*1E6, math.log(1.002/1.0)),
    #(fu*1E6, math.log(1.005/1.0)),
    #(fu*1E6, math.log(1.01/1.0)),
    #(fu*1E6, math.log(1.02/1.0)),
    #(fu*1E6, math.log(1.05/1.0)),
    (fu*1E6, math.log(1.1/1.0)),
    (fu*1E6, math.log(1.2/1.0)),
    (fu*1E6, math.log(1.5/1.0)),
))
Mat.Density(table=((7850*self.steelMassScale, ), ))
Mat.Damping(alpha=self.materialDampingAlpha, beta=self.materialDampingBeta)

StudTypes = ['b1', 'b2', 't1', 't2']
for studType in StudTypes:
    if studType == 'b1':
        fy = self.studb1fy
        fu = self.studb1fu
    elif studType == 'b2':
        fy = self.studb2fy
        fu = self.studb2fu
    elif studType == 't1':
        fy = self.studb1fy
        fu = self.studb2fu
    else:
        fy = self.studb1fy
        fu = self.studb2fu
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
fu = self.studb1fu
elif studType == '2':
fy = self.studb2fy
fu = self.studb2fu

if fy == 0:
    fy = fu - 50.0

if fu < 50:
    fy = 1.0
fu = 2.0

Mat = myM.Material(description='Steel - Studs', name='Steel_Stud_1'+studType)
Mat.Elastic(table=((210.0*1E9, 0.3), ))

Mat = myM.Material(description='Steel with failure', name='Steel_Stud_withFail_1'+studType)
Mat.Elastic(table=((210.0*1E9, 0.3), ))

Mat = myM.Material(description='Concrete', name='Concrete_1'+studType)
Mat.Elastic(table=((200.0*1E9, 0.3), ))

Mat = myM.Material(description='Steel for supports', name='Steel_Supports')
Mat.Elastic(table=((210.0*1E9/5.0, 0.3), ))

Mat = myM.Material(description='Transfer Beam Steel', name='Transfer_Beam')
Mat.Elastic(table=((210.0*1E9/5.0, 0.3), ))

def CreateBeamPart(self):
    # Create Beam part
    # Create a sketch
    myM.ConstrainedSketch(name='ConcreteBlock', sheetSize=200.0)
s = myM.sketches['ConcreteBlock']

    # Extrusion in the cross section direction

    for area in self.Areas:
        if area[0] == '402_Concrete':
            Points = area[1]
            print Points
```

358
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
for i in range(0, len(Points) - 1, 1):
    s.Line(point1=Points[i], point2=Points[i+1])
    s.Line(point1=Points[len(Points) - 1], point2=Points[0])

#Create a part - TWO_D_PLANAR, THREE_D
myM.Part(dimensionality=TWO_D_PLANAR, name="Beam", type=DEFORMABLE_BODY)

#Extrude
myPart = myM.parts["Beam"]
myPart.BaseShell(ske

p = myM.parts["Beam"]
f1, e, d = p.faces, p.edges, p.datums
i = p.MakeSketchTransform(sketchPlane=f1[0], sketchPlaneSide=SIDE1, origin=(0.0, 0.0, 0.0))
s1 = myM.ConstrainedSketch(name="Stud Areas", sheetSize=2.14, gridSpacing=0.05, transform=t)
studNo = 0

#Split at studs
for row in self.studPositions:
    studNo = studNo + 1
    Points = row[0]
    print('Split at studs: ' + str(row))
s1.Line(point1=Points[0], point2=Points[len(Points) - 1])

#Cut concrete lines at interactions and concrete lines
for line in self.Lines:
    if line[0] == "401_Beam Interactions" or line[0] == "403_Concrete Lines":
        Points = line[1]

for i in range(0, len(Points) - 1, 1):
    s1.Line(point1=Points[i], point2=Points[i+1])

p.PartitionFaceBySketch(faces=p.faces, sketch=s1)

#session.viewports[ 'Viewport : 1' ].setValues(displayedObject=p)

for area in self.Areas:
    if area[0] == "403_Fine Mesh":
        Points = area[1]
        xmin = +1000000.0
        xmax = -1000000.0
        ymin = +1000000.0
        ymax = -1000000.0
for i in range(0, len(Points), 2):
    if Points[i][0] < xmin:
        xmin = Points[i][0]
    if Points[i][0] > xmax:
        xmax = Points[i][0]
    if Points[i][1] < ymin:
        ymin = Points[i][1]
    if Points[i][1] > ymax:
        ymax = Points[i][1]

self.setFromCriteria(p.vertices, 'Fine Mesh Points', object=p, yimin=ymin, ymax=ymax, xmin=xmin, xmax=xmax)

vsStudT = []
try:
    for v1 in p.sets["Fine Mesh Points"].vertices:
        vsStudT.append(v1.index)
except KeyError:
    print('No fine points found')

if self.Explicit == True:
    #Explicit Elements
elemType1 = mesh.ElemType(elemCode=CPS4R, elemLibrary=EXPLICIT,
                            secondOrderAccuracy=OFF, hourglassControl=DEFAULT,
                            distortionControl=DEFAULT)
elemType2 = mesh.ElemType(elemCode=CPS3, elemLibrary=EXPLICIT,
                            distortionControl=DEFAULT)

359
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
e else:
    # Standard Elements
    elemType1 = mesh.ElemType(elemCode=CPS8, elemLibrary=STANDARD)
    elemType2 = mesh.ElemType(elemCode=CPS6, elemLibrary=STANDARD)

    p.setElementType(elemTypes=(elemType1, elemType2), regions=p.faces)

    mmM.HomogeneousSolidSection(material='Concrete', name='Concrete', thickness=1.0)
    mmM.HomogeneousSolidSection(material='Weak Concrete', name='Weak Concrete', thickness=1.0)

    p.SectionAssignment(offset=0.0, offsetField='', offsetType=MIDDLE_SURFACE, region=regionToolset.Region(faces=p.sets['Beam Areas'].faces), sectionName='Concrete', thicknessAssignment=FROM_SECTION)

    print p.sectionAssignments[0]

    #p.seedEdgeBySize(constraint=FINER, deviationFactor=0.1, edges=p.edges, size=self.meshScale)

    #print vsStudT
    edges1 = p.edges

    print '########'
    for edge in edges1:
        vs = edge.getVertices()

        v1 = False
        v2 = False
        if vs[0] in vsStudT:
            v1 = True
        if vs[1] in vsStudT:
            v2 = True

        if (v1 == True) and (v2 == True):
            p.seedEdgeBySize((edge,), size=self.studMeshScale, deviationFactor=0.1, minSizeFactor=0.1)
        elif (v1 == True) and (v2 == False):
            p.seedEdgeByBias(end1Edges=(edge,), biasMethod=SINGLE, minSize=self.studMeshScale, maxSize=self.meshScale)
        elif (v1 == False) and (v2 == True):
            p.seedEdgeByBias(end2Edges=(edge,), biasMethod=SINGLE, minSize=self.studMeshScale, maxSize=self.meshScale)
        else:
            p.seedEdgeBySize((edge,), size=self.meshScale, deviationFactor=0.1, minSizeFactor=0.1)

    for face in p.faces:
        print face
        p.Set(name='Beam Areas ' + str(face.index), faces=p.faces[face.index:face.index+1])

    p.generateMesh(regions=(p.sets['Beam Areas ' + str(face.index)].faces,))

    def CreateStudPart(self):
        import math
        studNo = 0
        for row in self.studPositions:
            studNo = studNo + 1

            Points = row[0]
            studType = row[1]
            studNum = row[2]

            # Stud Type Selection
            if studType == 'b1':
                studDia = self.studb1Dia
                studResistance = self.studb1Resistance
            elif studType == 'b2':
                studDia = self.studb2Dia
                studResistance = self.studb2Resistance
            elif studType == 't1':
                studDia = self.studt1Dia
                studResistance = self.studt1Resistance

        print str(360)
```

360
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
elif studType == 't2':
    studDia = self.studt2Dia
    studResistance = self.studt2Resistance
else:
    raise ValueError('studEntry[1] should be b1, b2, t1 or t2. ' + studEntry[1] + ' not found')

#End Stud Type Selection

name1 = 'Stud_Base-' + str(studNo)
name2 = 'Stud-' + str(studNo)
p2 = self.mM.p['Beam']

print 'Finding stud Nodes: ' + str(Points)
for i in range(0, len(Points) - 1):
    if Points[i][0] == Points[i + 1][0]:
        ymin = min(Points[i + 0][1], Points[i + 1][1])
        ymax = max(Points[i + 0][1], Points[i + 1][1])
        # Y line
        self.setFromCriteria(p2.nodes.getByBoundingBox(Points[i + 0][0] - 0.0001, Points[i + 0][1] - 0.0001, 0.0, Points[i + 1][0] + 0.0001, Points[i + 1][1] + 0.0001, 18.0),
            name + '-Nodes', object=p2, ymin=Points[i + 0][1], ymax=Points[i + 1][1], x=Points[i + 0][0])
    else:
        xmin = min(Points[i + 0][0], Points[i + 1][0])
        xmax = max(Points[i + 0][0], Points[i + 1][0])
        self.setFromCriteria(p2.nodes.getByBoundingBox(xmin - 0.0001, Points[i + 0][1] - 0.0001, 0.0, xmax + 0.0001, Points[i + 1][1] + 0.0001, 18.0),
            name + '-Nodes', object=p2, xmin=Points[i + 0][0], xmax=Points[i + 1][0], y=Points[i + 0][1])

self.setError = False
studNo = 0
for row in self.studPositions:
    studNo = studNo + 1

name1 = 'Stud_Base-' + str(studNo)
name2 = 'Stud-' + str(studNo)

Points = row[0]
studType = row[1]
studNum = row[2]

# Stud Type Selection
if studType == 'b1':
    studDia = self.studb1Dia
    studResistance = self.studb1Resistance
elif studType == 'b2':
    studDia = self.studb2Dia
    studResistance = self.studb2Resistance
elif studType == 't1':
    studDia = self.studt1Dia
    studResistance = self.studt1Resistance
elif studType == 't2':
    studDia = self.studt2Dia
    studResistance = self.studt2Resistance
else:
    raise ValueError('studEntry[1] should be b1, b2, t1 or t2. ' + studEntry[1] + ' not found')

Coords = []
Count = 1
try:
    for i in p2.sets[name2 + '-Nodes'].nodes:
        Coords.append((i.coordinates[0], i.coordinates[1], i.label))
        Count = Count + 1
except KeyError:
    name2 = name2 + '-error'
```

361
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
self.setsError = True

# Definition points
# for p1 in Points:
# Coords.append((p1[0],p1[1]))

print Points

# If there are 3 points it is stud, 4 for a tie-bar
# Only append centre point of stud
if len(Points) >= 4:
    Coords.append((Points[0][0], Points[0][1], 'end1'))
    Coords.append((Points[1][0], Points[1][1], 'end2'))
else:
    Coords.append((Points[2][0], Points[2][1], 'end1'))

CoordsSet = sorted(set(Coords))

s1 = myM.ConstrainedSketch(name=name2, sheetSize=200.0)
for i in range(len(CoordsSet) - 1, 0, -1):
    dx = CoordsSet[i][0] - CoordsSet[i + 1][0]
    dy = float(CoordsSet[i][1]) - float(CoordsSet[i + 1][1])

    s1.Line(point1=(CoordsSet[i][0], CoordsSet[i][1]), point2=(CoordsSet[i + 1][0], CoordsSet[i + 1][1]))

    p = myM.Part(name=name2, dimensionality=THREE_D, type=DEFORMABLE_BODY)
    p.BaseWire(sketch=s1)

for i in range(len(CoordsSet) - 1, 0, -1):
    dx = CoordsSet[i][0] - CoordsSet[i + 1][0]
    dy = float(CoordsSet[i][1]) - float(CoordsSet[i + 1][1])

    lineLength = math.sqrt(math.pow(dx, 2) + math.pow(dy, 2))

    if lineLength < 0.00001:
        for j in range(0, len(CoordsSet), 1):
            print CoordsSet[j]
            print '###'
            print CoordsSet[i]
            print p2.nodes[CoordSet[i][2] - 1]
            print CoordSet[i + 1]
            print p2.nodes[CoordSet[i + 1][2] - 1]

            raise ValueError('Gap between stud ' + str(studNo) + ' and stud is very small - ' + str(lineLength))

if 1 == 2:

    myM.CircularProfile(name='Shear Studs', r=0.019/2)
    myM.RectangularProfile(name=name1, b=studDia*2.0, a=studDia+studNum*myM.widthScale*2.0)
    self.SquareToCircleRatio = 3.14159/4
    myM.RectangularProfile(name=name2, b=studDia*1.0, a=studDia+studNum*myM.widthScale/1.0)
else:
    b = 0.3
    c = studNum*myM.widthScale*2.0
    d = studDia

    eqt1 = 0.25 + (2*b - math.pow(4*b, 2) - math.pow(4*c*d, 0.5))
    print eqt1

    eqt2 = 0.25 + (2*b - math.pow(4*b, 2) - math.pow(4*c*d, 0.5))
    print eqt2

    myM.BoxProfile(name=name1, b=b, a=b*2, uniformThickness=ON, t1=eqt1)
    myM.BoxProfile(name=name2, b=b, a=b, uniformThickness=ON, t1=eqt1)

    myM.BeamSection(integration=DURING_ANALYSIS,
                    material='Steel_Stub', studType, name=name1,
                    poissonRatio=0.2, profile=name1,
                    temperatureVar=LINEAR)
    myM.BeamSection(integration=DURING_ANALYSIS,
                    material='Steel_Stub', studType, name=name2,
                    poissonRatio=0.2, profile=name2,
                    temperatureVar=LINEAR)
```

362
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
name=name2,
poissonRatio=0.2,
profile=name2,
temperatureVar=LINEAR)

region = regionToolset.Region(edges=p.edges)
p.sectionAssignment(region=region, sectionName=name2, offset=0.0, offsetType=MIDDLE_SURFACE,
offsetField="", thicknessAssignment=FROM_SECTION)

p.assignBeamSectionOrientation(method=N1_COSINES, n1=(0.0, 0.0, -1.0), region=region)
p.seedEdgeBySize(constraint=FINER, deviationFactor=0.1, edges=p.edges, size=100.0)

self.meshScale = self.meshScale

# Non-Linear springs for shear connectors

a = mModel.rootAssembly

t = session.viewports["Viewport: 1"].setValues(displayedObject=a)
session.viewports["Viewport: 1"].assemblyDisplay.setValues(interactions=OFF,
constraints=OFF,
engineeringFeatures=OFF)

self.shearConnectorLoadSlip = []
self.shearConnectorTensionSlip = []


if self.connectorModel == 4 or self.connectorModel == 6:
    if self.loadSlips == 4:
        self.loadSlips = [(-10.0, -1.0),
                         (-2.0, -1.0),
                         (-1.0, -0.9),
                         (-0.2574, -0.7),
                         (-0.0, 0.0),
                         (+0.2574, +0.7),
                         (+1.0, +0.9),
                         (+2.0, +1.0),
                         (+10.0, +1.0)]

    elif self.connectorModel == 6:
        fc = self.concretefc
        self.loadSlips = [(-10.0, -1.0),
                         (-406.0+0.0+0.011-251.0+fc-0.00001)+studDia+1000.0, -1),
                         (-371.0+0.0+0.011-208.0+fc-0.00001)+studDia+1000.0, -1),
                         (-319.0+0.011-170.0+fc-0.00001)+studDia+1000.0, -0.99),
                         (-223.0+0.011-119.0+fc-0.00001)+studDia+1000.0, -0.95),
                         (-156.0+0.011-70.0+fc-0.00001)+studDia+1000.0, -0.9),
                         (-138.0+0.011-72.0+fc-0.00001)+studDia+1000.0, -0.85),
                         (-143.0+0.011-108.0+fc-0.00001)+studDia+1000.0, -0.8),
                         (-120.0+0.011-102.0+fc-0.00001)+studDia+1000.0, -0.7),
                         (-102.0+0.011-96.0+fc-0.00001)+studDia+1000.0, -0.6),
                         (-80.0+0.011-73.0+fc-0.00001)+studDia+1000.0, -0.5),
                         (-63.0+0.011-55.0+fc-0.00001)+studDia+1000.0, -0.4),
                         (-52.0+0.011-48.0+fc-0.00001)+studDia+1000.0, -0.3),
                         (-40.0+0.011-37.0+fc-0.00001)+studDia+1000.0, -0.2),
                         (-22.0+0.011-20.0+fc-0.00001)+studDia+1000.0, -0.1),
                         (10.0+0.011+0.0+fc-0.00001)+studDia+1000.0, 0),
                         (22.0+0.011+20.0+fc-0.00001)+studDia+1000.0, 0.1),
                         (40.0+0.011+37.0+fc-0.00001)+studDia+1000.0, 0.2),
                         (52.0+0.011+48.0+fc-0.00001)+studDia+1000.0, 0.3),
                         (63.0+0.011+55.0+fc-0.00001)+studDia+1000.0, 0.4),
                         (80.0+0.011+73.0+fc-0.00001)+studDia+1000.0, 0.5),
                         (102.0+0.011+96.0+fc-0.00001)+studDia+1000.0, 0.6),
                         (120.0+0.011+102.0+fc-0.00001)+studDia+1000.0, 0.7),
                         (143.0+0.011+108.0+fc-0.00001)+studDia+1000.0, 0.8),
                         (158.0+0.011+72.0+fc-0.00001)+studDia+1000.0, 0.85),
                         (156.0+0.011+70.0+fc-0.00001)+studDia+1000.0, 0.9),
                         (223.0+0.011+119.0+fc-0.00001)+studDia+1000.0, 0.95),
                         (319.0+0.011+170.0+fc-0.00001)+studDia+1000.0, 0.99),
                         (571.0+0.011+208.0+fc-0.00001)+studDia+1000.0, 1),
```

```
100.0)
```

```
1000.0)
```
```
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

((406.0×0.001+251.0×fc×−0.00001)×studDia×1000.0,1),
(+10.0,+1.0)]

for forcepair in self.loadSlips:
    force = studResistance + forcepair[1]
    slip = forcepair[0]/1000.0
    self.shearConnectorTensionSlip.append((force, slip + self.additionalSlipFactor))
    self.shearConnectorLoadSlip.append((force, slip + self.additionalSlipFactor))

elif self.connectorModel == 3 or self.connectorModel == 5:
    import math
    slip = -10.0/1000.0
    while (slip < 10.0/1000.0):
        slip2 = slip
        if (slip < 0.0):
            slip2 = -slip
        slip3 = slip2 + 1000.0
        if self.connectorModel == 3:
            forcefactor = math.pow((1.0 - math.exp(-1.0*slip3)),0.558)
        elif self.connectorModel == 5:
            forcefactor = math.pow((1.0 - math.exp(-1.535*slip3)),0.989)
        if forcefactor > 0.3:
            # Linear up to 30% resistance
            force = studResistance * forcefactor
        else:
            self.shearConnectorTensionSlip.append((force, slip + self.additionalSlipFactor))
            self.shearConnectorLoadSlip.append((force, slip + self.additionalSlipFactor))
        slip = slip + 0.1/1000.0

# No Shear Stiffness
#self.shearConnectorLoadSlip = [(0.0,0.0),(0.0,1.0)]

# Copy only positive end of load-slip curve for output
self.shearConnectorLoadFactorSlip = []
for i in range(len(self.shearConnectorLoadSlip[0])):
    if self.shearConnectorLoadSlip[i][0] >= 0.0:
        self.shearConnectorLoadFactorSlip.append((self.shearConnectorLoadSlip[i][0],self.shearConnectorLoadSlip[i][1]))

session.XYData(name='StudLoadSlip' + name2, data=self.swap1(self.shearConnectorLoadSlip), sourceDescription='')
session.XYData(name='StudLoadFactorSlip' + name2, data=self.swap1(self.shearConnectorLoadFactorSlip), sourceDescription='')
session.XYData(name='StudTensionSlip' + name2, data=self.swap1(self.shearConnectorTensionSlip), sourceDescription='')

# Create a connector
mpM.ConnectorSection(name='Spring-' + name2, translationalType=CARTESIAN, rotationalType=ROTATION)

# Scale by stud number and width scale
shearConnectorLoadSlipScaled = []
for entry in self.shearConnectorLoadSlip:
    shearConnectorLoadSlipScaled.append((entry[0]*self.widthScale*studNum,entry[1]))

shearConnectorTensionSlipScaled = []
for entry in self.shearConnectorTensionSlip:
    shearConnectorTensionSlipScaled.append((entry[0]*self.widthScale*studNum,entry[1]))

session.XYData(name='StudLoadFactorSlipScaled-' + name2, data=self.swap1(shearConnectorLoadSlipScaled), sourceDescription='Load Slip Relationship of ' + name2)

# 1 is the x direction
# 2 is y direction – beware directionality of element
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
# Damping – Analysis guide – 31.2.3 Connector damping behavior
yFixed = True
xFixed = False
if (yFixed == False) and (xFixed == False):
    elastic_0 = connectorBehavior.ConnectorElasticity(components=(1,), behavior=NONLINEAR, table=shearConnectorLoadSlipScaled)
    elastic_1 = connectorBehavior.ConnectorElasticity(components=(2,), behavior=NONLINEAR, table=shearConnectorTensionSlipScaled)
    elastic_3 = connectorBehavior.ConnectorElasticity(components=(3,4,5,6,), behavior=RIGID)
    failure_3 = connectorBehavior.ConnectorFailure(components=(2,), releaseComponent=ALL, maxMotion=sscc)
    mM.sections['Spring→' + name2].setValues(behaviorOptions=(elastic_0, elastic_1, elastic_3, failure_3))
    else if (yFixed == True) and (xFixed == False):
        elastic_0 = connectorBehavior.ConnectorElasticity(components=(1,), behavior=NONLINEAR, table=shearConnectorLoadSlipScaled)
        elastic_3 = connectorBehavior.ConnectorElasticity(components=(2,3,4,5,6,), behavior=RIGID)
        mM.sections['Spring→' + name2].setValues(behaviorOptions=(elastic_0, elastic_3))
    else if (yFixed == True) and (xFixed == True):
        elastic_3 = connectorBehavior.ConnectorElasticity(components=(1,2,3,4,5,6,), behavior=RIGID)
        mM.sections['Spring→' + name2].setValues(behaviorOptions=(elastic_3,))

def CreatePlatesPart(self):
    # Create Plates part
    # Create a sketch
    s = mM.ConstrainedSketch(name='Plate', sheetSize=200.0)
    # Extrusion in the cross section direction
    for area in self.Areas:
        if area[0] == "401_Outline": # "401_Outline":
            Points = area[1]
            for i in range(0,len(Points)-1,1):
                s.Line(point1=Points[i], point2=Points[i+1])
                s.Line(point1=Points[len(Points)-1], point2=Points[0])
    # Create a part – TWO_D_PLANAR, THREE_D
    p = mM.Part(dimensionality=TWO_D_PLANAR, name='Plates', type=DEFORMABLE_BODY)
    p.BaseShell(sketch=s)
    s = mM.ConstrainedSketch(name='Concrete_Areas', sheetSize=200.0)
    # Remove concrete areas
    for area in self.Areas:
        if area[0] == "402_Concrete": # "401_Outline":
            Points = area[1]
            for i in range(0,len(Points)-1,1):
                s.Line(point1=Points[i], point2=Points[i+1])
                s.Line(point1=Points[len(Points)-1], point2=Points[0])
    p.PartitionFaceBySketch(faces=p.faces, sketch=mM.sketches['Concrete_Areas'])
    print p.faces
    print len(p.faces)
cFaces = []
    for f in p.faces:
        vs = f.getVertices()
        points = []
        for v in vs:
            points.append((p.vertices[v].pointOn[0][0],p.vertices[v].pointOn[0][1]))
        print "###"
for area in self.Areas:
    if area[0] == '402_Concrete':
        inArea = True
        for p1 in points:
            p2inArea = False
            for p2 in area[1]:
                if abs(p1[0] - p2[0]) < 0.0005 and abs(p1[1] - p2[1]) < 0.0005:
                    p2inArea = True
                    break
            if p2inArea == False:
                inArea = False
        if inArea == True:
            cFaces.append(f)
            print 'Concrete Area Points: ' + str(Points)
        p.RemoveFaces(faceList = cFaces, deleteCells=False)

# Partition Lines
s = mph.ConstrainedSketch(name='Steel_Lines', sheetSize=200.0)
for line in self.Lines:
    if line[0] == '302_Steel Lines':
        Points = line[1]
        for i in range(0, len(Points)-1):
            s.Line(point1=Points[i], point2=Points[i+1])
for row in self.studPositions:
    Points = row[0]
    s.Line(point1=Points[0], point2=Points[len(Points)-1])
    p.PartitionFaceBySketch(faces=p.faces, sketch=s)

# Sort areas that are steel grade 2
anum = 0
for area in self.Areas:
    if area[0] == '351_Steel Grade 2' or area[0] == '352_Steel Grade Holes':
        anum = anum + 1
        Points = area[1]
        xmin = +1000000.0
        xmax = -1000000.0
        ymin = +1000000.0
        ymax = -1000000.0
        for i in range(0, len(Points), 1):
            if Points[i][0] < xmin:
                xmin = Points[i][0]
            if Points[i][0] > xmax:
                xmax = Points[i][0]
            if Points[i][1] < ymin:
                ymin = Points[i][1]
            if Points[i][1] > ymax:
                ymax = Points[i][1]
        if area[0] == '351_Steel Grade 2':
            self.setFromCriteria(p.faces, 'Steel Grade 2', object=p, ymin=ymin, ymax=ymax, xmin=xmin, xmax=xmax)
        elif area[0] == '352_Steel Grade Holes':
            self.setFromCriteria(p.faces, 'Steel Grade 3 - ' + str(anum), object=p, ymin=ymin, ymax=ymax, xmin=xmin, xmax=xmax)
        if len(p.sets['Steel Grade 3 - ' + str(anum)]) != 4:
            if self.ModelNo > 50 and self.ModelNo < 60:
                pass
                #Corner models
            else:
                raise ValueError('Steel Grade 3 - ' + str(anum) + ' doesn’t contain enough areas')
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
self.setFromCriteria(p.faces, 'Steel Grade 3', object=p, ymin=ymin, ymax=ymax, xmin=xmin, xmax=xmax)

# MoSh
if self.Explicit == True:
    # Explicit Elements
    elemType1 = mesh.ElemType(elemCode=CPS4R, elemLibrary=EXPLICIT,
                               secondOrderAccuracy=OFF, hourglassControl=DEFAULT,
                               distortionControl=DEFAULT)
    elemType2 = mesh.ElemType(elemCode=CPS3, elemLibrary=EXPLICIT,
                               distortionControl=DEFAULT)
else:
    # Standard Elements
    elemType1 = mesh.ElemType(elemCode=CPS8, elemLibrary=STANDARD)
    elemType2 = mesh.ElemType(elemCode=CPS6, elemLibrary=STANDARD)
p.setElementType(elemTypes=(elemType1, elemType2), regions=(p.faces, ))
self.addToSet(origin='Steel all faces', toadd=p.faces, object=p)

myM.HomogeneousSolidSection(material='Steel Plate 1', name='Plates', thickness=1.0)
p.SectionAssignment(offset=0.0, offsetField=' ', offsetType=MIDDLE_SURFACE, region=p.sets['Steel all faces'], sectionName='Plates', thicknessAssignment=FROM_SECTION)
try:
    p.sets['Steel Grade 2'].faces
except KeyError:
    print('No Steel Grade 2 Found')

myM.HomogeneousSolidSection(material='Steel Plate 2', name='Plates Grade 2', thickness=1.0)
p.SectionAssignment(offset=0.0, offsetField=' ', offsetType=MIDDLE_SURFACE, region=p.sets['Steel Grade 2'], sectionName='Plates Grade 2', thicknessAssignment=FROM_SECTION)
except KeyError:
    print('No Steel Grade 3 Found')
p.seedEdgeBySize(constraint=FINER, deviationFactor=0.1, edges=p.edges, size=self.plateMeshScale)
p.generateMesh()

def CreateSupportPart(sSelf):
    # Create Support part
    myM.ConstrainedSketch(name='Support', sheetSize=200.0)
    Support = myM.sketches['Support']
    Support.rectangle(point1=(-self.supportWidth*0.5, -self supporth), point2=(+self.supportWidth*0.5, 0.0))
    Support.FilletByRadius(radius=self.supportWidth*0.05, curve1=Support.geometry[3], nearPoint1=(0.0, -self.supportWidth*0.5), curve2=Support.geometry[2], nearPoint2=(0.0, -self.supportWidth*0.5))
    Support.FilletByRadius(radius=self.supportWidth*0.05, curve1=Support.geometry[3], nearPoint1=(0.0, -self.supportWidth*0.5), curve2=Support.geometry[4], nearPoint2=(0.0, -self.supportWidth*0.5))
    session.viewports['Viewport: 1'].setValues(displayedObject=Support)

myM.Part(dimensionality=TWO_D_PLANAR, name='Support', type=DEFORMABLE_BODY)
myPart = myM.parts['Support']
myPart.BaseShell(sketch=Support)
myPart.PartitionFaceByShortestPath(faces=myPart.faces, point1=(0.0, 0.0, 0.0), point2=(0.0, -self.supportWidth*0.0))
```

367
**APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS**

```python
# Select edges and seed
# Seed all edges for now
myEdges = myPart.edges # getByBoundingBox(-SlabWidth/2, -Slabh/2, 0.0, SlabWidth/2, +Slabh/2, 18.0)
myPart.Set(edges=myEdges, name='Edge Seeds')

#myPart.Set(edges=myPart.edges[4:5], name='Bottom Edges')
#myPart.Set(edges=myPart.edges[6:7], name='Top Edges')

if self.Explicit == True:
    # Explicit Elements
    elemType1 = mesh.ElemType(elemCode=CPS4R, elemLibrary=EXPLICIT,
    secondOrderAccuracy=OFF, hourglassControl=DEFAULT,
    distortionControl=DEFAULT)
    elemType2 = mesh.ElemType(elemCode=CPS3, elemLibrary=EXPLICIT,
    distortionControl=DEFAULT)
else:
    # Standard Elements
    elemType1 = mesh.ElemType(elemCode=CPS8, elemLibrary=STANDARD)
    elemType2 = mesh.ElemType(elemCode=CPS6, elemLibrary=STANDARD)

myPart.setElementType(elemTypes=(elemType1, elemType2), regions=(myPart.faces,))

myM.HomogeneousSolidSection(material='Steel_Supports', name='Support Steel', thickness=1.0)
myPart.SectionAssignment(offset=0.0, offsetField='', offsetType=MIDDLE_SURFACE, region=regionTools.Region(faces=myPart.faces), sectionName='Support Steel', thicknessAssignment=FROM_SECTION)

myPart.seedEdgeBySize(constraint=FINNER, deviationFactor=0.1, edges=myEdges, size=self.supportWidth/(64.0))
# works well

# Causing Crash
myPart.Set(edges=myPart.edges.getByBoundingBox(-self.supportWidth*0.5-0.0.0001, 0.0, self.supportWidth*0.5+0.0001, 0.0), name='Top Edges')
myPart.Set(edges=myPart.edges.getByBoundingBox(-self.supportWidth*0.5-0.0001, -self.supporth-0.0001, 0.0, self.supportWidth*0.5+0.0001, self.supporth+0.0001, 0.0), name='Bottom Edges')

myPart.generateMesh()

xTarget = 0.0 # -self.span+1/3 # Doesn't like 0 for some reason
yTarget = -self.supporth

n1 = self.findClosest(xTarget, yTarget, myPart.nodes)
myPart.Set(nodes=myPart.nodes[n1.label-1:n1.label], name='PointLoad')

v1 = self.findClosest(xTarget, yTarget, myPart.vertices)
myPart.Set(vertices=myPart.vertices[v1.index:v1.index+1], name='PointLoadV')
myPart.Set(faces=myPart.faces, name='All Faces')

session.viewports['Viewport: 1'].setValues(displayedObject=myPart)

p = myM.Part(name='DamperPart', dimensionality=TWO_D_PLANAR, type=DISCRETE_RIGID_SURFACE)

def CreateStudAssembly(self):
    myM.rootAssembly.DatumCsysByDefault(CARTESIAN)
    a = myM.rootAssembly
    myM.rootAssembly.Instance(dependent=ON, name='RC-1', part=myPart)

    session.viewports['Viewport: 1'].setValues(displayedObject=a)
    session.viewports['Viewport: 1'].assemblyDisplay.setValues(interactions=ON, constraints=ON, connectors=ON, engineeringFeatures=ON)

    studNo = 0
    for studEntry in self.studPositions:
        studNo = studNo + 1
        try:
            a.Instance(name='Stud-' + str(studNo), part=myM.parts['Stud-' + str(studNo)], dependent=ON)
        except:
            a.Instance(name='Stud-' + str(studNo) + '-error', part=myM.parts['Stud-' + str(studNo)+ '-error'], dependent=ON)
```

368
Appendix A: Python Code for Modelling an SCS Beam Using ABAQUS

```python
if self.setsError == True:
    raise ValueError('Studs sets not found -- Studs marked with "Error"')

def CreateAssembly(self):
    # Create Assembly
    m = self.m
    if self.Explicit == True:
        m.ExplicitDynamicsStep(name='Gravity-Step', previous='Initial',
                               setValues(timePeriod=self.GravityTime, scaleFactor=1.0,
                                          increment=0, linearViscosity=self.linearViscosityStep))
        m.ExplicitDynamicsStep(name='Cyclic-Step', previous='Gravity-Step',
                               setValues(timePeriod=self.LoadTime * (self.Cycles + self.EndCycles * 0 + 0.1)),
                                          scaleFactor=1.0, linearViscosity=self.linearViscosityStep))
        m.ExplicitDynamicsStep(name='Loading-Step', previous='Cyclic-Step',
                               setValues(timePeriod=0, scaleFactor=1.0, linearViscosity=self.linearViscosityStep))
    else:
        m.ElasticStep(name='Loading-Step', previous='Initial', maxNumInc=20000, initialInc=1.0/500.0,
                      maxInc=1.0/500.0, nlegeom=ON)
    # Create Assembly
    m.rootAssembly.DatumCsysByDefault(CARTESIAN)
    a = m.rootAssembly
    m.ElasticStep(name='Beam-1', previous='Initial',
                  setValues(timePeriod=self.LoadTime, scaleFactor=1.0, linearViscosity=self.linearViscosityStep))
    m.ElasticStep(name='Plates-1', previous='Beam-1',
                  setValues(timePeriod=self.LoadTime, scaleFactor=1.0, linearViscosity=self.linearViscosityStep))

    # Boundary Conditions
    supportNum = 0
    import math
    for line in self.Lines:
        name = ''
        Offset = 0.0
        if line[0] == '501_Load 1':
            name = 'Load-1'
            Offset = self.loadOffset
        elif line[0] == '502_Load 2':
            name = 'Load-2'
        elif line[0] == '503_Load 3':
            name = 'Load-3b'
        elif line[0] == '504_Load 4':
            name = 'Load-3c'
        elif line[0] == '505_Load 5':
            name = 'Load-3d'
        elif line[0] == '506_Load 6':
            name = 'Load-3e'
        elif line[0] == '507_Load 7':
            name = 'Load-3f'
        elif line[0] == '508_Load 8':
            name = 'Load-3g'
```

369
A
PPENDIX
A: P
YTHON CODE FOR MODELLING AN
SCS
BEAM USING
ABAQUS

e l i f l i n e [ 0 ] == " 551 _Supports ":
    supportNum = supportNum + 1
    name = ' Support ' + str(supportNum)

if name != " ":
    Points = line[1]
    a. Instance(name=name, part=myM. parts[ ' Support '], dependent=ON)
    a. translate(instanceList=(name, ), vector=(Points[0][0]+Offsets, Points[0][1], 0, 0))

dx = Points[1][0]−Points[0][0]
dy = Points[1][1]−Points[0][1]
angle = self. rotationAngle(dy, dx)+90.0
print angle
# angle = math. radians(angle)

#Direction is a a vector, not a coordinate
    a. rotate(instanceList=(name, ), axisPoint=(Points[0][0]+Offsets, Points[0][1], 0, 0),
               axisDirection=(0, 0, 0, 1), angle=angle)

#
# Contact Elements
#

myContactProps = myM. ContactProperty(' IntProp−Cyclic ')

# Rough contact − Stick the load points, but no reaction in x direction
# Seems to cause a problem with getting reaction forces, so assume frictionless instead
if 1 == 1:
    myContactProps. TangentialBehavior(formulation=FRICTIONLESS) #ROUGH, FRICTIONLESS
else:
    myContactProps. TangentialBehavior(formulation=PENALTY, directionality=ISOTROPIC,
                                           pressureDependency=OFF, temperatureDependency=OFF, dependencies=0,
                                           table=((0.5, ), ), shearStressLimit=None, maximumElasticSlip=FRACTION,
                                           fraction=0.005, elasticSlipStiffness=None)

    myContactProps. NormalBehavior(pressureOverclosure=HARD, allowSeparation=OFF, constraintEnforcementMethod=DEFAULT)

myContactProps = myM. ContactProperty(' IntProp−Loading ')
if 1 == 1:
    myContactProps. TangentialBehavior(formulation=FRICTIONLESS) #ROUGH, FRICTIONLESS
else:
    myContactProps. TangentialBehavior(formulation=PENALTY, directionality=ISOTROPIC,
                                           pressureDependency=OFF, temperatureDependency=OFF, dependencies=0,
                                           table=((0.5, ), ), shearStressLimit=None, maximumElasticSlip=FRACTION,
                                           fraction=0.005, elasticSlipStiffness=None)

    myContactProps. NormalBehavior(pressureOverclosure=HARD, allowSeparation=ON, constraintEnforcementMethod=DEFAULT)

    myContactProps. Damping(definition=CRI
    TICAL_DAMPING_FRACTION, tangentFraction=DEFAULT, clearanceDependence=STEP, table=((1.0, ), ))

myContactProps = myM. ContactProperty(' ShearConnection−Props ')
if 1 == 2:
    myContactProps. TangentialBehavior(formulation=FRICTIONLESS) #ROUGH, FRICTIONLESS
else:
    myContactProps. TangentialBehavior(formulation=PENALTY, directionality=ISOTROPIC,
                                           pressureDependency=OFF, temperatureDependency=OFF, dependencies=0,
                                           table=((0.3, ), ), shearStressLimit=None, maximumElasticSlip=FRACTION,
                                           fraction=0.005, elasticSlipStiffness=None)

    myContactProps. NormalBehavior(pressureOverclosure=HARD, allowSeparation=ON, constraintEnforcementMethod=DEFAULT)

myContactProps. Damping(definition=CRI
    TICAL_DAMPING_FRACTION, tangentFraction=DEFAULT, clearanceDependence=STEP, table=((1.0, ), ))
if self.Explicit == True:
    mM.ExpContactControl(name='ContCtrl-1')
else:
    mM.StdContactControl(name='ContCtrl-1')
mM.StdInitialization(name='CInit-1')

interactionNum = 0
for line in self.Lines:
    name1 = ''
    if line[0] == '602_Load Interactions':
        interactionNum = interactionNum + 1
        Points = line[1]
        if Points[0][0] == Points[1][0]:
            # Y line
            self.setFromCriteria(mM.parts['Plates'].edges, 'Interaction-Steel-' + str(interactionNum),
                object=mM.parts['Plates'].ymin=Points[0][1], ymax=Points[1][1], x=Points[0][0])
        else:
            # X line
            self.setFromCriteria(mM.parts['Plates'].edges, 'Interaction-Steel-' + str(interactionNum),
                object=mM.parts['Plates'], xmin=Points[0][0], xmax=Points[1][0], y=Points[0][1])
        xTarget = (Points[0][0] + Points[1][0]) / 2.0
        yTarget = (Points[0][1] + Points[1][1]) / 2.0
        deltaMin = 10000
        instances = re.findall(r'[A-Za-z0-9- ]+\*:', str(a.instances))
        for instance in instances:
            instance = instance.replace('"', '"
            instance = instance.replace('"', '"')
        if a.instances[instance].partName == 'Support':
            xNode = a.instances[instance].sets['PointLoad'].nodes[0].coordinates[0]
            yNode = a.instances[instance].sets['PointLoad'].nodes[0].coordinates[1]
            # delta = ((xTarget-xNode)**2+(yTarget-yNode)**2)**0.5
            if delta < deltaMin:
                deltaMin = delta
                nearestInstance = instance
        if Points[0][0] == Points[1][0]:
            # Y line
            self.setFromCriteria(a.instances[instance].edges, 'Interaction-' + instance + ' ' + str(interactionNum),
                object=a, ymin=Points[0][1], ymax=Points[1][1], x=Points[0][0])
        else:
            # X line
            self.setFromCriteria(a.instances[instance].edges, 'Interaction-' + instance + ' ' + str(interactionNum),
                object=a, xmin=Points[0][0], xmax=Points[1][0], y=Points[0][1])
        count = 0
        for instance in instances:
            instance = instance.replace('"', '"
            instance = instance.replace('"', '"')
        if found == True:
            count = count + 1
        if found == True:
            count = count + 1
        namel = 'Interaction-Load-' + str(interactionNum)
        sideRegion = a.Surface(sidelEdges=a.instances[instance].sets['Top Edges'].edges, name='Surface-Top Edges-' + instance)
        masterRegion = a.Surface(...

371
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
sideEdges = a.instances['Plates-1'].sets['Interaction-Steel-'] + str(interactionNum).edges, name='Surface-' + name1)

con = 1  # Penalty
con = 2  # Kinematic
if con == 1:
    mM.SurfaceToSurfaceContactExp(name = name1 + '-' + str(count), createStepName='Initial',
    slave = slaveRegion,
    master = masterRegion,
    mechanicalConstraint= Penalty, sliding=FINITE,
    weightingFactorType= SPECIFIED, weightingFactor= 1.0,
    interactionProperty= 'IntProp-Cyclic',
    initialClearance= OMIT,
    datumAxis= None,
    clearanceRegion= None)
else:
    mM.SurfaceToSurfaceContactExp(name = name1 + '-' + str(count), createStepName='Initial',
    slave = slaveRegion,
    master = masterRegion,
    mechanicalConstraint= KINEMATIC, sliding= SMALL,
    interactionProperty= 'IntProp-Cyclic',
    initialClearance= OMIT,
    datumAxis= None,
    clearanceRegion= None)

mM.interactions[name1 + '-' + str(count)].setValuesInStep(  
    stepName= 'Loading-Step', interactionProperty= 'IntProp-Loading')

elif line[0] == '601_Beam Interactions':
    interactionNum = interactionNum + 1
    Points = line[1]

    if interactionNum > 4:
        pass  # break

    print 'Beam Interaction: ' + str(Points)
    if Points[0][0] == Points[1][0]:
        # Y line
        self.setFromCriteria(mM.parts['Beam'].edges, 'Interaction-Concrete-' + str(interactionNum),
            object=mM.parts['Beam'], ymin=Points[0][1], ymax=Points[1][1], x=Points[0][0])
        self.setFromCriteria(mM.parts['Plates'].edges, 'Interaction-Steel-' + str(interactionNum),
            object=mM.parts['Plates'], ymin=Points[0][1], ymax=Points[1][1], x=Points[0][0])

    else:
        # x line
        self.setFromCriteria(mM.parts['Beam'].edges, 'Interaction-Concrete-' + str(interactionNum),
            object=mM.parts['Beam'], xmin=Points[0][0], xmax=Points[1][0], y=Points[0][1])
        self.setFromCriteria(mM.parts['Plates'].edges, 'Interaction-Steel-' + str(interactionNum),
            object=mM.parts['Plates'], xmin=Points[0][0], xmax=Points[1][0], y=Points[1][1])

    slaveRegion = a.Surface(sideEdges=a.instances['Beam-2'].sets['Interaction-Concrete-'] + str(interactionNum).edges, name='Surface-' + 'Interaction-Concrete-' + str(interactionNum))
    self.addToSet( 'Interaction-SlaveElements', slaveRegion.elements)

    masterRegion = a.Surface(sideEdges=a.instances['Plates-1'].sets['Interaction-Steel-'] + str(interactionNum).edges, name='Surface-' + 'Interaction-Steel-' + str(interactionNum))

    # Kinematic contact vs. penalty contact
    # = Kinematic contact
    # The default kinematic contact formulation achieves precise compliance with the contact conditions.
    # It works well in most cases, but some problems with chattering contact may work more easily using penalty contact.
```

372
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

# Cannot model rigid-to-rigid contact.
# – Penalty contact
# The penalty contact algorithm provides less stringent enforcement of contact constraints than the
kinematic algorithm.
# The penalty algorithm allows for treatment of more general types of contact; for example, contact
between two rigid bodies.
# Since the penalty algorithm introduces additional stiffness behavior into a model, this stiffness
can influence the stable time increment.

c = 1 # Penalty
#c = 2 # Kinematic
if c == 1:
    myM. SurfaceToSurfaceContactExp (name = 'Interaction + str(interactionNum), createStepName='
Initial',
    slave = slaveRegion,
    master = masterRegion,
    mechanicalConstraint= PENALTY, sliding=FINITE,
    weightingFactorType= SPECIFIED, weightingFactor=1.0,
    interactionProperty= 'ShearConnection - Props',
    initialClearence= CMT,
    datumAxis= None,
    clearanceRegion= None)
else:
    myM. SurfaceToSurfaceContactExp (name = 'Interaction + str(interactionNum), createStepName='
Initial',
    slave = slaveRegion,
    master = masterRegion,
    mechanicalConstraint= KINEMATIC, sliding= SMALL,
    interactionProperty= 'ShearConnection - Props',
    initialClearence= CMT,
    datumAxis= None,
    clearanceRegion= None)

a. regenerate()

########
#Shear Connectors
########

myM. ConnectorSection (name= 'Damper', translationalType= AXIAL)
myM. ConnectorSection (name= 'Damper', assembledType= CYLINDRICAL)
elastic_0 = connectorBehavior. ConnectorElasticity (components=(1,), table=((0.00001,),))
damping_1 = connectorBehavior. ConnectorDamping (components=(1,), table=((0.2,),))
friction_0 = connectorBehavior. ConnectorFriction (table=((10.0,),),
    frictionModel=USER_CUSTOMIZED, tangentDirection=1)
friction_0. TangentialBehavior (table=((0.5,),))

#friction_0 = connectorBehavior. ConnectorFriction (table=((10.0,),), frictionModel=PREDEFINED)
#friction_0. TangentialBehavior (table=((0.5,),))

myM. sections[ 'Damper' ]. setValues (behaviorOptions = [elastic_0,
    damping_1,
    friction_0,
    ])

myM. sections[ 'Damper' ]. setValues (behaviorOptions = (elastic_0, damping_1,))

#Rebar Slip
#Rebar Slip
myM. ConnectorSection (name= 'Rebar-Slip', translationalType= CARTESIAN, rotationalType= ROTATION)
#RebarSlip = 'Fixed'
#RebarSlip = 'Reduced'
#RebarSlip = 'None'

if RebarSlip == 'Reduced':
elastic_2 = connectorBehavior. ConnectorElasticity (components=(1,3), behavior=RIGID)
elastic_0 = connectorBehavior. ConnectorElasticity (components=(2,), table=((self.Ect/1.0,),))
myM. sections[ 'Rebar-Slip' ]. setValues (behaviorOptions = [elastic_0, elastic_2])
elif RebarSlip == 'None':
    # Stress and strain is more uniform in bar, but resistance drops, because force is not transferred into
    the concrete
    elastic_2 = connectorBehavior. ConnectorElasticity (components=(1,3,4,5,6), behavior=RIGID)
    myM. sections[ 'Rebar-Slip' ]. setValues (behaviorOptions = [elastic_2])

373
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
elif rebarSlip == 'Fixed':
    elastic_2 = connectorBehavior.ConectorElasticity(components=(1,2,3,4,5,6), behavior=RIGID)
    myM.sections['Rebar-Slip'].setValues(behaviorOptions=(elastic_2,))

# Link studs
studNo = 0
self.linkPoints = []
for row in self.studPositions:
    studNo = studNo + 1
    Points = row[0]
    studType = row[1]
    studNum = row[2]
    studEnds = row[3]

    self.createShearConnection(studEnds, Points, a.instances['Beam-1'], a.instances['Plates-1'], a.instances['Stud-' + str(studNo)])

session.viewports['Viewport: 1'].setValues(displayedObject=a)
session.viewports['Viewport: 1'].assemblyDisplay.setValues({interactions=OFF, constraints=OFF, connectors=OFF, engineeringFeatures=OFF})

studNo = 0
self.WirePositions = []
# Reference points renumber wires
for row in self.studPositions:
    studNo = studNo + 1
    Points = row[0]
    studType = row[1]
    studNum = row[2]
    studEnds = row[3]

    self.createShearConnectionWires(studEnds, Points, a.instances['Beam-1'], a.instances['Plates-1'], a.instances['Stud-' + str(studNo)])

# Start with one set then change
for wire in a.edges:
    if wire.featureName.find('LinkWireBase')>-1:
        studName = wire.featureName.replace('LinkWireBase-', '')
        studName = studName.replace('Stud-', '')
        print('Adding to: BaseWire' + str(wire.index))
        self.addToList('BaseWire-' + str(wire.index), a.edges[wire.index:wire.index+1])

    csa = a.SectionAssignment(sectionName='Spring-' + studName.replace('end1', '').replace('end2', ''), region=a.sets['BaseWire-' + str(wire.index)])
    a.ConnectorOrientation(region=csa.getSet(), localCsys=a.datums[a.features['Csys-' + studName].id])

newEdge1 = None
for newEdge1 in a.edges:
    if newEdge1.featureName.find('LinkWire-')>-1:
        if newEdge1.featureName.find('-')>-1:
            studName = newEdge1.featureName.replace('LinkWire-','')
            studName = studName.replace('Stud-','')
            print('Adding to: Concrete-Spring-' + studName)
            self.addToList('Concrete-Spring-' + studName, a.edges[newEdge1.index:newEdge1.index+1])

studNo = 0
for studEntry in self.studPositions:
    studNo = studNo + 1
    print('Assigning: Concrete-Spring-Stud-' + str(studNo))
    csa = a.SectionAssignment(sectionName='Rebar-Slip', region=a.sets['Concrete-Spring-Stud-' + str(studNo)])
```

374
def CreateLoadsAndResults(self):

class sparePoint():
    def __init__(self):
        self.xValue = -0.0
        self.yValue = -0.0
        self.zValue = -0.0

# #######
# Loads
# #######

a = mM.rootAssembly
.a.regenerate()

if self.IncludeGravity == True:
    # Gravity disabled for sideways tests
    for time in xrange(int(0.0), int(10000.0), int(1.0)):
        if time <= 0.3 * self.GravityTime:
            Gravity.append((time, time / (0.3 * self.GravityTime)))
        else:
            Gravity.append((time, 1.0))
    mM.TabularAmplitude(data=Gravity, name='Gravity-Loading', smooth=SOLVER_DEFAULT, timeSpan=TOTAL)
    #mM.Gravity(amplitude='Gravity-Loading', name='Gravity', createStepName='Gravity-Step', comp2=-9.81 * self.widthScale, distributionType=UNIFORM, field=' ', region=a.instances['Beam-1'].sets['Beam Areas'])

    mM.Gravity(amplitude='Gravity-Loading', name='Gravity', createStepName='Gravity-Step', comp2=-9.81 * self.widthScale, distributionType=UNIFORM, field=' ', region=regionToolset.Region(faces=a.instances['Beam-1'].faces))

    # Make Gravity Load instant
    #mM.loads['Gravity'].setValues(amplitude=UNSET, distributionType=UNIFORM, field='')
    #a.regenerate()

    # region = a.surfaces['Surface-Interaction-Load-4']
    #mM.Pressure(name='Gravity_Pressure', createStepName='Gravity-Step',
    #region=regionToolset.Region(faces=a.instances['Beam-1'].faces))

    # amplitude='Gravity-Loading')

Cyclic = []

for time in xrange(0, int(self.LoadTime * self.Cycles)),
    int(self.LoadTime)):
    Cyclic.append((time, 0.0))

for inc in xrange(0.05 * 1000.0, 0.95 * 1000.0, 0.025 * 1000.0):
    if inc <= 0.5 * 1000.0:
        Cyclic.append((time + self.LoadTime * inc / 1000.0, inc / 1000.0 + 2.0))
    else:
        Cyclic.append((time + self.LoadTime * inc / 1000.0, 1.0 - (inc / 1000.0 - 0.5) + 2.0))

for time in xrange(int(self.LoadTime * self.Cycles) * 1000.0),
    int(self.LoadTime * self.Cycles) * self.EndCycles * 1000.0),
    int(0.1 * 1000.0)):
    Cyclic.append((time / 1000.0, 0.0))

mM.TabularAmplitude(data=Cyclic, name='Cyclic', smooth=SOLVER_DEFAULT, timeSpan=STEP)

final = []

for inc in xrange(0,1000.1):
    incValue = 0.7143 * math.pow((inc / 1000.0), 2) + 0.2857 + inc / 1000.0
    final.append(((inc / 1000.0) * self.FinalLoadTime, incValue))
final.append((self.FinalLoadTime+1.0,1.0))
final.append((self.FinalLoadTime+100.0,1.0))

#final = []
#final.append((0.0,0.0))
#final.append((1000.0,0.0))
mM TabularAmplitude(data=final, name=‘Final-Loading’, smooth=SOLVER_DEFAULT, timeSpan=STEP)

a.regenerate()

def buildTransferBeams(self, s1):
    p = mM.Part(name=‘Transfer-Beams’, dimensionality=THREE_D_PLANAR, type=DEFORMABLE_BODY)
p.BaseWire(sketch=s1)

mM.RectangularProfile(name=‘Transfer-Beam’, a=10.0, b=10.0)
mM.BeamSection(integration=DURING_ANALYSIS, material=‘Transfer-Beam’,
    name=‘Transfer-Beam’,
    poissonRatio=0.2,
    profile=‘Transfer-Beam’,
    temperatureVar=LINEAR)

self.addToSet(‘Transfer-Beams’, p.edges, p)

region = p.sets[‘Transfer-Beams’]
p.SectionAssignment(region=region, sectionName=‘Transfer-Beam’)
p.assignBeamSectionOrientation(method=N1_COSINES, n1=(0.0, 0.0, -1.0), region=region)
p.seedEdgeBySize(constraint=FINER, deviationFactor=0.1, edges=p.edges, size=100.0) #self.meshScale
p.generateMesh()

def hingeSections(self,a):
    hingeCount = 0
    for wire in a.edges:
        print wire
        if wire.featureName.find(‘Hinge’)>-1:
            p1 = sparePoint()

            coord1 = a.DatumCsysByThreePoints( origin=(p1.xValue, p1.yValue, p1.zValue),
                                                point1=(p1.xValue+1.0, p1.yValue, p1.zValue),
                                                point2=(p1.xValue, p1.yValue+1.0, p1.zValue),
                                                name=‘Hinge csys’, coordSysType=CARTESIAN)
            datum1 = a.datums[coord1.id]

            hingeCount = hingeCount + 1
            self.addToSet(‘Load-Hinge-’ + str(hingeCount), a.edges[wire.index:wire.index+1])
            self.addToSet(‘Load-Hinges-’ + str(p1.yValue), a.edges[wire.index:wire.index+1])

            self.addToSet(‘Load-Hinges’, a.edges[wire.index:wire.index+1])

    #try:
    csa = a.SectionAssignment(sectionName=‘Transfer-Hinge’, region=a.sets[‘Load-Hinge-’ + str(hingeCount)])
    a.ConnectorOrientation(region=csa.getSet(), localCsys1=datum1)
    #except AbaqusException:
    #pass

#Spring for support
if self.Case_Settings.find(‘SpringBases’)>-1:
    self.supportLoadDef = []
    self.SupportScale = 1.0

    self.supportLoadDef.append((−10000000.0,−0.0031*self.SupportScale))
    self.supportLoadDef.append((−4000000.0,−0.003*self.SupportScale))
    self.supportLoadDef.append((−1000000.0,−0.002*self.SupportScale))
    self.supportLoadDef.append((0.0,0.0))
    self.supportLoadDef.append((100000.0,0.002*self.SupportScale))
    self.supportLoadDef.append((400000.0,0.003*self.SupportScale))
    self.supportLoadDef.append((10000000.0,0.0031*self.SupportScale))

mM.ConnectorSection(name=‘Support-Spring’, translationalType=CARTESIAN, rotationalType=ROTATION)
elastic_2 = connectorBehavior.ConectorElasticity(components=(1,3), behavior=RED)
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
elastic_0 = connectorBehavior.ConnectorElasticity(components=(2,), behavior=NONLINEAR, table=self.supportLoadDef)
step_2 = connectorBehavior.ConnectorStop(components=(2,), minMotion=0.004, maxMotion=0.004)
myM = self.sections['Support-Spring'].setValues(bbehaviorOptions=(elastic_0, elastic_2, ))
myMRectangularProfile(name='Support-Spring', a=10.0, b=10.0)
myMBeamSection(integration=DURING_ANALYSIS, material='Transfer_Beam', name='Support-Spring-Base', poissonRatio=0.2, profile='Support-Spring', temperatureVar=LINEAR)
supportNum = 0
for line in self.Lines:
  name1 = ""
  if line[0] == "501_Load 1":
    name1 = 'Load-1'
supportOrLoad = 'Load'
  elif line[0] == "502_Load 2":
    name1 = 'Load-1a'
supportOrLoad = 'Load'
  elif line[0] == "551_Supports":
    supportNum = supportNum + 1
    name1 = 'Support-' + str(supportNum)
supportOrLoad = 'Support'
  if name1 != "":
    Points = line[1]
x = a.instances[name1].sets['PointLoad'].nodes[0].coordinates[0]
y = a.instances[name1].sets['PointLoad'].nodes[0].coordinates[1]

    refPointLD = a.ReferencePoint(point=(x, y, 0.0))
    a.features.changeKey(fromName=refPointLD.name, toName='RP-'+name1)

    #Create pin in instance
    #Node still has rotational stiffness if an element is joined to it
    region1=a.instances[name1].sets['PointLoad']
    myM.RigidBody(name='Pin-' + name1, refPointRegion=region1, pinRegion=a.instances[name1].sets['Bottom Edges'])
    dx = Points[1][0] - Points[0][0]
dy = Points[1][1] - Points[0][1]
    angle = self.rotationAngle(dy, dx)

    import math
    #angle = angle - 90.0
    angled = angle + 90.0
    angle2 = math.radians(angled)
    angle3 = math.radians(angle - 90.0)
    coord1 = a.DatumCsysByThreePoints(origin=(x, y, 0.0),
    point1=(x*math.cos(angle), y*math.sin(angle), 0.0),
    point2=(x*math.cos(angle2), y*math.sin(angle2), 0.0),
    name='Csys-'+name1, coordSysType=CARTESIAN)
    datum1 = a.datums[coord1.id]

    if supportOrLoad == 'Support':
      if self.Case_Settings.find('SpringBases')>-1:
        #Create a small rod. Will take 0 force, just provides a node
        s1 = myM.ConstrainedSketch(name='Support-Spring-' + name1, sheetSize=200.0)
        s1.Line(point1=(x*math.cos(angle3)*0.004, y*math.sin(angle3)+0.004), point2=(x*math.cos(angle3)*0.000, y*math.sin(angle3)*0.000))
        pSupport = myM.Part(name='Support-Spring-' + name1, dimensionality=TWO_D_PLANAR, type=DEFORMABLE_BODY)
        pSupport.BaseWire(sketch=s1)
        pSupport.seedEdgeBySize(constraint=FINER, deviationFactor=0.1, edges=pSupport.edges, size=100.0)
```

377
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
self.addToSet('Support-Spring-' + name1 + '-Edges', pSupport.edges, pSupport)
region = pSupport.sets['Support-Spring-' + name1 + '-Edges']
pSupport.SectionAssignment(region=region, sectionName='Support-Spring-Base')
pSupport.assignBeamSectionOrientation(method=N1_COSINES, n1=(0.0, 0.0, -1.0), region=region)
pSupport.generateMesh()
self.addToSet('SupportPoint', pSupport.nodes[1:2], pSupport)
self.addToSet('SupportPoint', pSupport.vertices[1:2], pSupport)
a. Instance(name='Support-Spring-' + name1, part=myM.parts['Support-Spring-' + name1], dependent=ON)
supportPoint = a.instances['Support-Spring-' + name1].sets['SupportPoint']

# Springs
if self.version == '6.14-3':
    Spring1 = a.WirePolyLine(points=((supportPoint.vertices[0], a.instances[name1].sets['PointLoadV'].vertices[0]),), mergeType=IMPRINT, meshable=OFF)
else:
    Spring1 = a.WirePolyLine(points=((supportPoint.vertices[0], a.instances[name1].sets['PointLoadV'].vertices[0]),), mergeWire=ON, meshable=OFF)
a.features.changeKey(fromName=Spring1.name, toName='Support-Spring-Wire-' + name1)
for wire in a.edges:
    if wire.featureName.find('Support-Spring-Wire-' + name1) >= 1:
        print(wire.featureName)
        csa = a.SectionAssignment(sectionName='Support-Spring', region=regionToolset.Region(edges=a.edges[wire.index:wire.index + 1]))

region1 = supportPoint

myM.DisplacementBC(amplitude=UNSET, createStepName='Initial', distributionType=UNFORM, fieldName='', localCsys=datum1, name=name1 + '-pin-x', region=region1, a.instances[name1].sets['PointLoad'], u1=SET, u2=UNSET, u3=UNSET)
myM.DisplacementBC(amplitude=UNSET, createStepName='Initial', distributionType=UNFORM, fieldName='', localCsys=datum1, name=name1 + '-pin-y', region=region1, a.instances[name1].sets['PointLoad'], u1=UNSET, u2=SET, u3=UNSET)
else:
    pass
if name1 == 'Load-1':
datumLoad1 = datum
print(self.LoadArrangement)
if self.LoadArrangement == '1 Point Load':
    region1=a.instances['Load-1'].sets['PointLoad']

# Point load location used later
regionPointLoad = region1

# Cyclic step
myM.DisplacementBC(amplitude='Final-Loading', createStepName='Loading-Step', distributionType=UNFORM, fieldName='', localCsys=datumLoad1, name='PointLoad-1', region=regionPointLoad, a.instances[name1].sets['PointLoad'], u1=UNSET, u2=self.appliedDeflection, u3=UNSET)

# X-restraint of load
myM.DisplacementBC(amplitude=UNSET, createStepName='Initial', distributionType=UNFORM, fieldName='', localCsys=datumLoad1, name='PointLoad-1-x', region=regionPointLoad, u1=SET, u2=UNSET, u3=UNSET)
else:
    self.LoadArrangement = '2 Point Loads' or self.LoadArrangement == '2 Opposite Point Loads':
    region1=a.instances['Load-1'].sets['PointLoad']
    region1a=a.instances['Load-1a'].sets['PointLoad']
x1 = region1.nodes[0].coordinates[0]
y1 = region1.nodes[0].coordinates[1]
```

378
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
x2 = region1a.nodes[0].coordinates[0]
y2 = region1a.nodes[0].coordinates[1]

self.loadSpacing1 = self.loadFactor1 / (self.loadFactor1 + self.loadFactor1a)
self.loadSpacing1x = x1 + (x2 - x1) * (1.0 - self.loadSpacing1)
self.loadSpacing1y = y1 + (y2 - y1) * (1.0 - self.loadSpacing1)

# Create Transfer Beam Part
s1 = myM.ConstrainedSketch(name='TBeam', sheetSize=200.0)
s1.Line(point1=(x1, y1), point2=(self.loadSpacing1x, self.loadSpacing1y))
s1.Line(point1=(x2, y2), point2=(self.loadSpacing1x, self.loadSpacing1y))
s1.Line(point1=(x1, self.loadSpacing1y), point2=(self.loadSpacing1x, self.loadSpacing1y))

p = myM.Part(name='Transfer-Beams', dimensionality=TWO_D_PLANAR, type=DEFORMABLE_BODY)
p.BaseWire(sketch=s1)

myM.RectangularProfile(name='Transfer-Beam', a=10.0, b=10.0)

myM.BeamSection(integration=DURING_ANALYSIS,
material='Transfer_Beam',
name='Transfer-Beam',
poissonRatio=0.2,
profile='Transfer-Beam',
temperatureVar=LINEAR)

self.addToSet('Transfer-Beams', p.edges, p)

region = p.sets['Transfer-Beams']
p.SectionAssignment(region=region, sectionName='Transfer-Beam')
p.assignBeamSectionOrientation(method=N1_COSINES, n1=(0.0, 0.0, -1.0), region=region)
p.seedEdgeBySize(constraint=FINER, deviationFactor=0.1, edges=p.edges, size=100.0) #self.meshScale
p.generateMesh()

# Assembly Hinges
if self.version == '6.14-3':
    Hinge1 = a.WirePolyLine(points=[(a.instances['Transfer-Beams'].sets['L1'].vertices[0], a.instances['Load-1'].sets['PointLoadV'].vertices[0]),], mergeType=IMPRINT, meshable=OFF)
    Hinge2 = a.WirePolyLine(points=[(a.instances['Transfer-Beams'].sets['L2'].vertices[0], a.instances['Load-1a'].sets['PointLoadV'].vertices[0]),], mergeType=IMPRINT, meshable=OFF)
else:
    Hinge1 = a.WirePolyLine(points=[(a.instances['Transfer-Beams'].sets['L1'].vertices[0], a.instances['Load-1'].sets['PointLoadV'].vertices[0]),], mergeWire=ON, meshable=OFF)
    Hinge2 = a.WirePolyLine(points=[(a.instances['Transfer-Beams'].sets['L2'].vertices[0], a.instances['Load-1a'].sets['PointLoadV'].vertices[0]),], mergeWire=ON, meshable=OFF)

a.features.changeKey(fromName=Hinge1.name, toName='Load-1-Hinge')
a.features.changeKey(fromName=Hinge2.name, toName='Load-2-Hinge')

myM.ConnectorSection(name='Transfer-Hinge', translationalType=JOIN)
myM.sections['Transfer-Hinge'].setValues(integration=EXPLICIT)

for wire in a.edges:
    print wire
    if wire.featureName.find('Hinge') > -1:
        p1 = sparePoint()

coord1 = a.DatumCsysByThreePoints(origin=(p1.xValue, p1.yValue, p1.zValue), point1=(p1.xValue+1.0, p1.yValue, p1.zValue), point2=(p1.xValue, p1.yValue+1.0, p1.zValue), name='Hinge csys', coordSysType=CARTESIAN)
datum1 = a.datums[coord1.id]
```

379
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
try:
    csa = a.SectionAssignment(sectionName='Transfer-Hinge', region=regionToolset.Region(edges=a.edges[wire.index:wire.index+1]))
    a.ConnectorOrientation(region=csa.getSet(), localCsys1=datum1)
except AbaqusException:
    pass

self.addToSet('Load-Hinges', a.edges[wire.index:wire.index+1])

if wire.featureName.find('Load-1-Hinge') > -1:
    self.addToSet('Load-1-Hinge', a.edges[wire.index:wire.index+1])

if self.LoadArrangement == '2 Point Loads':
    # Point load location used later
    regionPointLoad = a.instances['Transfer-Beams'].sets['LMid']

    # Point Load
    myM.DisplacementBC(amplitude='Final-Loading', createStepName='Loading-Step', distributionType=UNIFORM, fieldName=' ', localCsys=a.datums[a.features['Csys-Load-1'].id], name='PointLoad-1', region=regionPointLoad, u1=UNSET, u2=self.appliedDeflection, ur3=UNSET)

    # Restrains of load
    myM.DisplacementBC(amplitude=UNSET, createStepName='Initial', distributionType=UNIFORM, fieldName='', localCsys=a.datums[a.features['Csys-Load-1'].id], name='PointLoad-1-x', region=regionPointLoad, u1=SET, u2=UNSET, ur3=UNSET)

if self.LoadArrangement == '2 Opposite Point Loads':
    # Point load location used later
    regionPointLoad = a.instances['Transfer-Beams'].sets['LEnd']

    # Point Load
    myM.DisplacementBC(amplitude='Final-Loading', createStepName='Loading-Step', distributionType=UNIFORM, fieldName=' ', localCsys=a.datums[a.features['Csys-Load-1'].id], name='PointLoad-1', region=regionPointLoad, u1=UNSET, u2=self.appliedDeflection, ur3=UNSET)

    # Fulcrum
    myM.DisplacementBC(amplitude=UNSET, createStepName='Initial', distributionType=UNIFORM, fieldName='', localCsys=a.datums[a.features['Csys-Load-1'].id], name='Fulcrum', region=a.instances['Transfer-Beams'].sets['LMid'], u1=SET, u2=SET, ur3=UNSET)

elif self.LoadArrangement == '4 Point Loads':
    regions=[]
    regions.append(0)
    regions.append(a.instances['Load-1']
    regions.append(a.instances['Load-1a'])
    regions.append(a.instances['Load-1b'])
    regions.append(a.instances['Load-1c'])

    y1 = regions[1].nodes[0].coordinates[1]+0.008
    y2 = y1 + 0.005

    p1 = regions[1].sets['PointLoad'].nodes[0].coordinates[0]
    p2 = regions[2].sets['PointLoad'].nodes[0].coordinates[0]
    p3 = regions[3].sets['PointLoad'].nodes[0].coordinates[0]
    p4 = regions[4].sets['PointLoad'].nodes[0].coordinates[0]

    p12 = (p1+p2)/2.0
    p34 = (p3+p4)/2.0
    p1234 = (p12+p34)/2.0
```

# Create Transfer Beam Part
s1 = myM.ConstrainedSketch(name='TBeam', sheetSize=200.0)
s1.Line(point1=(p1,y1), point2=(p12,y1))

380
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
s1.Line(point1=(p2, y1), point2=(p12, y1))
s1.Line(point1=(p3, y1), point2=(p34, y1))
s1.Line(point1=(p4, y1), point2=(p34, y1))
s1.Line(point1=(p12, y2), point2=(p1234, y2))
s1.Line(point1=(p34, y2), point2=(p1234, y2))
buildTransferBeams(self, s1)
a.Instance(name='Transfer-Beams', part=a.instances['Transfer-Beams'])
p = a.instances['Transfer-Beams']
#v1 = self.findClosest(p1, y1, p.vertices)
#p.Set(vertices=p.vertices[v1.index:v1.index+1], name='L1')
hingeLocs = []
for plate in regions:
    if plate != 0:
        hingeLocs.append((self.findClosest(plate.sets['PointLoad'].nodes[0].coordinates[0], y1, p.vertices), plate.sets['PointLoadV'].vertices[0]))

hingeLocs.append((self.findClosest(p12, y1, p.vertices), self.findClosest(p12, y2, p.vertices)))

hingeLocs.append((self.findClosest(p34, y1, p.vertices), self.findClosest(p34, y2, p.vertices)))

mM.ConnectorSection(name='Transfer-Hinge', translationalType=JOIN)
mM.sections['Transfer-Hinge'].setValues(integration=EXPLICIT)

hingeNo = 0
for hingeLoc in hingeLocs:
    if plate != 0:
        if self.version == '6.14-3':
            Hinge1 = a.WirePolyLine(points=(hingeLoc[0], hingeLoc[1],), mergeType=IMPRINT, meshable=OFF)
        else:
            Hinge1 = a.WirePolyLine(points=(hingeLoc[0], hingeLoc[1],), mergeWire=ON, meshable=OFF)
        hingeNo = hingeNo + 1
        a.features.changeKey(fromName='Hinge1.name', toName='Hinge-' + str(hingeNo))

hingeSections(self, a)
v1 = self.findClosest(p1234, y2, p.vertices)
a.Set(vertices=p.vertices[v1.index:v1.index+1], name='LMid')

#Point load location used later
regionPointLoad = a.sets['LMid']

#Point Load
mM.DisplacementBC(amplitude='Final-Loading', createStepName='Loading-Step', distributionType=UNIFORM, fieldNames='-', localCsys=a.datums[a.features['Csys-Load-1'].id], name='PointLoad-1', region=regionPointLoad, a.instances[name].sets['PointLoad'], u1=UNSET, u2=self.appliedDeflection, ur3=UNSET)

#X-restraint of load
mM.DisplacementBC(amplitude=UNSET, createStepName='Initial', distributionType=UNIFORM, fieldNames='-', localCsys=a.datums[a.features['Csys-Load-1'].id], name='PointLoad-1-x', region=regionPointLoad, u1=SET, u2=UNSET, ur3=UNSET)

if self.LoadArrangement == '8 Point Loads':
    regions = []
    regions.append(0)
    regions.append(a.instances['Load-1'])
    regions.append(a.instances['Load-1a'])
    regions.append(a.instances['Load-1b'])
    regions.append(a.instances['Load-1c'])
    regions.append(a.instances['Load-1d'])
    regions.append(a.instances['Load-1e'])
    regions.append(a.instances['Load-1f'])
    regions.append(a.instances['Load-1g'])
y1 = regions[1].nodes[0].coordinates[1] + 0.008
y2 = y1 + 0.005
```
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

y3 = y2 + 0.005

p1 = regions[1].sets["PointLoad"].nodes[0].coordinates[0]
p2 = regions[2].sets["PointLoad"].nodes[0].coordinates[0]
p3 = regions[3].sets["PointLoad"].nodes[0].coordinates[0]
p4 = regions[4].sets["PointLoad"].nodes[0].coordinates[0]
p5 = regions[5].sets["PointLoad"].nodes[0].coordinates[0]
p6 = regions[6].sets["PointLoad"].nodes[0].coordinates[0]
p7 = regions[7].sets["PointLoad"].nodes[0].coordinates[0]
p8 = regions[8].sets["PointLoad"].nodes[0].coordinates[0]
p12 = (p1+p2)/2.0
p34 = (p3+p4)/2.0
p56 = (p5+p6)/2.0
p78 = (p7+p8)/2.0
p1234 = (p12+p34)/2.0
p5678 = (p56+p78)/2.0
p12345678 = (p1234+p5678)/2.0

# Create Transfer Beam Part
s1 = myM.ConstrainedSketch(name=’TBeam’, sheetSize=200.0)
s1.Line(point1=(p1, y1), point2=(p12, y1))
s1.Line(point1=(p2, y1), point2=(p12, y1))
s1.Line(point1=(p3, y1), point2=(p34, y1))
s1.Line(point1=(p4, y1), point2=(p34, y1))
s1.Line(point1=(p5, y1), point2=(p56, y1))
s1.Line(point1=(p6, y1), point2=(p56, y1))
s1.Line(point1=(p7, y1), point2=(p78, y1))
s1.Line(point1=(p8, y1), point2=(p78, y1))
s1.Line(point1=(p12, y2), point2=(p1234, y2))
s1.Line(point1=(p34, y2), point2=(p1234, y2))
s1.Line(point1=(p56, y2), point2=(p5678, y2))
s1.Line(point1=(p78, y2), point2=(p5678, y2))
s1.Line(point1=(p1234, y3), point2=(p12345678, y3))
s1.Line(point1=(p5678, y3), point2=(p12345678, y3))
buildTransferBeams(self, s1)
a.Instance(name=’Transfer−Beams’, part=myM.parts[’Transfer−Beams’], dependent=ON)
p = a.instances[’Transfer−Beams’]

#v1 = self.findClosest(p1, y1, p.vertices)
#p.Set(vertices=p.vertices[1.i n d e x :1.i n d e x +1], name=’L1’)

hingeLocs = []
for plate in regions:
    if plate != 0:
        hingeLocs.append((self.findClosest(plate.sets[’PointLoad’].nodes[0].coordinates[0], y1, p.vertices), plate.sets[’PointLoadV’].vertices[0]))

hingeLocs.append((self.findClosest(p12, pl ytes), self.findClosest(p12, pvertices)))

hingeLocs.append((self.findClosest(p34, pvertices), self.findClosest(p34, pvertices)))

hingeLocs.append((self.findClosest(p56, pvertices), self.findClosest(p56, pvertices)))

hingeLocs.append((self.findClosest(p78, pvertices), self.findClosest(p78, pvertices)))

hingeLocs.append((self.findClosest(p1234, pvertices), self.findClosest(p1234, pvertices)))

hingeLocs.append((self.findClosest(p3456, pvertices), self.findClosest(p3456, pvertices)))

myM.ConectorSection(name=’Transfer−Hinge’, translationalType=JOIN)
myM.sections[’Transfer−Hinge’].setValues(integration=EXPLICIT)

hingeNo = 0
for hingeLoc in hingeLocs:
    if hingeLoc != 0:
        if self.version == ’6.14−3’:

382
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

Hinge1 = a.WirePolyLine(points=(hingeLoc[0], hingeLoc[1], ), mergeType=IMPRESS, meshable=OFF)
else:
    Hinge1 = a.WirePolyLine(points=(hingeLoc[0], hingeLoc[1], ), mergeWire=ON, meshable=OFF)
    hingeNo = hingeNo + 1
    a.features.changeKey(fromName=Hinge1.name, toName=’Hinge−’ + str(hingeNo) )

hingeSections(self, a)

v1 = self.findClosest(p12345678, p.vertices)
a.Set(vertices=p.vertices[v1.index:v1.index+1], name=’LMid’)

#Point load location used later
regionPointLoad = a.sets[’LMid’]

#Point Load
mMyM.DisplacementBC(amplitude=’Final−Loading’, createStepName=’Loading−Step’, distributionType=UNIFORM, fieldName='', localCsys=a.datums[a.features[’Csys−Load−1’].id].name, name=’PointLoad−1’, region=regionPointLoad, a.instances[’name1’].sets[’PointLoad’], u1=UNSET, u2=self.appliedDeflection, ur3=UNSET)

#X-restraint of load
mMyM.DisplacementBC(amplitude=UNSET, createStepName=’Initial’, distributionType=UNIFORM, fieldName='', localCsys=a.datums[a.features[’Csys−Load−1’].id].name, name=’PointLoad−1−x’, region=regionPointLoad, u1=SET, u2=UNSET, ur3=UNSET)

#Assign wires loading beam props

# Load-Def Points

self.loadDefPoints = []
for line in self.Lines:
    if line[0] == ’561_LoadDefs’:
        Points = line[1]
        dx = Points[1][0] – Points[0][0]
        dy = Points[1][1] – Points[0][1]
        angle = self.rotationAngle(dy, dx)
        self.loadDefPoints.append((Points[0][0], Points[0][1], angle, ’ ’))
    if line[0] == ’562_LoadDef_UnderLoad’:
        Offsetx = self.loadOffsets
        Points = line[1]
        dx = Points[1][0] – Points[0][0]
        dy = Points[1][1] – Points[0][1]
        angle = self.rotationAngle(dy, dx)
        self.loadDefPoints.append((Points[0][0]+Offsetx, Points[0][1], angle, ’ ’))

#Part name autonumbers
j = 0
for i in self.loadDefPoints:
    j = j + 1
    angle = i[2]
    x = i[0]
    y = i[1]
    PartName = ’LoadDefPoint’−’ + str(j) + str(angle) + ’ ’−’ + i[3]
    PartName = PartName.replace(’ ’, ’ ’)
    print PartName

383
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
p = mpy.M. Part(name=PartName,
    dimensionality=TWO_D_PLANAR, type=DISCRETE_RIGID_SURFACE)

refPointLD = p. ReferencePoint(point=(x, y, 0.0))
p. features. changeKey(fromName=refPointLD. name, toName=PartName)

i = a. Instance(name="LoadDefPoint−" + str(i) + "]", part=p, dependent=ON)
region2=a. Set(referencePoints=(i. referencePoints[1],), name=PartName)

myM. Coupling(name=PartName, controlPoint=regionPointLoad, surface=region2, influenceRadius=WHOLE_SURFACE, couplingType=KINEMATIC, localCsys=None, u1=ON, u2=ON, ur3=ON)

# ##################
#End slip measurement
# ##################
self. createEndSlipSpring(a. instances["Beam−1"], a. instances["Plates−1"])

# Element Type Sets
# ##################

print a. elements

elementSets = []
elementTypes = ['CPS4R', 'CPS3', 'B21 ']
for elementType in elementTypes:
    instances = re. findall(r" [A-Za-z\-0-9\*]+ASSEMBLY *: " + str(a. instances))
    print instances
    instanceSet = []
    for instance in instances:
        instance = instance. replace("\","")
        instance = instance. replace("\","")

        elemSet = []
        if instance == 'ASSEMBLY':
            loop = a. elements
        else:
            loop = a. instances[instance]. elements

        #print loop
        for element in loop:
            #print element
            #a-b
            if str(element. type) == elementType:
                elemSet. append(element. label)
            else:
                if str(element. type) in elementTypes:
                    pass
                else:
                    print str(element. type)

        if len(elemSet) > 0:
            instanceSet. append((instance, elemSet))
        #print instanceSet
    #print instanceSet
    newSet = a. SetFromElementLabels(elementType, instanceSet)
    elementSets. append(elementType)
    print elementSets

#Results
# ############
self. request1 = 10
myM. FieldOutputRequest(name="F−Output−2", variables=('A', 'CSTRESS', 'EVF', 'LE', 'PE', 'PEEQ', 'PEQAVG', 'PENAVG', 'RF', 'S', 'SWAVG', 'U', 'V'), numIntervals=10)
myM. fieldOutputRequests["F−Output−1"]. setValuesInStep(stepName='Gravity−Step', variables=('U', 'RF', 'A', 'V'), numIntervals=self. CycleIntervals)
myM. fieldOutputRequests["F−Output−1"]. setValuesInStep(stepName='Loading−Step', numIntervals=500)
```

384
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```
VarsName = 'Stresses_etc'

for elementSet in elementTypes:
    vars = []
    foundSet = 0
    if elementSet == 'CPS4R' or elementSet == 'CPS3':
        vars = ['A', 'IE', 'PE', 'PEEQ', 'RF', 'S', 'U', 'V', 'MBSMAX', 'NF', 'INER', 'ELIN', 'IELED', 'EXDEN', 'EDT', 'EMSF', 'DENSITY', 'STATUS',]
        foundSet = 1
    if elementSet == 'B21':
        vars = ['A', 'IE', 'PE', 'PEEQ', 'RF', 'S', 'U', 'V', 'MBSMAX', 'NF', 'INER', 'ELIN', 'IELED', 'EXDEN', 'EDT', 'EMSF', 'DENSITY', 'STATUS',]
        foundSet = 1
# 'CSTRESS', 'CFCOE', 'CTHICK', 'FSLIPR', 'FSLIP', 'PPRESS'
    if foundSet == 1:
        myM. FieldOutputRequest(name=VarsName + '−' + elementSet, createStepName='Gravity−Step', variables=(vars), numIntervals=self.request1, region=a.sets[elementSet])
        myM. fieldOutputRequests[VarsName + '−' + elementSet]. setValuesInStep(stepName='Cyclic−Step', numIntervals=self.CycleIntervals)
        myM. fieldOutputRequests[VarsName + '−' + elementSet]. setValuesInStep(stepName='Loading−Step', numIntervals=500)

myM. historyOutputRequests['H−Output−1']. setValues(
    variables=['ALLAE', 'ALLCD', 'ALLBD', 'ALLFD', 'ALLIE', 'ALLKE', 'ALLPD', 'ALLSE', 'ALLVD', 'ALLWK', 'ETOTAL', 'DT', 'TMASS'],
    numIntervals=self.request1)
myM. historyOutputRequests['H−Output−1']. setValuesInStep(stepName='Cyclic−Step', numIntervals=self.CycleIntervals)
myM. historyOutputRequests['H−Output−1']. setValuesInStep(stepName='Loading−Step', numIntervals=500)

interactions = re.findall(r'\(\[A−Za−z0−9\−]\+\+\+\)', str(myM. interactions))
for interaction in interactions:
    interaction = interaction.replace(' ', '')
```

385
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

interaction = interaction.replace('"', '')

mM.FieldOutputRequest(name=interaction, createStepName='Gravity-Step', variables=('CSTRESS', 'CFORCE', 'CTHICK', 'FSLIP', 'FSLIP'),
    interactions=(interaction, ), sectionPoints=DEFAULT, rebar=EXCLUDE, numIntervals=10)

mM.fieldOutputRequests[interaction].setValuesInStep(stepName='Cyclic-Step', numIntervals=self.CycleIntervals)

mM.fieldOutputRequests[interaction].setValuesInStep(stepName='Loading-Step', numIntervals=500)

instances = re.findall(r'[A-Za-z0-9- ]*: str(a.sets))
for instance in instances:
    instance = instance.replace('"', '')
    instance = instance.replace('"', '')
    #print a.sets[instance]

    if instance.find('Wire-') > -1 or instance.find('Hinge') > -1 or instance.find('LoadSlip') > -1:
        print a.sets[instance]
    #regionDef=a.sets[instance]
    mM.FieldOutputRequest(name=instance, createStepName='Gravity-Step', variables=('CTF', 'CEF', 'CVF', 'CDERU', 'CDERF', 'CU', 'CLE', 'CUF', 'CUP', 'CUPEQ'),
        region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE, numIntervals=self.request1)

mM.fieldOutputRequests[instance].setValuesInStep(stepName='Cyclic-Step', numIntervals=self.CycleIntervals)

mM.FieldOutputRequests[instance].setValuesInStep(stepName='Loading-Step', numIntervals=500)

regionDef=a.sets['Wire-Bottom-y'+str(ConstraintNum)]
regionDef=a.sets['Wire-Bottom-y'+str(ConstraintNum)]
#mM.FieldOutputRequest(name=regionDef, createStepName='Loading-Step', variables=('CTF', 'CEF', 'CVF', 'CDERU', 'CDERF', 'CU', 'CLE', 'CUF', 'CUP', 'CUPEQ'),
#    region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE, numIntervals=500)

# massScaling=
# massScaling=mM.steps['Gravity-Step'].setValues(massScaling=(
#    (SEMI_AUTOMATIC, a.sets['Interaction-SlaveElements'], AT_BEGINNING, self.InterfaceMassScale, 0.0, None),
#    (SEMI_AUTOMATIC, a.sets['Interaction-SlaveElements'], AT_BEGINNING, self.InterfaceMassScale, 0.0, None),
#    ))

# massScaling=
# massScaling=mM.steps['Gravity-Step'].setValues(massScaling=(
    (SEMI_AUTOMATIC, a.sets['Interaction-SlaveElements'], AT_BEGINNING, self.InterfaceMassScale, 0.0, None),
    (SEMI_AUTOMATIC, a.sets['Interaction-SlaveElements'], AT_BEGINNING, self.InterfaceMassScale, 0.0, None),
    ))

# massScaling=
# massScaling=mM.steps['Gravity-Step'].setValues(massScaling=(
    #massScaling=
    #massScaling=mM.steps['Gravity-Step'].setValues(massScaling=(

# Create Job
#
from time import gmtime, strftime

# print self.Model_Name
nowTime = strftime("%Y%m%d%H%M%S", gmtime())
self.Job_Name = self.Model_Name + '_' + nowTime + '_' + self.Case_Name.replace(' ', '_').replace('(', '_').replace(')', '_') + self.Case_Settings
print self.Job_Name

def saveVariablesToODB(self):
    #Cludge for saving variables into ODB file
    mM = mdb.models[self.Model_Name]
    
    #print mM parts
    myPart = mM.parts['Beam']
    print myPart

    self.storVar('Model_Name', self.Model_Name)
    self.storVar('Job_Name', self.Job_Name)
    self.storVar('Case_Name', self.Case_Name)
    self.storVar('shearConnectorLoadSlip', self.shearConnectorLoadSlip)
    self.storVar('shearConnectorTensionSlip', self.shearConnectorTensionSlip)
**APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS**

```python
self.storVar('ConcreteA1Compression', self.ConcreteA1Compression)
self.storVar('ConcreteA1Tension', self.ConcreteA1Tension)
self.storVar('ConcreteA4Compression', self.ConcreteA4Compression)
self.storVar('ConcreteA4Tension', self.ConcreteA4Tension)

import inspect
print.inspect.stack()[0][1]
currentFile = os.path.realpath(inspect.stack()[0][1])

import ntpath
#CurrentDir = os.path.realpath('..')
#CurrentDir = r'/UWS30089/Users/pf00062.SURREY/OneDrive/Surrey'
#CurrentDir = r'//pcv-pxf.sci.local//Users//PXF//OneDrive//Surrey'
copyPath = self.CurrentDir + r'/ABAQUS//Results//' + self.Job_Name + '/'
if not os.path.exists(copyPath):
    os.makedirs(copyPath)

reportPath = copyPath + ntpath.basename(currentFile)

import shutil
shutil.copy2(currentFile, reportPath)

a = myM.rootAssembly
a.regenerate()

#Output a py file with all self variables
#CurrentDir = r'/UWS30089/Users/pf00062.SURREY/OneDrive/Surrey'
#self.CurrentDir = r'//pcv-pxf.sci.local//Users//PXF//OneDrive//Surrey'
copyPath = self.CurrentDir + r'//ABAQUS//Results//' + self.Job_Name + '/'
if not os.path.exists(copyPath):
    os.makedirs(copyPath)
dispFile =open(copyPath + ' VariablesAndVals.py', 'w')
for key in vars(self):
    val = eval('self.' + key)
    if str(type(val)) == "<type 'str'>":
        dispFile.write('self.' + str(key) + ' = ' + str(val) + '
')
    else:
        dispFile.write('self.' + str(key) + ' = ' + str(val) + '
')
dispFile.close()

def CreateJob(self, runModeBoolean):
    if self.version == '6.14-3':
        j = mdb.Job(atTime=None,
                    contactPrint=CN,
                    description='",
                    echoPrint=CN,
                    explicitPrecision=DOUBLE,
                    getMemoryFromAnalysis=True,
                    historyPrint=CN,
                    memory=90,
                    memoryUnits=PERCENTAGE,
                    model=myM,
                    modelPrint=OFF,
                    multiprocessingMode=DEFAULT,
                    name=self.Job_Name,
                    nodalOutputPrecision=SINGLE,
                    numCpus=self.numCores,
                    numDomains=self.numCores,
                    activateLoadBalancing=1,
                    queue=None,
                    scratch=''
                    type=ANALYSIS
```
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
userSubroutine=''
waitHours=0, waitMinutes=0)
else:
j = mdb.Job(atTime=None,
contactPrint=CN,
description=''
echoPrint=CN,
explicitPrecision=DOUBLE,
getMemoryFromAnalysis=True,
historyPrint=CN,
memory=90,
memoryUnits=PERCENTAGE,
model=mM,
modelPrint=OFF,
multiprocessingMode=DEFAULT,
name=self.Job_Name,
nodalOutputPrecision=SINGLE,
umCPUs=self.numCores,
umDomains=self.numCores,
activateLoadBalancing=1,
queue=None,
scratch=''
type=ANALYSIS,
userSubroutine=''
waitHours=0,
waitMinutes=0)

if runModeBoolean == True:
    print 'Running Model'
j.submit(consistencyChecking=OFF)
else:
    print 'Run needs to be Run'
j.writeInput(consistencyChecking=OFF)

def createEndSlipSpring(self, concInstance, plateInstance):
    a = mM.rootAssembly
mM.ConnectorSection(name='LoadSlip', translationalType=AXIAL)
elastic_0 = connectorBehavior.ConnectorElasticity(components=(1,),
    table=((0.00001,),))
mM.sections['LoadSlip'].setValues(behaviorOptions=(elastic_0,))
for i in range(0,4):
    if i == 0:
        x = 0.0
        y = self.bottomt
    elif i == 1:
        x = 0.0
        y = self.beamh - self.top t
    elif i == 2:
        x = self.L
        y = self.bottomt
    elif i == 3:
        x = self.L
        y = self.beamh - self.top t

vertBeam = self.findClosest(x, y, concInstance.vertices.getByBoundingCylinder(center1=(x, y, 0.0),
    center2=(x, y, 1.0), radius=self.meshScale*3.0))
vertPlate = self.findClosest(x, y, plateInstance.vertices.getByBoundingCylinder(center1=(x, y, 0.0),
    center2=(x, y, 1.0), radius=self.meshScale*3.0))
coord1 = a.DatumCsysByThreePoints(origin=(x,y,0.0), point1=(x+1.0, y, 0.0), point2=(x, y+1.0, 0.0),
    name='LoadSlip csys', coordSysType=CARTESIAN)
datum1 = a.datums[coord1.id]
if self.version == '6.14-3':
    springWire = a.WirePolyLine(points=((vertBeam, vertPlate),),
        mergeType=IMPRINT,
        meshable=OFF)
```

388
```python
else:
    springWire = a.WirePolyLine(points=((vertBeam, vertPlate), ),
        mergeWire=OFF,
        meshable=OFF)

#mergeType=IMPRESS

if i == 0:
    a.features.changeKey(fromName=springWire.name, toName='SlipMeasurement-Bottom-0')
elif i == 1:
    a.features.changeKey(fromName=springWire.name, toName='SlipMeasurement-Top-0')
elif i == 2:
    a.features.changeKey(fromName=springWire.name, toName='SlipMeasurement-Bottom-L')
elif i == 3:
    a.features.changeKey(fromName=springWire.name, toName='SlipMeasurement-Top-L')

i = 0

for wire in a.edges:
    if wire.featureName.find('SlipMeasurement') > -1:
        i = i + 1
        self.addToSet('LoadSlipSet', a.edges[wire.index:wire.index + 1])
        self.addToSet('LoadSlipSet' + str(i), a.edges[wire.index:wire.index + 1])

csa = a.SectionAssignment(sectionName='LoadSlip', region=a.sets['LoadSlipSet'])
a.ConnectorOrientation(region=csa.getSet(), localCsys1=datum1)

def createShearConnection(self, studEnds, studPoints, concInstance, plateInstance, studInstance):
    a = myM.rootAssembly
    vertList = studInstance.vertices

    baseLocs = []
    if studEnds == 1:
        baseLocs.append((studPoints[2][0], studPoints[2][1]))
    elif studEnds == 2:
        baseLocs.append((studPoints[3][0], studPoints[3][1]))
        baseLocs.append((studPoints[0][0], studPoints[0][1]))

    end = 0
    for baseLoc in baseLocs:
        end = end + 1
        x = baseLoc[0]
        y = baseLoc[1]

        vertBeam = self.findClosest(x, y, concInstance.vertices.getByBoundingCylinder(center1=(x, y, 0.0),
            center2=(x, y, 1.0), radius=self.meshScale*3.0))
        vertStud = self.findClosest(x, y, studInstance.vertices.getByBoundingCylinder(center1=(x, y, 0.0),
            center2=(x, y, 1.0), radius=self.meshScale*3.0))

        print 'studInstance'
        vertPlate = self.findClosest(x, y, plateInstance.vertices.getByBoundingCylinder(center1=(x, y, 0.0),
            center2=(x, y, 1.0), radius=self.meshScale*3.0))

        dx = studPoints[1][0] - studPoints[0][0]
        dy = studPoints[1][1] - studPoints[0][1]

        import math
        angled = self.rotationAngle(dy, dx)
        angled = angled + 90.0

        if end == 2:
            angled = angled + 180.0

        angle = math.radians(angled)
        angle2 = math.radians(angled + 90.0)

        coord1 = a.DatumCsys3ByThreePoints(origin=(x, y, 0.0),
            point1=(x + math.cos(angle), y + math.sin(angle), 0.0),
            point2=(x + math.cos(angle2), y + math.sin(angle2), 0.0),
            name='Csys-' + str(studInstance.name) + '-end' + str(end), coordSysType=CARTESIAN)

        self.datum1 = a.datums[coord1.id]
```

APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUUS

# Springs Stud Base

```python
if self.version == '6.14-3':
    springWire = a.WirePolyLine(points=((vertStud, vertPlate),),
                              mergeType=IMPRINT,
                              meshable=OFF)
else:
    springWire = a.WirePolyLine(points=((vertStud, vertPlate),),
                              mergeWire=OFF,
                              meshable=OFF)
    #mergeType=IMPRINT

a.features.changeKey(fromName=springWire.name, toName='LinkWireBase-' + str(studInstance.name)+'-end' + str(end))

springWireName = springWire.name

name0 = 'Inertia-Base-' + str(studInstance.name) + '-end' + str(end)
a.Set(name=name0, vertices=studInstance.vertices[vertPlate.index:vertPlate.index+1])

# Constraint at base of stud in plates
# Vert is in plate, linked to vert stud in each case
# Only one per location

verts1 = plateInstance.vertices[vertPlate.index:vertPlate.index+1]

name1 = 'Inertia-Base-' + str(plateInstance.name) + '-' + str(vertPlate.index) + '-' + str(studInstance.name)

foundSet = False
try:
    a.sets['Constraint-' + plateInstance.name + '-' + str(vertPlate.index)]
    print 'Found Constraint-' + plateInstance.name + '-' + str(vertPlate.index)
    foundSet = True
except KeyError:
    print 'Not Found Constraint-' + plateInstance.name + '-' + str(vertPlate.index)

if foundSet == False:
    restraintFaces = a.Set(name=name1 + '-faces', faces=plateInstance.faces.getByBoundingCylinder(center1=(x, y, 0.0),center2 = (x, y, 1.0),radius=self.meshScale*3.0))

    a.Set(vert1s=verts1, name='Constraint-' + plateInstance.name + '-' + str(vertPlate.index))

    constraint = myM.Coupling(name='Constraint-' + plateInstance.name + '-' + str(vertPlate.index),
                               controlPoint=a.sets['Constraint-' + plateInstance.name + '-' + str(vertPlate.index)],
                               surface=restraintFaces,
                               influenceRadius=0.005,SURFACE, couplingType=DISTRIBUTING, DISTRIBUTING, 'Fracture localCsys=NONE, u1=ON, u2=ON, ur3=OFF')

# Damper at spring base

datumCoords = (x-math.cos(angle2)*0.25, y-math.sin(angle2)*0.25, 0.0)

refPoint1 = a.ReferencePoint(point=datumCoords)

#a.features.changeKey(fromName=refPoint1.name, toName='RP-Damper-' + studInstance.name + '-' + str(end))

ConstraintNum = 0

print studInstance.vertices
for j in studInstance.vertices:
    ConstraintNum = ConstraintNum + 1

x = j.pointOn[0][0]
y = j.pointOn[0][1]

nodeSet = myM.parts['Beam'].nodes.getByBoundingCylinder(center1=(x, y, 0.0),center2 = (x, y, 1.0),
                           radius=self.meshScale*1.0)
if len(nodeSet)<1:
```
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
nodeSet = myM.parts['Beam'].nodes.getByBoundingCylinder(center1=(x, y, 0.0), center2=(x, y, 1.0),
radius=self.meshScale*3.0)
if len(nodeSet)<1:
    raise ValueError('Node selector radius is too small')

nearestNode = self.findClosest(x, y, nodeSet)

attachmentPoint1 = myM.parts['Beam'].AttachmentPoints(
    name='Attachment Points' + str(studInstance.name) + str(ConstraintNum),
    points=(nearestNode,),
    setName='Attachment Points' + str(studInstance.name) + str(ConstraintNum))

self.addToSet('Inertia' + str(studInstance.name), studInstance.vertices[j.index:j.index+1])

# a engineeringFeatures.PointMassInertia(name='Inertia-' + str(studInstance.name) + '-' + str(ConstraintNum),
# region=a.sets['Inertia-' + str(studInstance.name)], mass=self.springStudBaseMass, alpha=0.0, composite=0.0, i11=self.springStudRotation, i22=self.springStudRotation, i33=self.springStudRotation)

for v in a.vertices.getByBoundingCylinder(center1=(x, y, 0.0), center2=(x, y, 1.0),
radius=self.meshScale*3.0):
    if v.featureName == 'DamperPoint-' + str(studInstance.name) + str(ConstraintNum):
        attachmentPoint1 = a.AttachmentPoints(
            name='DamperPoint-' + str(studInstance.name) + str(ConstraintNum),
            points=((x-math.cos(angle2)*0.10, y-math.sin(angle2)*0.10, 0.0),),
            setName='DamperPoint-' + str(studInstance.name) + str(ConstraintNum))

def createShearConnectionWires(studEnds, studPoints, concInstance, plateInstance, studInstance):
    a = myM.rootAssembly
    baseLocs = []
    if studEnds == 1:
        baseLocs.append((studPoints[1][0], studPoints[1][1]))
    elif studEnds == 2:
        baseLocs.append((studPoints[1][0], studPoints[1][1]))
        baseLocs.append((studPoints[2][0], studPoints[2][1]))

    end = 0
    for baseLoc in baseLocs:
        end = end + 1
        x = baseLoc[0]
        y = baseLoc[1]
        dx = studPoints[1][0] - studPoints[0][0]
        dy = studPoints[1][1] - studPoints[0][1]

        import math
        angled = self.rotationAngle(dy, dx)
        angled = angled+90.0
        if end == 2:
            angled = angled + 180.0

        angle = math.radians(angled)
        angle2 = math.radians(angled+90.0)

        vertBeam = self.findClosest(x, y, studInstance.vertices.getByBoundingCylinder(center1=(x, y, 0.0),
center2=(x, y, 1.0), radius=self.meshScale*3.0))
```

391
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
anchor = v

region1 = a.Set(vertices=a.vertices[anchor.index:anchor.index+1], name=’DamperInertia’ + str(end)+’-1’)
a.engineeringFeatures.PointMassInertia(name=’DamperInertia’ + str(end)+’-1’, region=region1, mass=1.0, alpha=0.0, composite=0.0, i11=self.springStudRotation, i22=self.springStudRotation, i33=self.springStudRotation)

# Setting rotation restraint causes crashes
myM.DisplacementBC(amplitude=UNSET, createStepName=’Initial’, distributionType=UNIFORM, fieldName=’ ’, localCsys=New, name=’DamperPoint’ + str(end)+’-1’, region=region1, u1=SET, u2=SET, ur3=UNSET)

if self.version == ’6.14-3’:
    newWire = a.WirePolyLine(points=(anchor, vertBeam), mergeType=IMPRINT, meshable=OFF)
else:
    newWire = a.WirePolyLine(points=(anchor, vertBeam), mergeWire=OFF, meshable=OFF)
a.features.changeKey(fromName=newWire.name, toName=’DamperWire’ + str(end)+’-1’)
for wire in a.edges:
    if wire.featureName == newWire.name:
        self.addToSet(’DamperWires’, wire=wire)
break

attachmentPoint2 = a.AttachmentPoints(name=’DamperPoint’ + str(end)+’-2’, points=((x-math.cos(angle)*0.10, y-math.sin(angle)*0.10, 0.0),), setName=’DamperPoint’ + str(end)+’-2’)
for v in a.vertices:
    if v.featureName == ’DamperPoint’ + str(end)+’-2’:
        anchor2 = v

region2 = a.Set(vertices=a.vertices[anchor.index:anchor.index+1], name=’DamperInertia’ + str(end)+’-2’)
a.engineeringFeatures.PointMassInertia(name=’DamperInertia’ + str(end)+’-2’, region=region2, mass=1.0, alpha=0.0, composite=0.0, i11=self.springStudRotation, i22=self.springStudRotation, i33=self.springStudRotation)

myM.DisplacementBC(amplitude=UNSET, createStepName=’Initial’, distributionType=UNIFORM, fieldName=’ ’, localCsys=New, name=’DamperPoint’ + str(end)+’-2’, region=region2, u1=SET, u2=SET, ur3=UNSET)

if self.version == ’6.14-3’:
    newWire2 = a.WirePolyLine(points=(anchor2, vertBeam), mergeType=IMPRINT, meshable=OFF)
else:
    newWire2 = a.WirePolyLine(points=(anchor2, vertBeam), mergeWire=OFF, meshable=OFF)
a.features.changeKey(fromName=newWire2.name, toName=’DamperWire’ + str(end)+’-2’)
for wire in a.edges:
    if wire.featureName == newWire2.name:
        self.addToSet(’DamperWires’, wire=wire)
break

for j in studInstance.vertices:
    x = j.pointOn[0][0]
y = j.pointOn[0][1]

searchNodes = concInstance.nodes.getByBoundingCylinder(center1=(x, y, 0.0),center2 = (x, y, 1.0),radius =self.meshScale/10.0)
```

392
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
if len(searchNodes) < 2:
    searchNodes = concInstance.nodes.getByBoundingCylinder(center1=(x, y, 0.0), center2=(x, y, 1.0),
    radius=self.meshScale*3.0)

attachmentPoint = self.findClosest(x, y, searchNodes)

if 1 == 2:
    xNode2 = attachmentPoint.pointOn[0][0]
    yNode2 = attachmentPoint.pointOn[0][1]
else:
    xNode2 = attachmentPoint.coordinates[0]
    yNode2 = attachmentPoint.coordinates[1]

delta = (xNode2-x)**2+(yNode2-y)**2)**0.5
if delta >= 0.000001:
    # print j
    # print attachmentPoint
    print str(x) + ': ' + str(xNode2) + ': ' + str(y) + ': ' + str(yNode2) + ': ' + str(delta)
    #a=bcd
else:
    # self.WirePositions.append((attachmentPoint, j))
    if self.version == '6.14-3':
        newWire1 = a.WirePolyLine(points=((attachmentPoint, j), ),
        mergeType=IMPRINT,
        meshable=OFF)
    else:
        newWire1 = a.WirePolyLine(points=((attachmentPoint, j), ),
        mergeWire=OFF,
        meshable=OFF)
    a.features.changeKey(fromName=newWire1.name, toName='LinkWire--' + str(studInstance.name) + '---' +
    str(j.index) + '-' + str(delta))

def swap1(self, array):
    temp = []
    for element in array:
        temp.append((element[1], element[0]))
    return temp

def addToSet(self, original, toadd, object='a', typeTest='ND'):
    # print original
    # print toadd
    if object == 'a':
        a = self.mym.RootAssembly
    else:
        a = object
    if typeTest=='ND':
        typeTest = str(type(toadd[0]).__name__)
    # print typeTest
    exCode = 'a.Set(name=original, ' + str(toadd + '))
    exCode = 'a.Set(name="' + original + '" + original + "", ' +

# Edges
try:
    oldEdges = a.sets[original].edges
    oldEdgesLen = len(oldEdges)
except KeyError:
    oldEdgesLen = 0
if oldEdgesLen == 0:
    if typeTest == 'Edge':
        exCode = exCode + 'edges=toadd, '  
    else:
        exCode = exCode + 'edges=oldEdges+toadd, '  
else:
    exCode = exCode + 'edges=oldEdges, '  
```

393
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
# ##############
# Faces
try:
    oldFaces = a.sets[original].faces
    oldFacesLen = len(oldFaces)
except KeyError:
    oldFacesLen = 0
if oldFacesLen == 0:
    if typeTest == 'Face':
        exCode = exCode + ' faces=toadd,
    else:
        if typeTest == 'Face':
            exCode = exCode + ' faces=oldFaces+toadd,
        else:
            exCode = exCode + ' faces=oldFaces,

# ##############
# Vertices
try:
    oldVertices = a.sets[original].vertices
    oldVerticesLen = len(oldVertices)
except KeyError:
    oldVerticesLen = 0
if oldVerticesLen == 0:
    if typeTest == 'Vertex':
        exCode = exCode + ' vertices=toadd,
    else:
        if typeTest == 'Vertex':
            exCode = exCode + ' vertices=oldVertices+toadd,
        else:
            exCode = exCode + ' vertices=oldVertices,

# ##############
# Cells
try:
    oldCells = a.sets[original].cells
    oldCellsLen = len(oldCells)
except KeyError:
    oldCellsLen = 0
if oldCellsLen == 0:
    if typeTest == 'Cell':
        exCode = exCode + ' cells=toadd,
    else:
        if typeTest == 'Vertex':
            exCode = exCode + ' cells=oldCells+toadd,
        else:
            exCode = exCode + ' cells=oldCells,

# Can surfaces be part of sets?
if 1 == 2:
    # ##############
    # Surfaces
    try:
        oldSurfaces = a.sets[original].surfaces
        oldSurfacesLen = len(oldSurfaces)
    except KeyError:
        oldSurfacesLen = 0
if oldSurfacesLen == 0:
    if typeTest == 'Surface':
        exCode = exCode + ' surfaces=toadd,
    else:
        if typeTest == 'Surface':
            exCode = exCode + ' surfaces=oldSurfaces+toadd,
        else:
            exCode = exCode + ' surfaces=oldSurfaces,
```

394
Can’t combine mesh and geometry sets
Set will include nodes when edge etc. selected, so only allow when making a node specific set

```python
if typeTest == 'MeshNode':
    # Mesh Nodes
    oldNodesLen = 0
    try:
        oldNodes = a.sets[original].nodes
        oldNodesLen = len(oldNodes)
    except:
        oldNodesLen = 0
    if oldNodesLen == 0:
        if typeTest == 'MeshNode':
            exCode = exCode + 'nodes=toadd,'  
        else:
            if typeTest == 'MeshNode':
                exCode = exCode + 'nodes=oldNodes+toadd,'  
            else:
                exCode = exCode + 'nodes=oldNodes,'  
    exCode = exCode + ')

# print exCode
exec exCode
```

```python
def findClosest(self, xTarget, yTarget, searchSet):
    deltaMin = 10000
    for i in searchSet:
        typeTest = str(type(i).__name__)
        if typeTest == 'MeshNode':
            xNode = i.coordinates[0]
            yNode = i.coordinates[1]
        else:
            xNode = i.isOn[0][0]
            yNode = i.isOn[0][1]
        delta = ((xTarget-xNode)**2+(yTarget-yNode)**2)**0.5
        if delta <= deltaMin:
            deltaMin = delta
            nearestPoint = i
        if deltaMin == 0:
            break
    return nearestPoint
```

```python
def storVar(self, VarName, Value):
    session.XYData(name='Variable#' + VarName, data=((0,0)), sourceDescription=str(Value))
```
import ntpath
#CurrentDir = os.path.realpath('..')
#CurrentDir = r'\UWS30089\Users\pf00062.SURREY\OneDrive\Surrey'
copyPath = self.CurrentDir + r'//ABAQUS//Results//' + self.Job_Name + r'//Variables//'

if not os.path.exists(copyPath):
    os.makedirs(copyPath)

FilePath = copyPath + VarName + '.txt'
dispFile = open(FilePath, 'w')
if type(Value) is list:
    for i in Value:
        dispFile.write(' %10.4E, %10.4E
' % (i[0], i[1]))
elif type(Value) is tuple:
    for i in Value:
        dispFile.write(' %10.4E, %10.4E
' % (i[0], i[1]))
else:
    dispFile.write(str(Value))
dispFile.close()

def setFromCriteria(self, objectList, setName, object='a',
x='none', y='none', z='none',
xmin='none', xmax='none',
ymin='none', ymax='none'):
    a = self.mM.rootAssembly
    if object == 'a':
        a = self.mM.rootAssembly
    else:
        a = object
    for object1 in objectList:
        typeTest = str(type(object1).__name__)
        xs = []
        ys = []
        zs = []
        if typeTest == 'Vertex':
            xs.append(object1.pointOn[0][0])
            ys.append(object1.pointOn[0][1])
            zs.append(object1.pointOn[0][2])
        elif typeTest == 'Surface':
            for face in object1.faces:
                verts = face.getVertices()
                for vert in verts:
                    xs.append(vert.pointOn[0][0])
                    ys.append(vert.pointOn[0][1])
                    zs.append(vert.pointOn[0][2])
        elif typeTest == 'MeshNode':
            xs.append(object1.coordinates[0])
            ys.append(object1.coordinates[1])
            zs.append(object1.coordinates[2])
        else:  # Covers Edges, Cells, Faces
            if typeTest == 'Edge':
                pass
            elif typeTest == 'Cell':
                pass
            elif typeTest == 'Face':
                pass
            else:
                raise ValueError(typeTest + ' is not covered by setFromCriteria')
        verts = object1.getVertices()
        for vert in verts:
try:
    xs.append(a.vertices[vert].pointOn[0][0])
    ys.append(a.vertices[vert].pointOn[0][1])
    zs.append(a.vertices[vert].pointOn[0][2])
except:
    xs.append(a.instances[object1.instanceName].vertices[vert].pointOn[0][0])
    ys.append(a.instances[object1.instanceName].vertices[vert].pointOn[0][1])
    zs.append(a.instances[object1.instanceName].vertices[vert].pointOn[0][2])

xs=list(set(xs))
y=list(set(ys))
zs=list(set(zs))

d = True

d = False

if d == True:
    print str(xs) + ' : ' + str(x)
    print str(ys) + ' : ' + str(y)
    print str(zs) + ' : ' + str(z)
    print typeTest

a=b

# import time
# time.sleep(0.3)

tolerance = 0.00001

addObject = True

if x !='none':
    if len(xs) == 1 and xs[0] >= x-tolerance and xs[0] <= x+tolerance:
        pass
    else:
        addObject = False

if xmin != 'none':
    for xs1 in xs:
        if xmin > xmax:
            xminTemp = xmin
            xmin=xmax
            xmax = xminTemp
        if xs1 >= xmin-tolerance and xs1 <= xmax+tolerance:
            pass
        else:
            addObject = False

if ymin != 'none':
    for ys1 in ys:
        if ymin > ymax:
            yminTemp = ymin
            ymin=ymax
            ymax = yminTemp
        if ys1 >= ymin-tolerance and ys1 <= ymax+tolerance:
            pass
        else:
            addObject = False

if z != 'none':
    if len(zs) == 1 and zs[0] == z:
        pass
    else:
        addObject = False
APPENDIX A: PYTHON CODE FOR MODELLING AN SCS BEAM USING ABAQUS

```python
if addObject == True:
    try:
        b = a.instances[object1.instanceName]
    except:
        b=a

if typeTest == 'Edge':
    self.addToSet(original=setName, toadd=b.edges[object1.index:object1.index+1], object=a)
elif typeTest == 'Face':
    self.addToSet(original=setName, toadd=b.faces[object1.index:object1.index+1], object=a)
elif typeTest == 'Vertex':
    self.addToSet(original=setName, toadd=b.vertices[object1.index:object1.index+1], object=a)
elif typeTest == 'Cell':
    self.addToSet(original=setName, toadd=b.cells[object1.index:object1.index+1], object=a)
elif typeTest == 'MeshNode':
    self.addToSet(original=setName, toadd=b.nodes[object1.label-1:object1.label], object=a)
elif typeTest == 'Surface':
    pass
    # self.addToSet(original=setName, toadd=(object1,), object=a)

def rotationAngle(self, dy, dx):
    import math
    # Initial result
    if dx == 0.0:
        angle = 0.0
    else:
        angle = math.degrees(math.atan(dy/dx))

    # Quadrant
    if dx == 0.0:
        if dy == 0.0:
            a=b
        elif dy > 0.0:
            angle = 90.0
        elif dy < 0.0:
            angle = 270.0
    elif dx > 0.0:
        if dy == 0.0:
            angle = 0.0
        elif dy > 0.0:
            angle = angle
        elif dy < 0.0:
            angle = 360.0-angle
    elif dx < 0.0:
        if dy == 0.0:
            angle = 180.0
        elif dy > 0.0:
            angle = 180.0-angle
        elif dy < 0.0:
            angle = 180.0+angle

    # angle=angle+180.0
    return angle

class sparePoint():
    def __init__(self):
        self.xValue = -100.0
        self.yValue = -100.0
        self.zValue = -100.0

    sp = sparePoint()

global Model
Model=AssemblyClass()

try:
    if runModel == 2:
        pass
    else:
        pass
```

except:
    print 'runModel not found'
runModel = 2
try:
    if runModel == 1:
        runModeBoolean = False
    else:
        runModeBoolean = True
    print 'Build Model and Run ' + str(runModeBoolean)
    print Model
    Model.BuildModel(runModeBoolean)
except:
    import traceback
    print '##################'
    for frame in traceback.extract_tb(sys.exc_info()[2]):
        fname,lineno,ln,text = frame
        print 'Error in %s on line %d' % (fname,lineno)
        print '##################'
        print(traceback.format_exc())
        print '##################'
        #print(sys.exc_info()[0])
APPENDIX B

Test database in Python code format

This appendix presents a listing of all the parameters used to describe the tests presented in the test database (Chapter 3). The data is presented in Python format, as required by ABAQUS.

```python
if self.Case_Name.find('BeamTest') > -1:
    self.beamCaseNo = self.ModelNo
else:
    self.beamCaseNo = 0

if self.Case_Name == 'Oduyemi_A1' or self.beamCaseNo == 6:
    self.L = 1500.00/1000.0
    self.beamh = 150.00/1000.0
    self.beamb = 150.00/1000.0
    self.bottomt = 3.00/1000.0
    self.topt = 2.00/1000.0
    self.plateFyBottom = 268.00
    self.plateFyTop = 242.00
    self.plateFuBottom = 318.00
    self.plateFuTop = 292.00
    self.concretefC = 49.70
    self.studb1h = 35.00/1000.0
    self.studb1Dia = 6.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 11.29*1000.0
    self.studb1fy = 449.00
    self.studb1fu = 499.00
    self.studb2h = 0.00/1000.0
    self.studb2Dia = 0.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 0.00*1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50
    self.studt1h = 35.00/1000.0
    self.studt1Dia = 6.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 11.29*1000.0
    self.studt1fy = 449.00
    self.studt1fu = 499.00
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00*1000.0
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studt2fy = 1.00
self.studt2fu = 1.50
self.CyclicLoad = 50.00/1000.0
self.appliedDeflection = 0.05
self.support1 = 25.00/1000.0
self.support1a = 1475.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 575.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.05, 'b1', 2,(2)), (0.15, 'b1', 2,(2)), (0.25, 'b1', 2,(2)), (0.35, 'b1', 2,(2)),
                                       (0.45, 'b1', 2,(2)), (0.55, 'b1', 2,(2)), (0.65, 'b1', 2,(2)), (0.75, 'b1', 2,(2)), (0.85, 'b1', 2,(2)), (0.95, 'b1', 2,(2)),
                                       (1.05, 'b1', 2,(2)), (1.15, 'b1', 2,(2)), (1.25, 'b1', 2,(2)), (1.35, 'b1', 2,(2)), (1.45, 'b1', 2,(2)),))
self.studPositionDescriptions.extend(((0.1, 't1', 2,(2)), (0.2, 't1', 2,(2)), (0.3, 't1', 2,(2)), (0.4, 't1', 2,(2)), (0.5, 't1', 2,(2)), (0.6, 't1', 2,(2)), (0.7, 't1', 2,(2)), (0.8, 't1', 2,(2)), (0.9, 't1', 2,(2)), (1.0, 't1', 2,(2)), (1.1, 't1', 2,(2)),
                                       (1.2, 't1', 2,(2)), (1.3, 't1', 2,(2)), (1.4, 't1', 2,(2)),))

print(self.Case_Name)
if self.Case_Name == 'Oduyemi_A2' or self.beamCaseNo==7:
    self.L = 1500.00/1000.0
    self.beamb = 150.00/1000.0
    self.bottomt = 3.00/1000.0
    self.topt = 2.00/1000.0
    self.plateFyBottom = 268.00
    self.plateFyTop = 242.00
    self.plateFuBottom = 318.00
    self.plateFuTop = 292.00
    self.concretefc = 51.20
    self.studb1h = 0.08/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00*1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50
    self.studb2h = 135.00/1000.0
    self.studb2Dia = 6.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 11.29*1000.0
    self.studb2fy = 449.00
    self.studb2fu = 499.00
    self.studt1h = 35.00/1000.0
    self.studt1Dia = 6.00/1000.0
```

401
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud1FullHeight = False
self.stud1Resistance = 11.29 * 1000.0
self.stud1fy = 449.00
self.stud1fu = 499.00

self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 25.00/1000.0
self.supportLa = 1475.00/1000.0
self.supportlb = -100.00/1000.0
self.supportlc = -100.00/1000.0

self.load1 = 575.00/1000.0
self.loada = 925.00/1000.0
self.loadb = -100.00/1000.0
self.loadc = -100.00/1000.0
self.loadd = -100.00/1000.0
self.loade = -100.00/1000.0
self.loadf = -100.00/1000.0
self.loadg = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.05,'b2',2,(2)),(0.15,'b2',2,(2)),(0.25,'b2',2,(2)),(0.35,'b2',2,(2))
,(0.45,'b2',2,(2)),(0.55,'b2',2,(2)),(0.65,'b2',2,(2)),(0.75,'b2',2,(2)),
,(0.75,'b2',2,(2)),(0.85,'b2',2,(2)),(0.95,'b2',2,(2)),
,(1.05,'b2',2,(2)),(1.15,'b2',2,(2)),(1.25,'b2',2,(2)),(1.35,'b2',2,(2)),
,(1.45,'b2',2,(2)),)
self.studPositionDescriptions.extend(((0.1,'t1',2,(2)),(0.2,'t1',2,(2)),(0.3,'t1',2,(2)),(0.4,'t1',2,(2)),
,(0.5,'t1',2,(2)),(0.6,'t1',2,(2)),(0.7,'t1',2,(2)),(0.8,'t1',2,(2)),
,(0.9,'t1',2,(2)),(1,'t1',2,(2)),(1.1,'t1',2,(2)),
,(1.2,'t1',2,(2)),(1.3,'t1',2,(2)),(1.4,'t1',2,(2)),))

print self.Case_Name
if self.Case_Name == 'Olayemi_B1' or self.beamCaseNo==8:

self.L = 1500.00/1000.0
self.beamb = 150.00/1000.0
self.beamb = 150.00/1000.0
self.bottomt = 2.08/1000.0
self.topt = 3.00/1000.0

self.plateFyBottom = 242.00
self.plateFyTop = 268.00
self.plateFuBottom = 292.00
self.plateFuTop = 318.00
self.concretefc = 51.00

self.studh1 = 0.00/1000.0
self.studDia1 = 0.00/1000.0
self.studFullHeight1 = False
self.stud1Resistance = 0.00+1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50
```

402
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2h = 135.00/1000.0
self.studb2Dia = 6.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 11.29*1000.0
self.studb2fy = 449.00
self.studb2fu = 499.00

self.stud1h = 35.00/1000.0
self.stud1Dia = 6.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 11.29*1000.0
self.stud1fy = 449.00
self.stud1fu = 499.00

self.studt1h = 35.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 11.29*1000.0
self.studt1fy = 449.00
self.studt1fu = 499.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 25.00/1000.0
self.support1a = 1475.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 575.00/1000.0
self.load1a = 925.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadDefPoints = []
self.loadDefPoints.extend([(750/1000.0, 0.0),])
self.studPositionDescriptions.extend([(0.05, 'b2', 2, (2)), (0.15, 'b2', 2, (2)), (0.25, 'b2', 2, (2)), (0.35, 'b2', 2, (2)), (0.45, 'b2', 2, (2)), (0.55, 'b2', 2, (2)), (0.65, 'b2', 2, (2)), (0.75, 'b2', 2, (2)), (0.85, 'b2', 2, (2)), (0.95, 'b2', 2, (2)), (1.05, 'b2', 2, (2)), (1.15, 'b2', 2, (2)), (1.25, 'b2', 2, (2)), (1.35, 'b2', 2, (2)), (1.45, 'b2', 2, (2)),])
self.studPositionDescriptions.extend([(0.0, 't1', 2, (2)), (0.1, 't1', 2, (2)), (0.2, 't1', 2, (2)), (0.3, 't1', 2, (2)), (0.4, 't1', 2, (2)), (0.5, 't1', 2, (2)), (0.6, 't1', 2, (2)), (0.7, 't1', 2, (2)), (0.8, 't1', 2, (2)), (0.9, 't1', 2, (2)), (1.0, 't1', 2, (2)), (1.1, 't1', 2, (2)), (1.2, 't1', 2, (2)), (1.3, 't1', 2, (2)), (1.4, 't1', 2, (2)),])
print self.Case_Name
if self.Case_Name == 'Oduyemi_B2' or self.beamCaseNo==9:

self.L = 1500.00/1000.0
self.beamb = 150.00/1000.0
self.bottomt = 3.00/1000.0
self.topt = 3.00/1000.0

self.plateFyBottom = 268.00
self.plateFyTop = 268.00
self.plateFuBottom = 318.00
self.plateFuTop = 318.00
```

403
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.concrete fc = 52.20
self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50
self.studb2h = 135.00/1000.0
self.studb2Dia = 6.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 11.29+1000.0
self.studb2fy = 449.00
self.studb2fu = 499.00
self.studt1h = 35.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 11.29+1000.0
self.studt1fy = 449.00
self.studt1fu = 499.00
self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00
self.studt2fy = 1.00
self.studt2fu = 1.50
self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05
self.support1 = 25.00/1000.0
self.support1a = 1475.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 575.00/1000.0
self.load1a = 925.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.05,'b2',2,(2)),(0.15,'b2',2,(2)),(0.25,'b2',2,(2)),(0.35,'b2',2,(2)),(0.45,'b2',2,(2)),(0.55,'b2',2,(2)),(0.65,'b2',2,(2)),(0.75,'b2',2,(2)),(0.85,'b2',2,(2)),(0.95,'b2',2,(2))
(0.05,'b2',2,(2)),(0.15,'b2',2,(2)),(0.25,'b2',2,(2)),(0.35,'b2',2,(2)),(0.45,'b2',2,(2)),(0.55,'b2',2,(2)),(0.65,'b2',2,(2)),(0.75,'b2',2,(2)),(0.85,'b2',2,(2)),(0.95,'b2',2,(2))
(0.05,'t1',2,(2)),(0.15,'t1',2,(2)),(0.25,'t1',2,(2)),(0.35,'t1',2,(2)),(0.45,'t1',2,(2)),(0.55,'t1',2,(2)),(0.65,'t1',2,(2)),(0.75,'t1',2,(2)),(0.85,'t1',2,(2)),(0.95,'t1',2,(2))
(1.05,'t1',2,(2)),(1.15,'t1',2,(2)),(1.25,'t1',2,(2)),(1.35,'t1',2,(2)),(1.45,'t1',2,(2)))
self.studPositionDescriptions.extend(((0.1,'t1',2,(2)),(0.2,'t1',2,(2)),(0.3,'t1',2,(2)),(0.4,'t1',2,(2)),(0.5,'t1',2,(2)),(0.6,'t1',2,(2)),(0.7,'t1',2,(2)),(0.8,'t1',2,(2)),(0.9,'t1',2,(2)),(1.0,'t1',2,(2)),(1.1,'t1',2,(2))
(1.2,'t1',2,(2)),(1.3,'t1',2,(2)),(1.4,'t1',2,(2)),(1.5,'t1',2,(2)))
```

print self.Case_Name
if self.Case_Name == 'Oduyemi_B3' or self.beamCaseNo == 10:
    self.L = 1500.00/1000.0
```
self.bottom = 150.00
self.beam = 150.00
self.bottom = 4.00
self.top = 4.00

self.plateFyBottom = 252.00
self.plateFyTop = 252.00
self.plateFuBottom = 302.00
self.plateFuTop = 302.00
self.concrete fc = 49.00

self.stud1h = 0.00
self.stud1Dia = 0.00
self.stud1FullHeight = False
self.stud1Resistance = 0.00+1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50

self.stud2h = 135.00
self.stud2Dia = 6.00
self.stud2FullHeight = False
self.stud2Resistance = 11.29+1000.0
self.stud2fy = 449.00
self.stud2fu = 499.00

self.stud3h = 35.00
self.stud3Dia = 6.00
self.stud3FullHeight = False
self.stud3Resistance = 11.29+1000.0
self.stud3fy = 449.00
self.stud3fu = 499.00

self.stud4h = 0.00
self.stud4Dia = 6.00
self.stud4FullHeight = False
self.stud4Resistance = 11.29+1000.0
self.stud4fy = 449.00
self.stud4fu = 499.00

self.load1 = 575.00
self.load1a = 925.00
self.load1b = -100.00
self.load1c = -100.00
self.load1d = -100.00
self.load1e = -100.00
self.load1f = -100.00
self.load1g = -100.00

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000.0, 0.0),))
self.studentPositionDescriptions.extend(((0.05, 'b2', 3, (3)), (0.15, 'b2', 3, (3)), (0.25, 'b2', 3, (3)), (0.35, 'b2', 3, (3)), (0.45, 'b2', 3, (3)), (0.55, 'b2', 3, (3)), (0.65, 'b2', 3, (3)), (0.75, 'b2', 3, (3)), (0.85, 'b2', 3, (3)), (0.95, 'b2', 3, (3))
(1.05, 'b2', 3, (3)), (1.15, 'b2', 3, (3)), (1.25, 'b2', 3, (3)), (1.35, 'b2', 3, (3)), (1.45, 'b2', 3, (3))))
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studPositionDescriptions.extend(((0.1, 't1', 2, (2)), (0.2, 't1', 2, (2)), (0.3, 't1', 2, (2)), (0.4, 't1', 2, (2)), (0.5, 't1', 2, (2)), (0.6, 't1', 2, (2)), (0.7, 't1', 2, (2)), (0.8, 't1', 2, (2)), (0.9, 't1', 2, (2)), (1.0, 't1', 2, (2)), (1.1, 't1', 2, (2)), (1.2, 't1', 2, (2)), (1.3, 't1', 2, (2)), (1.4, 't1', 2, (2)),))

print self.Case_Name
if self.Case_Name == 'Oduyemi_B4' or self.beamCaseNo==11:

    self.L = 1500.00/1000.0
    self.beamb = 150.00/1000.0
    self.beambottom = 6.00/1000.0
    self.top = 4.00/1000.0

    self.plateFyBottom = 288.00
    self.plateFyTop = 252.00
    self.plateFuBottom = 338.00
    self.plateFuTop = 302.00
    self.concretefc = 51.60

    self.studb1h = 0.00/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00*1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50

    self.studb2h = 135.00/1000.0
    self.studb2Dia = 6.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 11.29*1000.0
    self.studb2fy = 449.00
    self.studb2fu = 499.00

    self.studt1h = 35.00/1000.0
    self.studt1Dia = 6.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 11.29*1000.0
    self.studt1fy = 449.00
    self.studt1fu = 499.00

    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00*1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50

    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05

    self.support1 = 25.00/1000.0
    self.support1a = 1475.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0

    self.load1 = 575.00/1000.0
    self.load1a = 925.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = 1.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
    self.loadFactor1d = -100.00
    self.loadFactor1e = -100.00
```

406
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000.0,0.0),))
self.studyPositionDescriptions.extend(((0.05,'b2',2,(2)),(0.1,'b2',2,(2)),(0.15,'b2',2,(2)),(0.2,'b2',2,(2)),(0.3,'b2',2,(2)),(0.35,'b2',2,(2)),(0.4,'b2',2,(2)),(0.45,'b2',2,(2)),(0.5,'b2',2,(2)),(0.55,'b2',2,(2)),(0.6,'b2',2,(2)),(0.65,'b2',2,(2)),(0.7,'b2',2,(2)),(0.75,'b2',2,(2)),(0.8,'b2',2,(2)),(0.85,'b2',2,(2)),(0.9,'b2',2,(2)),(0.95,'b2',2,(2)),(1,'b2',2,(2)),(1.05,'b2',2,(2)),(1.1,'b2',2,(2)),(1.15,'b2',2,(2)),(1.2,'b2',2,(2)),(1.25,'b2',2,(2)),(1.3,'b2',2,(2)),(1.35,'b2',2,(2)),(1.4,'b2',2,(2)),(1.45,'b2',2,(2)),(1.5,'b2',2,(2)),))
self.studyPositionDescriptions.extend(((0.1,'t1',2,(2)),(0.2,'t1',2,(2)),(0.3,'t1',2,(2)),(0.4,'t1',2,(2)),(0.5,'t1',2,(2)),(0.6,'t1',2,(2)),(0.7,'t1',2,(2)),(0.8,'t1',2,(2)),(0.9,'t1',2,(2)),(1,'t1',2,(2)),(1.1,'t1',2,(2)),(1.2,'t1',2,(2)),(1.3,'t1',2,(2)),(1.4,'t1',2,(2)),(1.5,'t1',2,(2)),(1.6,'t1',2,(2)),(1.7,'t1',2,(2)),(1.8,'t1',2,(2)),(1.9,'t1',2,(2)),(2,'t1',2,(2)),))

if self.Case_Name == 'Oduyemi_C1' or self.beamCaseNo==12:
    self.L = 1500.00/1000.0
    self.beamb = 150.00/1000.0
    self.beamb = 150.00/1000.0
    self.bottomt = 3.00/1000.0
    self.top = 2.00/1000.0
    self.plateFyBottom = 268.00
    self.plateFyTop = 242.00
    self.plateFuBottom = 318.00
    self.plateFuTop = 292.00
    self.concRefc = 51.60
    self.studb1h = 0.00/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00+1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50
    self.studb2h = 135.00/1000.0
    self.studb2Dia = 6.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 11.29+1000.0
    self.studb2fy = 449.00
    self.studb2fu = 499.00
    self.studth = 35.00/1000.0
    self.stud1Dia = 6.00/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 11.29+1000.0
    self.stud1fy = 449.00
    self.stud1fu = 499.00
    self.stud2h = 0.00/1000.0
    self.stud2Dia = 0.00/1000.0
    self.stud2FullHeight = False
    self.stud2Resistance = 0.00+1000.0
    self.stud2fy = 1.00
    self.stud2fu = 1.50

if self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 25.00/1000.0
self.support1a = 1475.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 575.00/1000.0
self.load1a = 925.00/1000.0
self.load1b = -100.00/1000.0
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.05,'b2',2,(2)),(0.15,'b2',2,(2)),(0.25,'b2',2,(2)),(0.35,'b2',2,(2)),(0.45,'b2',2,(2)),(0.55,'b2',2,(2)),(0.65,'b2',2,(2)),(0.75,'b2',2,(2)),(0.85,'b2',2,(2)),(0.95,'b2',2,(2)),(1.05,'b2',2,(2)),(1.15,'b2',2,(2)),(1.25,'b2',2,(2)),(1.35,'b2',2,(2)),(1.45,'b2',2,(2)),))
self.studPositionDescriptions.extend(((0.15,'t1',2,(2)),(0.35,'t1',2,(2)),(0.55,'t1',2,(2)),(0.75,'t1',2,(2)),(0.95,'t1',2,(2)),(1.15,'t1',2,(2)),(1.35,'t1',2,(2)),))

print self.Case_Name
if self.Case_Name == 'Oduyemi_C2' or self.beamCaseNo==13:
    self.L = 1500.00/1000.0
    self.beamb = 150.00/1000.0
    self.bottomt = 3.00/1000.0
    self.topf = 2.00/1000.0
    self.plateFyBottom = 268.00
    self.plateFyTop = 242.00
    self.plateFuBottom = 318.00
    self.plateFuTop = 292.00
    self.concretefc = 53.20
    self.studh1h = 0.08/1000.0
    self.studh1Dia = 0.00/1000.0
    self.studh1FullHeight = False
    self.studh1Resistance = 0.00+1000.0
    self.studh1fy = 1.00
    self.studh1fu = 1.50
    self.studh2h = 135.00/1000.0
    self.studh2Dia = 6.00/1000.0
    self.studh2FullHeight = False
    self.studh2Resistance = 11.29+1000.0
    self.studh2fy = 449.00
    self.studh2fu = 499.00
    self.studt1h = 35.00/1000.0
    self.studt1Dia = 6.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 11.29+1000.0
    self.studt1fy = 449.00
    self.studt1fu = 499.00
    self.studt2h = 0.08/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00+1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05
```

408
Appendix B: Test Database in Python Code Format

```python
self.support1 = 25.00/1000.0
self.support1a = 1475.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 575.00/1000.0
self.load1a = 925.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000.0, 0.0),))
self.studDescriptions.extend(((0.05, 'b2', 2,(2)), (0.15, 'b2', 2,(2)), (0.25, 'b2', 2,(2)), (0.35, 'b2', 2,(2)), (0.45, 'b2', 2,(2)), (0.55, 'b2', 2,(2)), (0.65, 'b2', 2,(2)), (0.75, 'b2', 2,(2)), (0.85, 'b2', 2,(2)), (0.95, 'b2', 2,(2)), (1.05, 'b2', 2,(2)), (1.15, 'b2', 2,(2)), (1.25, 'b2', 2,(2)), (1.35, 'b2', 2,(2)), (1.45, 'b2', 2,(2)),))
self.studDescriptions.extend(((0.15, 't1', 2,(2)), (0.55, 't1', 2,(2)), (0.95, 't1', 2,(2)), (1.35, 't1', 2,(2)),))

print self.Case_Name
if self.Case_Name == 'Oduyemi_D1' or self.beamCaseNo==14:

self.L = 1500.00/1000.0
self.beamh = 150.00/1000.0
self.beamb = 150.00/1000.0
self.bottomt = 3.00/1000.0
self.topt = 3.00/1000.0
self.plateFyBottom = 268.00
self.plateFyTop = 268.00
self.plateFuBottom = 318.00
self.plateFuTop = 318.00
self.concreteFc = 45.00
self.studb1h = 35.00/1000.0
self.studb1Dia = 6.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 11.29*1000.0
self.studb1fy = 449.00
self.studb1fu = 499.00
self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50
self.studt1h = 35.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 11.29*1000.0
self.studt1fy = 449.00
self.studt1fu = 499.00
self.studt2h = 135.00/1000.0
self.studt2Dia = 6.00/1000.0
```

409
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2FullHeight = False
self.stud2Resistance = 11.29*1000.0
self.stud2fy = 449.00
self.stud2fu = 499.00
self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 25.00/1000.0
self.support1a = 1475.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 575.00/1000.0
self.load1a = 925.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.supportFactor1 = 1.00
self.supportFactor1a = 1.00
self.supportFactor1b = -100.00
self.supportFactor1c = -100.00
self.supportFactor1d = -100.00
self.supportFactor1e = -100.00
self.supportFactor1f = -100.00
self.supportFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000.0, 0.0),))
self.loadPositionDescriptions.extend(((0.085, 'b1', 2, (2)), (0.218, 'b1', 2, (2)), (0.351, 'b1', 2, (2)), (0.484, 'b1', 2, (2)), (0.617, 'b1', 2, (2)), (0.75, 'b1', 2, (2)), (0.883, 'b1', 2, (2)), (1.016, 'b1', 2, (2)), (1.149, 'b1', 2, (2)), (1.282, 'b1', 2, (2)), (1.415, 'b1', 2, (2)),))
self.loadPositionDescriptions.extend(((0.1, 't1', 2, (2)), (0.2, 't1', 2, (2)), (0.3, 't1', 2, (2)), (0.4, 't1', 2, (2)), (0.5, 't1', 2, (2)), (0.6, 't1', 2, (2)), (0.7, 't1', 2, (2)), (0.8, 't1', 2, (2)), (0.9, 't1', 2, (2)), (1.0, 't1', 2, (2)), (1.1, 't1', 2, (2)), (1.2, 't1', 2, (2)), (1.3, 't1', 2, (2)), (1.4, 't1', 2, (2)),))
self.loadPositionDescriptions.extend(((0.05, 't2', 2, (2)), (0.15, 't2', 2, (2)), (0.25, 't2', 2, (2)), (0.35, 't2', 2, (2)), (0.45, 't2', 2, (2)), (0.55, 't2', 2, (2)), (0.65, 't2', 2, (2)), (0.75, 't2', 2, (2)), (0.85, 't2', 2, (2)), (0.95, 't2', 2, (2)), (1.05, 't2', 2, (2)), (1.15, 't2', 2, (2)), (1.25, 't2', 2, (2)), (1.35, 't2', 2, (2)), (1.45, 't2', 2, (2)),))

print self.Case_Name
if self.Case_Name == 'Oduyemi_D2' or self.beamCaseNo==15:
    self.L = 1500.00/1000.0
    self.beamh = 150.00/1000.0
    self.beamb = 150.00/1000.0
    self.bottomt = 3.00/1000.0
    self.top = 3.00/1000.0

self.plateFyBottom = 268.00
self.plateFyTop = 268.00
self.plateFullBottom = 318.00
self.plateFullTop = 318.00
self.concretefc = 45.80
self.stud1h = 35.00/1000.0
self.stud1Dia = 6.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 11.29*1000.0
self.stud1fy = 449.00
self.stud1fu = 499.00
self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00*1000.0
```

410
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 35.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 11.29*1000.0
self.studt1fy = 449.00
self.studt1fu = 499.00

self.studt2h = 135.00/1000.0
self.studt2Dia = 6.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 11.29*1000.0
self.studt2fy = 449.00
self.studt2fu = 499.00

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 25.00/1000.0
self.support1a = 1475.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 575.00/1000.0
self.load1a = 925.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000.0, 0.0),))
self.loadPositionDescriptions.extend(((0.05, 'b1', 2,(2)), (0.25, 'b1', 2,(2)), (0.45, 'b1', 2,(2)), (0.65, 'b1', 2,(2)), (0.85, 'b1', 2,(2)), (1.05, 'b1', 2,(2)), (1.25, 'b1', 2,(2)), (1.45, 'b1', 2,(2)), ))
self.loadPositionDescriptions.extend(((0.0, 't1', 2,(2)), (0.05, 't1', 2,(2)), (0.1, 't1', 2,(2)), (0.15, 't1', 2,(2)), (0.2, 't1', 2,(2)), (0.25, 't1', 2,(2)), (0.3, 't1', 2,(2)), (0.35, 't1', 2,(2)), (0.4, 't1', 2,(2)), (0.45, 't1', 2,(2)), (0.5, 't1', 2,(2)), (0.55, 't1', 2,(2)), (0.6, 't1', 2,(2)), (0.65, 't1', 2,(2)), (0.7, 't1', 2,(2)), (0.75, 't1', 2,(2)), (0.8, 't1', 2,(2)), (0.85, 't1', 2,(2)), (0.9, 't1', 2,(2)), (0.95, 't1', 2,(2)), (1.0, 't1', 2,(2)), (1.05, 't1', 2,(2)), (1.1, 't1', 2,(2)), (1.15, 't1', 2,(2)), (1.2, 't1', 2,(2)), (1.25, 't1', 2,(2)), (1.3, 't1', 2,(2)), (1.35, 't1', 2,(2)), (1.4, 't1', 2,(2)), (1.45, 't1', 2,(2)), ))
self.loadPositionDescriptions.extend(((0.05, 't2', 2,(2)), (0.15, 't2', 2,(2)), (0.25, 't2', 2,(2)), (0.35, 't2', 2,(2)), (0.45, 't2', 2,(2)), (0.55, 't2', 2,(2)), (0.65, 't2', 2,(2)), (0.75, 't2', 2,(2)), (0.85, 't2', 2,(2)), (0.95, 't2', 2,(2)), (1.0, 't2', 2,(2)), (1.1, 't2', 2,(2)), (1.15, 't2', 2,(2)), (1.2, 't2', 2,(2)), (1.25, 't2', 2,(2)), (1.3, 't2', 2,(2)), (1.35, 't2', 2,(2)), (1.4, 't2', 2,(2)), (1.45, 't2', 2,(2)), ))

print self.Case_Name
if self.Case_Name == 'Oduyemi_D3' or self.beamCaseNo==16:

self.L = 1500.00/1000.0
self.beamb = 150.00/1000.0
self.beamb = 150.00/1000.0
self.bottomt = 3.00/1000.0
self.topdf = 3.00/1000.0

self.plateFyBottom = 268.00
self.plateFyTop = 268.00
self.plateFuBottom = 318.00
self.plateFuTop = 318.00
self.concretestc = 47.20
```

411
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb1h = 35.00/1000.0
self.studb1Dia = 6.00/1000.0
self.studb1Resistance = 11.29*1000.0
self.studb1fy = 449.00
self.studb1fu = 499.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 35.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1Resistance = 11.29*1000.0
self.studt1fy = 449.00
self.studt1fu = 499.00

self.studt2h = 135.00/1000.0
self.studt2Dia = 6.00/1000.0
self.studt2Resistance = 11.29*1000.0
self.studt2fy = 449.00
self.studt2fu = 499.00

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 25.00/1000.0
self.support1a = 1475.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 575.00/1000.0
self.load1a = 925.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000, 0.0),))

self.studPositionDescriptions.extend(((0.15, 'b1', 2, (2)), (0.55, 'b1', 2, (2)), (0.95, 'b1', 2, (2)), (1.35, 'b1', 2, (2)),))
self.studPositionDescriptions.extend(((0.1, 't1', 2, (2)), (0.2, 't1', 2, (2)), (0.3, 't1', 2, (2)), (0.4, 't1', 2, (2)), (0.5, 't1', 2, (2)), (0.6, 't1', 2, (2)), (0.7, 't1', 2, (2)), (0.8, 't1', 2, (2)), (0.9, 't1', 2, (2)), (1.0, 't1', 2, (2)), (1.1, 't1', 2, (2)),))
self.studPositionDescriptions.extend(((0.05, 't2', 2, (2)), (0.15, 't2', 2, (2)), (0.25, 't2', 2, (2)), (0.35, 't2', 2, (2)), (0.45, 't2', 2, (2)), (0.55, 't2', 2, (2)), (0.65, 't2', 2, (2)), (0.75, 't2', 2, (2)), (0.85, 't2', 2, (2)), (0.95, 't2', 2, (2)), (1.05, 't2', 2, (2)), (1.15, 't2', 2, (2)), (1.25, 't2', 2, (2)), (1.35, 't2', 2, (2)), (1.45, 't2', 2, (2)),))

print self.Case_Name
if self.Case_Name == 'Oduyemi_E2' or self.beamCaseNo==18:
    self.L = 1500.00/1000.0
```

412
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.beamh = 150.00/1000.0
self.beamb = 150.00/1000.0
self.bottomt = 6.00/1000.0
self.top = 4.00/1000.0

self.plateFyBottom = 288.00
self.plateFyTop = 252.00
self.plateFuBottom = 338.00
self.plateFuTop = 302.00
self.concretefc = 37.68

self.studh1h = 35.00/1000.0
self.studh1Dia = 6.00/1000.0
self.studh1FullHeight = False
self.studh1Resistance = 11.29*1000.0
self.studh1fy = 449.00
self.studh1fu = 499.00

self.studh2h = 135.00/1000.0
self.studh2Dia = 6.00/1000.0
self.studh2FullHeight = False
self.studh2Resistance = 11.29*1000.0
self.studh2fy = 449.00
self.studh2fu = 499.00

self.studt1h = 35.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 11.29*1000.0
self.studt1fy = 449.00
self.studt1fu = 499.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 25.00/1000.0
self.support1a = 1475.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 575.00/1000.0
self.load1a = 925.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000,0,0,0),))
self.studyPositionDescriptions.extend(((0.05, 'b1', 2, (2)), (0.1, 'b1', 2, (2)), (0.15, 'b1', 2, (2)), (0.2, 'b1', 2, (2)), (0.25, 'b1', 2, (2)), (0.3, 'b1', 2, (2)), (0.35, 'b1', 2, (2)), (0.4, 'b1', 2, (2)), (0.45, 'b1', 2, (2)), (0.5, 'b1', 2, (2)), (0.55, 'b1', 2, (2)), (0.6, 'b1', 2, (2)), (0.65, 'b1', 2, (2)), (0.7, 'b1', 2, (2)), (0.75, 'b1', 2, (2)), (0.8, 'b1', 2, (2)))
```

413
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studDescriptions.extend(((0.05, 'b1', 2, (2)), (0.09, 'b1', 2, (2)), (0.095, 'b1', 2, (2)), (0.10, 'b1', 2, (2)), (0.105, 'b1', 2, (2)), (0.11, 'b1', 2, (2)), (0.115, 'b1', 2, (2)), (1.2, 'b1', 2, (2)), (1.25, 'b1', 2, (2)), (1.3, 'b1', 2, (2)), (1.35, 'b1', 2, (2)), (1.4, 'b1', 2, (2)), (1.45, 'b1', 2, (2))))
self.studPositionDescriptions.extend(((0.05, 'b2', 2, (2)), (0.09, 'b2', 2, (2)), (0.095, 'b2', 2, (2)), (0.10, 'b2', 2, (2)), (0.105, 'b2', 2, (2)), (0.11, 'b2', 2, (2)), (0.115, 'b2', 2, (2)), (0.12, 'b2', 2, (2)), (0.125, 'b2', 2, (2)), (0.13, 'b2', 2, (2)), (0.14, 'b2', 2, (2)), (1.0, 'b2', 2, (2)), (1.05, 'b2', 2, (2)), (1.1, 'b2', 2, (2)), (1.15, 'b2', 2, (2)), (1.2, 'b2', 2, (2)), (1.25, 'b2', 2, (2)), (1.3, 'b2', 2, (2)), (1.35, 'b2', 2, (2)), (1.4, 'b2', 2, (2)), (1.45, 'b2', 2, (2))))
self.studPositionDescriptions.extend(((0.1, 't1', 2, (2)), (0.15, 't1', 2, (2)), (0.2, 't1', 2, (2)), (0.25, 't1', 2, (2)), (0.3, 't1', 2, (2)), (0.35, 't1', 2, (2)), (0.4, 't1', 2, (2)), (0.45, 't1', 2, (2)), (0.5, 't1', 2, (2)), (0.55, 't1', 2, (2)), (0.6, 't1', 2, (2)), (0.65, 't1', 2, (2)), (0.7, 't1', 2, (2)), (0.75, 't1', 2, (2)), (0.8, 't1', 2, (2)), (0.85, 't1', 2, (2)), (0.9, 't1', 2, (2)), (0.95, 't1', 2, (2)), (1.0, 't1', 2, (2)), (1.05, 't1', 2, (2)), (1.1, 't1', 2, (2)), (1.15, 't1', 2, (2)), (1.2, 't1', 2, (2)), (1.25, 't1', 2, (2)), (1.3, 't1', 2, (2)), (1.35, 't1', 2, (2)), (1.4, 't1', 2, (2)), (1.45, 't1', 2, (2))))

print self.Case_Name
if self.Case_Name == 'Oduyemi_F1' or self.beamCaseNo==19:

    self.L = 1500.00/1000.0
    self.beamh = 150.00/1000.0
    self.beamb = 150.00/1000.0
    self.bottomt = 3.00/1000.0
    self.topf = 3.00/1000.0
    self.plateFyBottom = 268.00
    self.plateFyTop = 268.00
    self.plateFuBottom = 318.00
    self.plateFuTop = 318.00
    self.concretefc = 27.50

    self.studb1h = 0.08/1000.0
    self.studb1Dia = 0.08/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00+1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50

    self.studb2h = 135.00/1000.0
    self.studb2Dia = 6.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 9.45+1000.0
    self.studb2fy = 449.00
    self.studb2fu = 499.00

    self.studt1h = 35.00/1000.0
    self.studt1Dia = 6.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 9.45+1000.0
    self.studt1fy = 449.00
    self.studt1fu = 499.00

    self.studt2h = 0.08/1000.0
    self.studt2Dia = 0.08/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00+1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50

    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05

    self.support1 = 25.00/1000.0
    self.support1a = 1475.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 575.00/1000.0
    self.load1a = 925.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000.0, 0.0),))
self.studyPositionDescriptions.extend(((0.05, 'b2', 2,(2)),(0.15, 'b2', 2,(2)),(0.25, 'b2', 2,(2)),(0.35, 'b2', 2,(2)),(0.45, 'b2', 2,(2)),(0.55, 'b2', 2,(2)),(0.65, 'b2', 2,(2)),(0.75, 'b2', 2,(2)),(0.85, 'b2', 2,(2)),(0.95, 'b2', 2,(2)),(1.05, 'b2', 2,(2)),(1.15, 'b2', 2,(2)),(1.25, 'b2', 2,(2)),(1.35, 'b2', 2,(2)),(1.45, 'b2', 2,(2)),))
self.studyPositionDescriptions.extend(((0.0, 't1', 2,(2)),(0.1, 't1', 2,(2)),(0.2, 't1', 2,(2)),(0.3, 't1', 2,(2)),(0.4, 't1', 2,(2)),(0.5, 't1', 2,(2)),(0.6, 't1', 2,(2)),(0.7, 't1', 2,(2)),(0.8, 't1', 2,(2)),(0.9, 't1', 2,(2)),(1.0, 't1', 2,(2)),(1.1, 't1', 2,(2)),(1.2, 't1', 2,(2)),(1.3, 't1', 2,(2)),(1.4, 't1', 2,(2)),(1.5, 't1', 2,(2)),)
print self.Case_Name
if self.Case_Name == 'Oduyemi_F2' or self.beamCaseNo==20:
    self.L = 1500.00/1000.0
    self.beamh = 150.00/1000.0
    self.beamb = 150.00/1000.0
    self.bottomt = 3.00/1000.0
    self.top t = 3.00/1000.0
    self.plateFyBottom = 268.00
    self.plateFyTop = 268.00
    self.plateFuBottom = 318.00
    self.plateFuTop = 318.00
    self.concretef = 30.60
    self.studh1 = 35.00/1000.0
    self.stud1Dia = 6.00/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 10.13*1000.0
    self.stud1fy = 449.00
    self.stud1fu = 499.00
    self.stud2h = 135.00/1000.0
    self.stud2Dia = 6.00/1000.0
    self.stud2FullHeight = False
    self.stud2Resistance = 10.13*1000.0
    self.stud2fy = 449.00
    self.stud2fu = 499.00
    self.stud1h = 35.00/1000.0
    self.stud1Dia = 6.00/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 10.13*1000.0
    self.stud1fy = 449.00
    self.stud1fu = 499.00
    self.stud2h = 0.00/1000.0
    self.stud2Dia = 0.00/1000.0
    self.stud2FullHeight = False
    self.stud2Resistance = 0.00*1000.0
    self.stud2fy = 1.00
    self.stud2fu = 1.50
    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05
    self.support1 = 25.00/1000.0
    self.support1a = 1475.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
```

415
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.load1 = 575.00/1000.0
def.load1a = 925.00/1000.0
def.load1b = -100.00/1000.0
def.load1c = -100.00/1000.0
def.load1d = -100.00/1000.0
def.load1e = -100.00/1000.0
def.load1f = -100.00/1000.0
def.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
def.loadFactor1a = 1.00
def.loadFactor1b = -100.00
def.loadFactor1c = -100.00
def.loadFactor1d = -100.00
def.loadFactor1e = -100.00
def.loadFactor1f = -100.00
def.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000.0, 0.0), ))
self.studPositionDescriptions.extend(((0.05, 'b1', 2,(2)), (0.15, 'b1', 2,(2)), (0.25, 'b1', 2,(2)), (0.35, 'b1', 2,(2)), (0.45, 'b1', 2,(2)), (0.55, 'b1', 2,(2)), (0.65, 'b1', 2,(2)), (0.75, 'b1', 2,(2)), (0.85, 'b1', 2,(2)), (0.95, 'b1', 2,(2))
(1.05, 'b1', 2,(2)), (1.15, 'b1', 2,(2)), (1.25, 'b1', 2,(2)), (1.35, 'b1', 2,(2)), (1.45, 'b1', 2,(2)), ))
self.studPositionDescriptions.extend(((0.05, 'b2', 2,(2)), (0.15, 'b2', 2,(2)), (0.25, 'b2', 2,(2)), (0.35, 'b2', 2,(2)), (0.45, 'b2', 2,(2)), (0.55, 'b2', 2,(2)), (0.65, 'b2', 2,(2)), (0.75, 'b2', 2,(2)), (0.85, 'b2', 2,(2)), (0.95, 'b2', 2,(2))
(1.05, 'b2', 2,(2)), (1.15, 'b2', 2,(2)), (1.25, 'b2', 2,(2)), (1.35, 'b2', 2,(2)), (1.45, 'b2', 2,(2)), )
self.studPositionDescriptions.extend(((0.05, 't1', 2,(2)), (0.15, 't1', 2,(2)), (0.25, 't1', 2,(2)), (0.35, 't1', 2,(2)), (0.45, 't1', 2,(2)), (0.55, 't1', 2,(2)), (0.65, 't1', 2,(2)), (0.75, 't1', 2,(2)), (0.85, 't1', 2,(2)), (0.95, 't1', 2,(2))
(1.05, 't1', 2,(2)), (1.15, 't1', 2,(2)), (1.25, 't1', 2,(2)), (1.35, 't1', 2,(2)), (1.45, 't1', 2,(2)), ))
```

```python
print self.Case_Name
if self.Case_Name == 'Oduyemi_F3' or self.beamCaseNo==21:
    L = 1500.00/1000.0
    Lm = 150.00/1000.0
    bottomt = 6.00/1000.0
    top = 4.00/1000.0
    plateFyBottom = 288.00
    plateFyTop = 252.00
    plateFbBottom = 338.00
    plateFbTop = 302.00
    concreteFu = 29.80
    stud1h = 0.08/1000.0
    stud1Dia = 0.00/1000.0
    stud1FullHeight = False
    stud1Resistance = 0.00+1000.0
    stud1fy = 1.00
    stud1fu = 1.50
    stud2h = 135.00/1000.0
    stud2Dia = 6.00/1000.0
    stud2FullHeight = False
    stud2Resistance = 9.96+1000.0
    stud2fy = 449.00
    stud2fu = 499.00
    stud3h = 35.00/1000.0
    stud3Dia = 6.00/1000.0
    stud3FullHeight = False
    stud3Resistance = 9.96+1000.0
    stud3fy = 449.00
    stud3fu = 499.00
    stud2h = 0.00/1000.0
    stud2Dia = 0.00/1000.0
    stud2FullHeight = False
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self. studt2Resistance = 0.00 * 1000.0
self. studt2fy = 1.00
self. studt2fu = 1.50
self. CyclicLoad = 50.00 * 1000.0
self. appliedDeflection = 0.05
self. support1 = 25.00 / 1000.0
self. support1a = 1475.00 / 1000.0
self. support1b = -100.00 / 1000.0
self. support1c = -100.00 / 1000.0
self. load1 = 575.00 / 1000.0
self. load1a = 925.00 / 1000.0
self. load1b = -100.00 / 1000.0
self. load1c = -100.00 / 1000.0
self. load1d = -100.00 / 1000.0
self. load1e = -100.00 / 1000.0
self. load1f = -100.00 / 1000.0
self. load1g = -100.00 / 1000.0
self. loadFactor1 = 1.00
self. loadFactor1a = 1.00
self. loadFactor1b = -100.00
self. loadFactor1c = -100.00
self. loadFactor1d = -100.00
self. loadFactor1e = -100.00
self. loadFactor1f = -100.00
self. loadFactor1g = -100.00
self. loadDefPoints = []
self. loadDefPoints.extend(((750 / 1000.0, 0.0),))
self. studPositionDescriptions.extend(((0.05, 'b2', 3, (3)), (0.1, 'b2', 3, (3)), (0.15, 'b2', 3, (3)), (0.2, 'b2', 3, (3))
                                        , (0.25, 'b2', 3, (3)), (0.3, 'b2', 3, (3)), (0.35, 'b2', 3, (3)), (0.4, 'b2', 3, (3)), (0.45, 'b2', 3, (3)), (0.5, 'b2', 3, (3))
                                        , (0.55, 'b2', 3, (3)), (0.6, 'b2', 3, (3)), (0.65, 'b2', 3, (3)), (0.7, 'b2', 3, (3)), (0.75, 'b2', 3, (3))
                                        , (0.8, 'b2', 3, (3)), (0.85, 'b2', 3, (3)), (0.9, 'b2', 3, (3)), (0.95, 'b2', 3, (3)), (1, 'b2', 3, (3)), (1.05, 'b2', 3, (3)), (1.1, 'b2', 3, (3)), (1.15, 'b2', 3, (3))
                                        , (1.2, 'b2', 3, (3)), (1.25, 'b2', 3, (3)), (1.3, 'b2', 3, (3)), (1.35, 'b2', 3, (3)), (1.4, 'b2', 3, (3))
                                        , (1.45, 'b2', 3, (3)), (1.5, 'b2', 3, (3))))
self. studPositionDescriptions.extend(((0.0, 't1', 2, (2)), (0.1, 't1', 2, (2)), (0.2, 't1', 2, (2)), (0.3, 't1', 2, (2)),
                                        (0.4, 't1', 2, (2)), (0.5, 't1', 2, (2)), (0.6, 't1', 2, (2)), (0.7, 't1', 2, (2)),
                                        (0.8, 't1', 2, (2)), (0.9, 't1', 2, (2)), (1, 't1', 2, (2)), (1.1, 't1', 2, (2))
                                        , (1.2, 't1', 2, (2)), (1.3, 't1', 2, (2)), (1.4, 't1', 2, (2)),))

print self.Case_Name
if self.Case_Name == 'Oluwemi_F4' or self.beamCaseNo==22:

self.L = 1500.00 / 1000.0
self.beamh = 150.00 / 1000.0
self.beamb = 150.00 / 1000.0
self.bottomt = 6.00 / 1000.0
self.topt = 2.00 / 1000.0
self.plateFyBottom = 288.00
self.plateFyTop = 242.00
self.plateFuBottom = 338.00
self.plateFuTop = 292.00
self.concretefc = 27.60
self. studt1hth = 0.08 / 1000.0
self. studt1Dia = 0.09 / 1000.0
self. studt1FullHeight = False
self. studt1Resistance = 0.00 + 1000.0
self. studt1fy = 1.00
self. studt1fu = 1.50
self. studt2h = 135.00 / 1000.0
self. studt2Dia = 6.00 / 1000.0
self. studt2FullHeight = False
self. studt2Resistance = 9.47 + 1000.0
self. studt2fy = 449.00
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2fu = 499.00
self.studb2h = 0.08/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 9.47+1000.0
self.studb2fy = 449.00
self.studb2fu = 499.00

self.studt1h = 35.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 9.47+1000.0
self.studt1fy = 449.00
self.studt1fu = 499.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00+1000.0
self.studt2fy = 449.00
self.studt2fu = 499.00

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 25.00/1000.0
self.support1a = 1475.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 575.00/1000.0
self.load1a = 925.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadfactor1 = 1.00
self.loadfactor1a = 1.00
self.loadfactor1b = -100.00
self.loadfactor1c = -100.00
self.loadfactor1d = -100.00
self.loadfactor1e = -100.00
self.loadfactor1f = -100.00
self.loadfactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints . extend ( ((750/1000.0, 0.0), ))

self.studPositionDescriptions . extend ( ((0.05, 'b2', 3,(3)), (0.1, 'b2', 3,(3)), (0.15, 'b2', 3,(3)), (0.2, 'b2', 3,(3)), (0.25, 'b2', 3,(3)), (0.3, 'b2', 3,(3)), (0.35, 'b2', 3,(3)), (0.4, 'b2', 3,(3)), (0.45, 'b2', 3,(3)), (0.5, 'b2', 3,(3)), (0.55, 'b2', 3,(3)), (0.6, 'b2', 3,(3)), (0.65, 'b2', 3,(3)), (0.7, 'b2', 3,(3)), (0.75, 'b2', 3,(3)), (0.8, 'b2', 3,(3)), (0.85, 'b2', 3,(3)), (0.9, 'b2', 3,(3)), (0.95, 'b2', 3,(3)), (1.0, 'b2', 3,(3)), (1.05, 'b2', 3,(3)), (1.1, 'b2', 3,(3)), (1.15, 'b2', 3,(3)), (1.2, 'b2', 3,(3)), (1.25, 'b2', 3,(3)), (1.3, 'b2', 3,(3)), (1.35, 'b2', 3,(3)), (1.4, 'b2', 3,(3)), (1.45, 'b2', 3,(3)), (1.5, 'b2', 3,(3)), ))
self.studPositionDescriptions . extend ( ((0.0, 't1', 2,(2)), (0.05, 't1', 2,(2)), (0.1, 't1', 2,(2)), (0.15, 't1', 2,(2)), (0.2, 't1', 2,(2)), (0.25, 't1', 2,(2)), (0.3, 't1', 2,(2)), (0.35, 't1', 2,(2)), (0.4, 't1', 2,(2)), (0.45, 't1', 2,(2)), (0.5, 't1', 2,(2)), (0.55, 't1', 2,(2)), (0.6, 't1', 2,(2)), (0.65, 't1', 2,(2)), (0.7, 't1', 2,(2)), (0.75, 't1', 2,(2)), (0.8, 't1', 2,(2)), (0.85, 't1', 2,(2)), (0.9, 't1', 2,(2)), (0.95, 't1', 2,(2)), (1.0, 't1', 2,(2)), (1.05, 't1', 2,(2)), (1.1, 't1', 2,(2)), (1.15, 't1', 2,(2)), (1.2, 't1', 2,(2)), (1.25, 't1', 2,(2)), (1.3, 't1', 2,(2)), (1.35, 't1', 2,(2)), (1.4, 't1', 2,(2)), (1.45, 't1', 2,(2)), (1.5, 't1', 2,(2)), ))

print self.Case_Name
if self.Case_Name == 'Oduyemi_F5' or self.beamCaseNo==23:

self.L = 1500.00/1000.0
self.beamh = 150.00/1000.0
self.bamb = 150.00/1000.0
self.bottomt = 4.00/1000.0
self.topt = 4.00/1000.0

self.plateFyBottom = 252.00
self.plateFyTop = 252.00
self.plateFullBottom = 302.00
self.plateFuTop = 302.00
self.concreteFc = 31.30
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00∗1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 135.00/1000.0
self.studb2Dia = 6.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 10.28∗1000.0
self.studb2fy = 449.00
self.studb2fu = 499.00

self.studt1h = 35.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 10.28∗1000.0
self.studt1fy = 449.00
self.studt1fu = 499.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00∗1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00∗1000.0
self.appliedDeflection = 0.05

self.support1 = 25.00/1000.0
self.support1a = 1475.00/1000.0
self.support1b = −100.00/1000.0
self.support1c = −100.00/1000.0

self.load1 = 575.00/1000.0
self.load1a = 925.00/1000.0
self.load1b = −100.00/1000.0
self.load1c = −100.00/1000.0
self.load1d = −100.00/1000.0
self.load1e = −100.00/1000.0
self.load1f = −100.00/1000.0
self.load1g = −100.00/1000.0

self.loadDefPoints = []
self.loadDefPoints.extend(((750/1000,0.0,0.0),))
self.studPositionDescriptions.extend(((0.05,'b2',3,(3)),(0.15,'b2',3,(3)),(0.25,'b2',3,(3)),(0.35,'b2',3,(3)),(0.45,'b2',3,(3)),(0.55,'b2',3,(3)),(0.65,'b2',3,(3)),(0.75,'b2',3,(3)),(0.85,'b2',3,(3)),(0.95,'b2',3,(3)),(1.05,'b2',3,(3)),(1.15,'b2',3,(3)),(1.25,'b2',3,(3)),(1.35,'b2',3,(3)),(1.45,'b2',3,(3)),))

print self.Case_Name
if self.Case_Name == Varma_SP1−1 or self.beamCaseNo==24:

self.L = 3061.90/1000.0
self.beamh = 457.00/1000.0
self.beamb = 306.19/1000.0
```

419
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.bottomt = 6.40 / 1000.0
self.top = 6.40 / 1000.0
self.plateFyBottom = 448.20
self.plateFyTop = 448.20
self.plateFuBottom = 498.20
self.plateFuTop = 498.20
self.concreteFc = 42.10
self.studb1h = 63.5 / 1000.0
self.studb1Dia = 12.70 / 1000.0
self.studb1FullHeight = False
self.studb1Resistance = 49.45 * 1000.0
self.studb1fy = 438.00
self.studb1fu = 488.00
self.studb2h = 0.00 / 1000.0
self.studb2Dia = 0.00 / 1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00 * 1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50
self.stud1h = 63.5 / 1000.0
self.stud1Dia = 12.70 / 1000.0
self.stud1FullHeight = False
self.stud1Resistance = 49.45 * 1000.0
self.stud1fy = 438.00
self.stud1fu = 488.00
self.stud2h = 0.00 / 1000.0
self.stud2Dia = 0.00 / 1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00 * 1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50
self.CyclicLoad = 50.00 * 1000.0
self.appliedDeflection = 0.05
self.support1 = 76.20 / 1000.0
self.support1a = 2985.70 / 1000.0
self.support1b = -100.00 / 1000.0
self.support1c = -100.00 / 1000.0
self.load1 = 1530.95 / 1000.0
self.load1a = -100.00 / 1000.0
self.load1b = -100.00 / 1000.0
self.load1c = -100.00 / 1000.0
self.load1d = -100.00 / 1000.0
self.load1e = -100.00 / 1000.0
self.load1f = -100.00 / 1000.0
self.load1g = -100.00 / 1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((1530.95 / 1000.0, 0.0), ))
self.studPositionDescriptions.extend(((0.0762, 'b1', 2, (2)), (0.2286, 'b1', 2, (2)), (0.381, 'b1', 2, (2)), (0.5334, 'b1', 2, (2)), (0.6858, 'b1', 2, (2)), (0.8382, 'b1', 2, (2)), (0.8906, 'b1', 2, (2)), (1.143, 'b1', 2, (2)), (1.2954, 'b1', 2, (2)), (1.4478, 'b1', 2, (2)), (1.602, 'b1', 2, (2)), (1.7352, 'b1', 2, (2)), (1.8905, 'b1', 2, (2)), (2.0074, 'b1', 2, (2)), (2.0574, 'b1', 2, (2)), (2.2098, 'b1', 2, (2)), (2.3622, 'b1', 2, (2)), (2.5146, 'b1', 2, (2)), (2.667, 'b1', 2, (2)), (2.8194, 'b1', 2, (2)), (2.9718, 'b1', 2, (2)), )
```
APPENDIX B: TEST DATABSE IN PYTHON CODE FORMAT

```python
self.studPositionDescriptions.extend((
    (0.0762, 't1', 2, (2)),
    (0.2286, 't1', 2, (2)),
    (0.381, 't1', 2, (2)),
    (0.5334, 't1', 2, (2)),
    (0.6858, 't1', 2, (2)),
    (0.8382, 't1', 2, (2)),
    (0.9906, 't1', 2, (2)),
    (1.143, 't1', 2, (2)),
    (1.2954, 't1', 2, (2)),
    (1.4478, 't1', 2, (2)),
    (1.6002, 't1', 2, (2)),
    (1.7526, 't1', 2, (2)),
    (1.905, 't1', 2, (2)),
    (2.0574, 't1', 2, (2)),
    (2.2098, 't1', 2, (2)),
    (2.3622, 't1', 2, (2)),
    (2.5146, 't1', 2, (2)),
    (2.667, 't1', 2, (2)),
    (2.8194, 't1', 2, (2)),
    (2.9718, 't1', 2, (2))
))

print self.Case_Name

if self.Case_Name == 'Varma_SP1-2' or self.beamCaseNo==25:

    self.L = 3061.90/1000.0
    self.beamb = 457.00/1000.0
    self.beamb = 306.19/1000.0
    self.bottomt = 6.40/1000.0
    self.top t = 6.40/1000.0
    self.beamb = 306.19/1000.0
    self.bottomt = 6.40/1000.0
    self.top t = 6.40/1000.0
    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05
    self.support1 = 76.20/1000.0
    self.support1a = 2985.70/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 1530.95/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = -100.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
```

421
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1d = -100.0
self.loadFactor1e = -100.0
self.loadFactor1f = -100.0
self.loadFactor1g = -100.0

self.loadDefPoints = []
self.loadDefPoints.extend(((1530.95/1000.0,0.0), (1.0674/1000.0,0.0),
                           (2.8974/1000.0,0.0), (2.8974/1000.0,0.0)
                           ))
self.studyPositionDescriptions.extend(((0.1524,'b1',2,(2)),(0.4574,'b1',2,(2)),(0.7624,'b1',2,(2)),(1.0674,'b1',2,(2)),(1.3724,'b1',2,(2)),(1.6774,'b1',2,(2)),(1.9824,'b1',2,(2)),(2.2874,'b1',2,(2)),(2.5924,'b1',2,(2)),
                           (2.8974,'b1',2,(2)), (2.8974,'b1',2,(2)), (2.8974,'b1',2,(2)),(2.8974,'b1',2,(2)), (2.8974,'b1',2,(2)), (2.8974,'b1',2,(2)), (2.8974,'b1',2,(2)), (2.8974,'b1',2,(2)), (2.8974,'b1',2,(2)), (2.8974,'b1',2,(2)))

self.studyPositionDescriptions.extend(((0.1524,'t1',2,(2)),(0.4574,'t1',2,(2)),(0.7624,'t1',2,(2)),(1.0674,'t1',2,(2)),(1.3724,'t1',2,(2)),(1.6774,'t1',2,(2)),(1.9824,'t1',2,(2)),(2.2874,'t1',2,(2)),(2.5924,'t1',2,(2)),
                           (2.8974,'t1',2,(2)), (2.8974,'t1',2,(2)), (2.8974,'t1',2,(2)),(2.8974,'t1',2,(2)), (2.8974,'t1',2,(2)), (2.8974,'t1',2,(2)), (2.8974,'t1',2,(2)), (2.8974,'t1',2,(2)), (2.8974,'t1',2,(2)), (2.8974,'t1',2,(2)))

print self.Case_Name
if self.Case_Name == 'Varma_SP1-3' or self.beamCaseNo==26:

    self.L = 3061.90/1000.0
    self.beamb = 457.00/1000.0
    self.bemma = 306.19/1000.0
    self.bottomt = 9.50/1000.0
    self.topt = 9.50/1000.0
    self.plateFyBottom = 448.20
    self.plateFyTop = 448.20
    self.plateFuBottom = 498.20
    self.plateFuTop = 498.20
    self.concretefc = 42.10
    self.studb1h = 63.5/1000.0
    self.studb1Dia = 12.70/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 49.45*1000.0
    self.studb1fy = 438.00
    self.studb1fu = 488.00
    self.studb2h = 0.00/1000.0
    self.studb2Dia = 0.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 0.00*1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50
    self.studt1h = 63.5/1000.0
    self.studt1Dia = 12.70/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 49.45*1000.0
    self.studt1fy = 438.00
    self.studt1fu = 488.00
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00*1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05
    self.support1 = 76.20/1000.0
    self.support1a = 2985.70/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 1530.95/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
```

422
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend([(1530.95/1000.0, 0.0)])

self.studPositionDescriptions.extend([(0.0762, 'b1', 2, (2)), (0.2286, 'b1', 2, (2)), (0.381, 'b1', 2, (2)), (0.5334, 'b1', 2, (2)), (0.6858, 'b1', 2, (2)), (0.8382, 'b1', 2, (2)), (0.9906, 'b1', 2, (2)), (1.143, 'b1', 2, (2)), (1.2954, 'b1', 2, (2)), (1.4478, 'b1', 2, (2)), (1.6002, 'b1', 2, (2)), (1.7526, 'b1', 2, (2)), (1.905, 'b1', 2, (2)), (2.0574, 'b1', 2, (2)), (2.2098, 'b1', 2, (2)), (2.3622, 'b1', 2, (2)), (2.5146, 'b1', 2, (2)), (2.667, 'b1', 2, (2)), (2.8194, 'b1', 2, (2)), (2.9718, 'b1', 2, (2))])

self.studPositionDescriptions.extend([(0.0762, 't1', 2, (2)), (0.2286, 't1', 2, (2)), (0.381, 't1', 2, (2)), (0.5334, 't1', 2, (2)), (0.6858, 't1', 2, (2)), (0.8382, 't1', 2, (2)), (0.9906, 't1', 2, (2)), (1.143, 't1', 2, (2)), (1.2954, 't1', 2, (2)), (1.4478, 't1', 2, (2)), (1.6002, 't1', 2, (2)), (1.7526, 't1', 2, (2)), (1.905, 't1', 2, (2)), (2.0574, 't1', 2, (2)), (2.2098, 't1', 2, (2)), (2.3622, 't1', 2, (2)), (2.5146, 't1', 2, (2)), (2.667, 't1', 2, (2)), (2.8194, 't1', 2, (2)), (2.9718, 't1', 2, (2))])

print self.Case_Name
if self.Case_Name == 'Varma_SP1-4' or self.beamCaseNo==27:

    self.L = 2422.10/1000.0
    self.beamh = 457.00/1000.0
    self.beamb = 306.19/1000.0
    self.bottomt = 6.40/1000.0
    self.topt = 6.40/1000.0

    self.plateFyBottom = 448.20
    self.plateFyTop = 448.20
    self.plateFuBottom = 448.20
    self.plateFuTop = 448.20
    self.concretefc = 42.10

    self.stud1h = 63.5/1000.0
    self.stud1Dia = 12.70/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 49.45+1000.0
    self.stud1fy = 438.00
    self.stud1fu = 488.00

    self.stud2h = 0.00/1000.0
    self.stud2Dia = 0.00/1000.0
    self.stud2FullHeight = False
    self.stud2Resistance = 0.00+1000.0
    self.stud2fy = 1.00
    self.stud2fu = 1.50

    self.stud1h = 63.5/1000.0
    self.stud1Dia = 12.70/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 49.45+1000.0
    self.stud1fy = 438.00
    self.stud1fu = 488.00

    self.stud2h = 0.00/1000.0
    self.stud2Dia = 0.00/1000.0
    self.stud2FullHeight = False
    self.stud2Resistance = 0.00+1000.0
    self.stud2fy = 1.00
    self.stud2fu = 1.50
```

423
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 76.20/1000.0
self.support1a = 2345.90/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1211.05/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1211.05/1000.0, 0.0),))

self.studPositionDescriptions.extend(((0.0762, 'b1', 2, (2)), (0.2286, 'b1', 2, (2)), (0.381, 'b1', 2, (2)), (0.5334, 'b1', 2, (2)), (0.6858, 'b1', 2, (2)), (0.8382, 'b1', 2, (2)), (0.9906, 'b1', 2, (2)), (1.143, 'b1', 2, (2)), (1.2954, 'b1', 2, (2)), (1.4478, 'b1', 2, (2)), (1.6002, 'b1', 2, (2)), (1.7526, 'b1', 2, (2)), (1.905, 'b1', 2, (2)), (2.0574, 'b1', 2, (2)), (2.2098, 'b1', 2, (2)), (2.3622, 'b1', 2, (2)),))
self.studPositionDescriptions.extend(((0.0762, 't1', 2, (2)), (0.2286, 't1', 2, (2)), (0.381, 't1', 2, (2)), (0.5334, 't1', 2, (2)), (0.6858, 't1', 2, (2)), (0.8382, 't1', 2, (2)), (0.9906, 't1', 2, (2)), (1.143, 't1', 2, (2)), (1.2954, 't1', 2, (2)), (1.4478, 't1', 2, (2)), (1.6002, 't1', 2, (2)), (1.7526, 't1', 2, (2)), (1.905, 't1', 2, (2)), (2.0574, 't1', 2, (2)), (2.2098, 't1', 2, (2)), (2.3622, 't1', 2, (2)),))

print self.Case_Name
if self.Case_Name == 'Varma_SP1-5' or self.beamCaseNo==28:
    self.L = 10054.00/1000.0
    self.beamb = 914.00/1000.0
    self.beamb = 914.00/1000.0
    self.bottomt = 12.70/1000.0
    self.topt = 12.70/1000.0
    self.plateFyBottom = 448.20
    self.plateFyTop = 448.20
    self.plateFuBottom = 498.20
    self.plateFuTop = 498.20
    self.concretefc = 42.70
    self.studb1h = 152.40/1000.0
    self.studb1Dia = 9.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 111.14*1000.0
    self.studb1fy = 440.00
    self.studb1fu = 490.00
    self.studb2h = 0.00/1000.0
    self.studb2Dia = 0.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 0.00*1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50
    self.studt1h = 152.40/1000.0
    self.studt1Dia = 19.00/1000.0
```
self.stud1FullHeight = False
self.stud1Resistance = 111.14*1000.0
self.stud1fy = 440.0
self.stud1fu = 490.0
self.stud2h = 0.08/1000.0
self.stud2Dia = 0.09/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00*1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CycleLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 914.00/1000.0
self.support1a = 9140.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 4113.00/1000.0
self.load1a = 5941.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1l = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((5027/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.0762, 'b1', 3, (3)), (0.3302, 'b1', 3, (3)), (0.5842, 'b1', 3, (3)), (0.8382, 'b1', 3, (3)), (1.0922, 'b1', 3, (3)), (1.3462, 'b1', 3, (3)), (1.6002, 'b1', 3, (3)), (1.8542, 'b1', 3, (3)), (2.1082, 'b1', 3, (3)),
(2.3622, 'b1', 3, (3)), (2.6162, 'b1', 3, (3)), (2.8702, 'b1', 3, (3)), (3.1242, 'b1', 3, (3)), (3.3782, 'b1', 3, (3)), (3.6322, 'b1', 3, (3)), (3.8862, 'b1', 3, (3)), (4.1402, 'b1', 3, (3)), (4.3942, 'b1', 3, (3)), (4.6482, 'b1', 3, (3)), (4.9022, 'b1', 3, (3)), (5.1562, 'b1', 3, (3)), (5.4102, 'b1', 3, (3)), (5.6642, 'b1', 3, (3)), (5.9182, 'b1', 3, (3)), (6.1722, 'b1', 3, (3)), (6.4262, 'b1', 3, (3)), (6.6802, 'b1', 3, (3)), (6.9342, 'b1', 3, (3)), (7.1882, 'b1', 3, (3)), (7.4422, 'b1', 3, (3)), (7.6962, 'b1', 3, (3)), (7.9502, 'b1', 3, (3)), (8.2042, 'b1', 3, (3)), (8.4582, 'b1', 3, (3)), (8.7122, 'b1', 3, (3)), (8.9662, 'b1', 3, (3)), (9.2202, 'b1', 3, (3)), (9.4742, 'b1', 3, (3)), (9.7282, 'b1', 3, (3)),))

self.studPositionDescriptions.extend(((0.0762, 't1', 3, (3)), (0.3302, 't1', 3, (3)), (0.5842, 't1', 3, (3)), (0.8382, 't1', 3, (3)), (1.0922, 't1', 3, (3)), (1.3462, 't1', 3, (3)), (1.6002, 't1', 3, (3)), (1.8542, 't1', 3, (3)), (2.1082, 't1', 3, (3)), (2.3622, 't1', 3, (3)), (2.6162, 't1', 3, (3)), (2.8702, 't1', 3, (3)), (3.1242, 't1', 3, (3)), (3.3782, 't1', 3, (3)), (3.6322, 't1', 3, (3)), (3.8862, 't1', 3, (3)), (4.1402, 't1', 3, (3)), (4.3942, 't1', 3, (3)), (4.6482, 't1', 3, (3)), (4.9022, 't1', 3, (3)), (5.1562, 't1', 3, (3)), (5.4102, 't1', 3, (3)), (5.6642, 't1', 3, (3)), (5.9182, 't1', 3, (3)), (6.1722, 't1', 3, (3)), (6.4262, 't1', 3, (3)), (6.6802, 't1', 3, (3)), (6.9342, 't1', 3, (3)), (7.1882, 't1', 3, (3)), (7.4422, 't1', 3, (3)), (7.6962, 't1', 3, (3)), (7.9502, 't1', 3, (3)), (8.2042, 't1', 3, (3)), (8.4582, 't1', 3, (3)), (8.7122, 't1', 3, (3)), (8.9662, 't1', 3, (3)), (9.2202, 't1', 3, (3)), (9.4742, 't1', 3, (3)), (9.7282, 't1', 3, (3)),))

print self.Case_Name
if self.Case_Name == 'Varma_SP2a-1' or self.beamCaseNo==29:

self.L = 10054.00/1000.0
self.bamb = 914.00/1000.0
self.bamb = 859.16/1000.0
self.bottom = 19.10/1000.0
self.top = 19.10/1000.0

self.plateFyBottom = 399.90
self.plateFyTop = 399.90
self.plateFuBottom = 449.90

425
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.plateFuTop = 449.90
self.concretefc = 4830
self.studb1h = 152.40/1000.0
self.studb1Dia = 19.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 125.02/1000.0
self.studb1fy = 424.00
self.studb1fu = 551.20
self.studb2h = 457.00/1000.0
self.studb2Dia = 19.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 136.67/1000.0
self.studb2fy = 595.00
self.studb2fu = 645.00
self.stud1h = 152.40/1000.0
self.stud1Dia = 19.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 9167.19*1000.0
self.stud1fy = 374.00
self.stud1fu = 424.00
self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00/1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50
self.CyclicLoad = 50.00/1000.0
self.appliedDeflection = 0.05
self.support1 = 914.00/1000.0
self.support1a = 9140.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 4113.00/1000.0
self.load1a = 5941.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((5027/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.2285,'b1',3,(3)),(0.6885,'b1',3,(3)),(1.1425,'b1',3,(3)),(1.5995,'b1',3,(3)),(2.0565,'b1',3,(3)),(2.5133,'b1',3,(3)),(2.9705,'b1',3,(3)),(3.4275,'b1',3,(3)),(3.8845,'b1',3,(3)),(4.3415,'b1',3,(3)),(4.7985,'b1',3,(3)),(5.2555,'b1',3,(3)),(5.7125,'b1',3,(3)),(6.1695,'b1',3,(3)),(6.6265,'b1',3,(3)),(7.0835,'b1',3,(3)),(7.5405,'b1',3,(3)),(7.9975,'b1',3,(3)),(8.4545,'b1',3,(3)),(8.9115,'b1',3,(3)),(9.3685,'b1',3,(3)),(9.8255,'b1',3,(3)),(10.2825,'b1',3,(3)),(10.7395,'b1',3,(3)),(11.1965,'b1',3,(3)),(11.6535,'b1',3,(3)),(12.1105,'b1',3,(3)),(12.5675,'b1',3,(3)),(13.0245,'b1',3,(3)),(13.4815,'b1',3,(3)),(13.9385,'b1',3,(3)),(14.3955,'b1',3,(3)),(14.8525,'b1',3,(3)),(15.3095,'b1',3,(3)),(15.7665,'b1',3,(3)),(16.2235,'b1',3,(3)),(16.6805,'b1',3,(3)),(17.1375,'b1',3,(3)),(17.5945,'b1',3,(3)),(18.0515,'b1',3,(3)),(18.5085,'b1',3,(3)),(18.9655,'b1',3,(3)),(19.4225,'b1',3,(3)),(19.8795,'b1',3,(3)),(20.3365,'b1',3,(3)),(20.7935,'b1',3,(3)),(21.2505,'b1',3,(3)),(21.7075,'b1',3,(3)),(22.1645,'b1',3,(3)),(22.6215,'b1',3,(3)),(23.0785,'b1',3,(3)),(23.5355,'b1',3,(3)),(24.0005,)))
```

426
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studPositionDescriptions.extend(((0.2285, 't1', 3, (3)), (0.6855, 't1', 3, (3)), (1.1425, 't1', 3, (3)), (1.5995, 't1', 3, (3)), (2.0565, 't1', 3, (3)), (2.5135, 't1', 3, (3)), (2.9705, 't1', 3, (3)), (3.4275, 't1', 3, (3)), (3.8845, 't1', 3, (3)), (4.3415, 't1', 3, (3)), (4.7985, 't1', 3, (3)), (5.2555, 't1', 3, (3)), (5.7125, 't1', 3, (3)), (6.1695, 't1', 3, (3)), (6.6265, 't1', 3, (3)), (7.0835, 't1', 3, (3)), (7.5405, 't1', 3, (3)), (7.9975, 't1', 3, (3)), (8.4575, 't1', 3, (3)), (8.9145, 't1', 3, (3)), (9.3685, 't1', 3, (3)), (9.8255, 't1', 3, (3)), ))

print self.Case_Name

if self.Case_Name == 'Varma_SP2a-2' or self.beamCaseNo==30:

    self.L = 10054.00/1000.0
    self.beammh = 914.00/1000.0
    self.beamb = 859.16/1000.0
    self.bottomt = 19.10/1000.0
    self.topt = 19.10/1000.0

    self.plateFyBottom = 387.50
    self.plateFyTop = 387.50
    self.plateFuBottom = 437.50
    self.plateFuTop = 437.50

    self.concretefc = 48.30

    self.studth1 = 152.40/1000.0
    self.studth1Dia = 19.00/1000.0
    self.studth1FullHeight = False
    self.studth1Resistance = 96.17*1000.0
    self.studth1fy = 374.00
    self.studth1fu = 424.00

    self.studth2h = 457.00/1000.0
    self.studth2Dia = 19.00/1000.0
    self.studth2FullHeight = True
    self.studth2Resistance = 136.67*1000.0
    self.studth2fy = 595.00
    self.studth2fu = 645.00

    self.studth3 = 152.40/1000.0
    self.studth3Dia = 19.00/1000.0
    self.studth3FullHeight = False
    self.studth3Resistance = 96.17*1000.0
    self.studth3fy = 374.00
    self.studth3fu = 424.00

    self.studth2h = 0.08/1000.0
    self.studth2Dia = 0.00/1000.0
    self.studth2FullHeight = False
    self.studth2Resistance = 0.00+1000.0
    self.studth2fy = 1.00
    self.studth2fu = 1.50

    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05

    self.support1 = 914.00/1000.0
    self.support1a = 914.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0

    self.load1 = 4113.00/1000.0
    self.load1a = 5941.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
```

427
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFa = -100.00/1000.0
self.loadDg = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactorLa = 1.00
self.loadFactorNb = -100.00
self.loadFactorNc = -100.00
self.loadFactorId = -100.00
self.loadFactorIf = -100.00
self.loadFactorIg = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((0.5027/1000.0, 0.0, 0.0)))

self.StudPositionDescriptions.extend(((0.2285, 'b1', 3, (3)), (0.6855, 'b1', 3, (3)), (1.1425, 'b1', 3, (3)), (1.5995, 'b1', 3, (3)), (2.0655, 'b1', 3, (3)), (2.5135, 'b1', 3, (3)), (2.9705, 'b1', 3, (3)), (3.4275, 'b1', 3, (3)), (3.8845, 'b1', 3, (3)), (4.3415, 'b1', 3, (3)), (4.7985, 'b1', 3, (3)), (5.2555, 'b1', 3, (3)), (5.7125, 'b1', 3, (3)), (6.1695, 'b1', 3, (3)), (6.6265, 'b1', 3, (3)), (7.0835, 'b1', 3, (3)), (7.5405, 'b1', 3, (3)), (7.9975, 'b1', 3, (3)), (8.4545, 'b1', 3, (3)), (8.9115, 'b1', 3, (3)), (9.3685, 'b1', 3, (3)), (9.8255, 'b1', 3, (3)), (1.047, 'b1', 3, (3)), (1.0914, 'b1', 3, (3)), (1.371, 'b1', 3, (3)), (1.828, 'b1', 3, (3)), (2.828, 'b1', 3, (3)), (2.742, 'b1', 3, (3)), (3.199, 'b1', 3, (3)), (3.656, 'b1', 3, (3)), (4.113, 'b1', 3, (3)), (4.57, 'b1', 3, (3)), (5.027, 'b1', 3, (3)), (5.484, 'b1', 3, (3)), (5.941, 'b1', 3, (3)), (6.398, 'b1', 3, (3)), (6.855, 'b1', 3, (3)), (7.312, 'b1', 3, (3)), (7.769, 'b1', 3, (3)), (8.226, 'b1', 3, (3)), (8.683, 'b1', 3, (3)), (9.14, 'b1', 3, (3)), (9.597, 'b1', 3, (3)))

self.StudPositionDescriptions.extend(((0.2285, 'b2', 2, (2)), (0.6855, 'b2', 2, (2)), (1.1425, 'b2', 2, (2)), (1.5995, 'b2', 2, (2)), (2.0655, 'b2', 2, (2)), (2.5135, 'b2', 2, (2)), (2.9705, 'b2', 2, (2)), (3.4275, 'b2', 2, (2)), (3.8845, 'b2', 2, (2)), (4.3415, 'b2', 2, (2)), (4.7985, 'b2', 2, (2)), (5.2555, 'b2', 2, (2)), (5.7125, 'b2', 2, (2)), (6.1695, 'b2', 2, (2)), (6.6265, 'b2', 2, (2)), (7.0835, 'b2', 2, (2)), (7.5405, 'b2', 2, (2)), (7.9975, 'b2', 2, (2)), (8.4545, 'b2', 2, (2)), (8.9115, 'b2', 2, (2)), (9.3685, 'b2', 2, (2)), (9.8255, 'b2', 2, (2)), (9.8255, 'b2', 2, (2)), (10.2825, 'b2', 2, (2)), (10.7405, 'b2', 2, (2)), (11.1975, 'b2', 2, (2)), (11.6555, 'b2', 2, (2)), (12.1125, 'b2', 2, (2)), (12.5705, 'b2', 2, (2)), (13.0285, 'b2', 2, (2)), (13.4865, 'b2', 2, (2)), (13.9445, 'b2', 2, (2)), (14.3925, 'b2', 2, (2)), (14.8505, 'b2', 2, (2)), (15.3085, 'b2', 2, (2)), (15.7665, 'b2', 2, (2)), (16.2245, 'b2', 2, (2)), (16.6825, 'b2', 2, (2)))

self.StudPositionDescriptions.extend(((0.2285, 't1', 3, (3)), (0.6855, 't1', 3, (3)), (1.1425, 't1', 3, (3)), (1.5995, 't1', 3, (3)), (2.0655, 't1', 3, (3)), (2.5135, 't1', 3, (3)), (2.9705, 't1', 3, (3)), (3.4275, 't1', 3, (3)), (3.8845, 't1', 3, (3)), (4.3415, 't1', 3, (3)), (4.7985, 't1', 3, (3)), (5.2555, 't1', 3, (3)), (5.7125, 't1', 3, (3)), (6.1695, 't1', 3, (3)), (6.6265, 't1', 3, (3)), (7.0835, 't1', 3, (3)), (7.5405, 't1', 3, (3)), (7.9975, 't1', 3, (3)), (8.4545, 't1', 3, (3)), (8.9115, 't1', 3, (3)), (9.3685, 't1', 3, (3)), (9.8255, 't1', 3, (3)), (10.2825, 't1', 3, (3)), (10.7405, 't1', 3, (3)), (11.1975, 't1', 3, (3)), (11.6555, 't1', 3, (3)), (12.1125, 't1', 3, (3)), (12.5705, 't1', 3, (3)), (13.0285, 't1', 3, (3)), (13.4865, 't1', 3, (3)), (13.9445, 't1', 3, (3)), (14.3925, 't1', 3, (3)), (14.8505, 't1', 3, (3)), (15.3085, 't1', 3, (3)), (15.7665, 't1', 3, (3)), (16.2245, 't1', 3, (3)), (16.6825, 't1', 3, (3)))

print self.Case_Name

if self.Case_Name == 'Varma_SP2a-3' or self.beamCaseNo==31:
    self.L = 6398.00/1000.0
    self.beam = 914.00/1000.0
    self.bamb = 859.16/1000.0
    self.bottomt = 19.10/1000.0
    self.top = 19.10/1000.0
    self.plateFyBottom = 426.10
    self.plateFyTop = 426.10
    self.plateFullBottom = 476.10
    self.plateFullTop = 476.10
    self.concretefc = 52.10

    self.studb1h = 152.40/1000.0
    self.studb1Dia = 19.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 96.17*1000.0
    self.studb1fyy = 374.00
    self.studb1fu = 424.00

    self.studb2h = 457.00/1000.0
    self.studb2Dia = 19.00/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 143.57*1000.0
    self.studb2fyy = 595.00
    self.studb2fu = 645.00

    self.stud1h = 152.40/1000.0
    self.stud1Dia = 19.00/1000.0
```

428
self.studt1FullHeight = False
self.studt1Resistance = 96.17 * 1000.0
self.stud1fy = 374.00
self.stud1fu = 424.00

self.stud2h = 0.00 / 1000.0
self.stud2Dia = 0.00 / 1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00 * 1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00 * 1000.0
self.appliedDeflection = 0.05

self.support1 = 228.50 / 1000.0
self.support1a = 6169.50 / 1000.0
self.support1b = -100.00 / 1000.0
self.support1c = -100.00 / 1000.0
self.load1 = 2513.50 / 1000.0
self.load1a = 3884.50 / 1000.0
self.load1b = -100.00 / 1000.0
self.load1c = -100.00 / 1000.0
self.load1d = -100.00 / 1000.0
self.load1e = -100.00 / 1000.0
self.load1f = -100.00 / 1000.0
self.load1g = -100.00 / 1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((3199 / 1000.0, 0.0), ))

self.studt2Resistance = 0.00 * 1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.loadDefPoints.extend(((0.2285, 'b1', 3, (3)), (0.6855, 'b1', 3, (3)), (1.1425, 'b1', 3, (3)), (1.5995, 'b1', 3, (3)), (2.0565, 'b1', 3, (3)), (2.5135, 'b1', 3, (3)), (2.9705, 'b1', 3, (3)), (3.4275, 'b1', 3, (3)), (3.8845, 'b1', 3, (3)), (4.3415, 'b1', 3, (3)), (4.7985, 'b1', 3, (3)), (5.2555, 'b1', 3, (3)), (5.7125, 'b1', 3, (3)), (6.1695, 'b1', 3, (3)), (6.6265, 'b1', 3, (3)), (7.0835, 'b1', 3, (3)), (7.5405, 'b1', 3, (3)), (7.9975, 'b1', 3, (3)), (8.4545, 'b1', 3, (3)), (8.9115, 'b1', 3, (3)), (9.3685, 'b1', 3, (3)), (9.8255, 'b1', 3, (3)), (10.2825, 'b1', 3, (3)))

self.loadDefPoints.extend(((3199 / 1000.0, 0.0), ))

print self.Case_Name
if self.Case_Name == 'Varma_SP2a-4' or self.beamCaseNo==32:

self.L = 12156.20 / 1000.0
self.beamb = 914.00 / 1000.0
self.bottomt = 19.10 / 1000.0
self.topt = 19.10 / 1000.0

self.plateFyBottom = 387.50
self.plateFyTop = 387.50
self.plateFuBottom = 437.50
self.plateFuTop = 437.50
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.concreteE = 51.00
self.studD1h = 152.40/1000.0
self.studD1Dia = 19.00/1000.0
self.studD1FullHeight = False
self.studD1Resistance = 96.17+1000.0
self.studD1fy = 374.00
self.studD1fu = 424.00
self.studD2h = 457.00/1000.0
self.studD2Dia = 19.00/1000.0
self.studD2FullHeight = True
self.studD2Resistance = 141.59+1000.0
self.studD2fy = 595.00
self.studD2fu = 645.00
self.studT1h = 152.40/1000.0
self.studT1Dia = 19.00/1000.0
self.studT1FullHeight = False
self.studT1Resistance = 96.17+1000.0
self.studT1fy = 374.00
self.studT1fu = 424.00
self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05
self.support1 = 457.00/1000.0
self.support1a = 11699.20/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 5484.00/1000.0
self.load1a = 6672.20/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((6078.1/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.2285, 'b1', 3,(3)),(0.6855, 'b1', 3,(3)),(1.1425, 'b1', 3,(3)),(1.5995, 'b1', 3,(3)),(2.0565, 'b1', 3,(3)),(2.5135, 'b1', 3,(3)),(2.9705, 'b1', 3,(3)),(3.4275, 'b1', 3,(3)),(3.8845, 'b1', 3,(3)),(4.3415, 'b1', 3,(3)),(4.7985, 'b1', 3,(3)),(5.2555, 'b1', 3,(3)),(5.7125, 'b1', 3,(3)),(6.1695, 'b1', 3,(3)),(6.6265, 'b1', 3,(3)),(7.0835, 'b1', 3,(3)),(7.5405, 'b1', 3,(3)),(7.9975, 'b1', 3,(3)),(8.4545, 'b1', 3,(3)),(8.9115, 'b1', 3,(3)),(9.3685, 'b1', 3,(3)),(9.8255, 'b1', 3,(3)),(10.2825, 'b1', 3,(3)),(10.7395, 'b1', 3,(3)),(11.1965, 'b1', 3,(3)),(11.5355, 'b1', 3,(3)),(0.457, 'b1', 3,(3)),(0.914, 'b1', 3,(3)),(1.371, 'b1', 3,(3)),(1.828, 'b1', 3,(3)),(2.285, 'b1', 3,(3)),(2.742, 'b1', 3,(3)),(3.199, 'b1', 3,(3)),(3.658, 'b1', 3,(3)),(4.113, 'b1', 3,(3)),(4.57, 'b1', 3,(3)),(5.027, 'b1', 3,(3)),(5.484, 'b1', 3,(3)),(5.941, 'b1', 3,(3)),(6.398, 'b1', 3,(3)),(6.855, 'b1', 3,(3)),(7.312, 'b1', 3,(3)),(7.769, 'b1', 3,(3)),(8.226, 'b1', 3,(3)),(8.683, 'b1', 3,(3)),(9.14, 'b1', 3,(3)),(9.597, 'b1', 3,(3)),(10.054, 'b1', 3,(3)),(10.511, 'b1', 3,(3)),(10.968, 'b1', 3,(3)),(11.425, 'b1', 3,(3)),(11.882, 'b1', 3,(3)))
self.studPositionDescriptions.extend(((0.2285, 'b2', 2,(2)),(0.6855, 'b2', 2,(2)),(1.1425, 'b2', 2,(2)),(1.5995, 'b2', 2,(2)),(2.0565, 'b2', 2,(2)),(2.5135, 'b2', 2,(2)),(2.9705, 'b2', 2,(2)),(3.4275, 'b2', 2,(2)),(3.8845, 'b2', 2,(2))
```

APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
print self.Case_Name
if self.Case_Name == 'Varma_SP2c−1' or self.beamCaseNo==33:
    self.L = 5638.80/1000.0
    self.beamh = 762.00/1000.0
    self.beamb = 381.00/1000.0
    self.bottomt = 19.10/1000.0
    self.top t = 19.10/1000.0
    self.plateFyBottom = 419.20
    self.plateFyTop = 419.20
    self.plateFyBottom = 469.20
    self.plateFyTop = 469.20
    self.concretefc = 41.70
    self.studb1h = 152.40/1000.0
    self.studb1Dia = 12.70/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 42.97*1000.0
    self.studb1fy = 374.00
    self.studb1fu = 424.00
    self.studb2h = 381.00/1000.0
    self.studb2Dia = 15.46/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 42.97*1000.0
    self.studb2fy = 416.40
    self.studb2fu = 466.40
    self.studt1h = 127.00/1000.0
    self.studt1Dia = 12.70/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 42.97*1000.0
    self.studt1fy = 374.00
    self.studt1fu = 424.00
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00*1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05
    self.load1 = 2286.00/1000.0
    self.load2a = 3352.80/1000.0
    self.load2b = −100.00/1000.0
    self.load2c = −100.00/1000.0
```

431
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadId = -100.0/1000.0
self.loadId1 = -100.0/1000.0
self.loadId2 = -100.0/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.0
self.loadFactor2 = -100.0
self.loadFactor3 = -100.0
self.loadFactor4 = 1.00
self.loadDefPoints = [(0.2319, 'b1', 3, (3))]

self.loadPositionDescriptions.extend([(0.0319, 't1', 3, (3)), (0.1239, 't1', 3, (3)), (0.2159, 't1', 3, (3)), (0.3079, 't1', 3, (3)), (0.4009, 't1', 3, (3)), (0.4929, 't1', 3, (3)), (0.5849, 't1', 3, (3)), (0.6769, 't1', 3, (3)), (0.7689, 't1', 3, (3)), (0.8609, 't1', 3, (3)), (0.9529, 't1', 3, (3)), (1.0449, 't1', 3, (3))])

self.loadFactor1d = -1.00
self.loadFactor1e = -1.00
self.loadFactor1f = -1.00
self.loadFactor1g = -1.00
self.loadFactor1h = -1.00
self.loadFactor1i = -1.00
self.loadFactor1j = -1.00
self.loadFactor1k = -1.00
self.loadFactor1l = -1.00
self.loadFactor1m = -1.00
self.loadFactor1n = -1.00
self.loadFactor1o = -1.00
self.loadFactor1p = -1.00
self.loadFactor1q = -1.00
self.loadFactor1r = -1.00
self.loadFactor1s = -1.00
self.loadFactor1t = -1.00
self.loadFactor1u = -1.00
self.loadFactor1v = -1.00
self.loadFactor1w = -1.00
self.loadFactor1x = -1.00
self.loadFactor1y = -1.00
self.loadFactor1z = -1.00
self.loadFactor2a = -1.00
self.loadFactor2b = -1.00
self.loadFactor2c = -1.00
self.loadFactor2d = -1.00
self.loadFactor2e = -1.00
self.loadFactor2f = -1.00
self.loadFactor2g = -1.00
self.loadFactor2h = -1.00
self.loadFactor2i = -1.00
self.loadFactor2j = -1.00
self.loadFactor2k = -1.00
self.loadFactor2l = -1.00
self.loadFactor2m = -1.00
self.loadFactor2n = -1.00
self.loadFactor2o = -1.00
self.loadFactor2p = -1.00
self.loadFactor2q = -1.00
self.loadFactor2r = -1.00
self.loadFactor2s = -1.00
self.loadFactor2t = -1.00
self.loadFactor2u = -1.00
self.loadFactor2v = -1.00
self.loadFactor2w = -1.00
self.loadFactor2x = -1.00
self.loadFactor2y = -1.00
self.loadFactor2z = -1.00
self.loadFactor3a = -1.00
self.loadFactor3b = -1.00
self.loadFactor3c = -1.00
self.loadFactor3d = -1.00
self.loadFactor3e = -1.00
self.loadFactor3f = -1.00
self.loadFactor3g = -1.00
self.loadFactor3h = -1.00
self.loadFactor3i = -1.00
self.loadFactor3j = -1.00
self.loadFactor3k = -1.00
self.loadFactor3l = -1.00
self.loadFactor3m = -1.00
self.loadFactor3n = -1.00
self.loadFactor3o = -1.00
self.loadFactor3p = -1.00
self.loadFactor3q = -1.00
self.loadFactor3r = -1.00
self.loadFactor3s = -1.00
self.loadFactor3t = -1.00
self.loadFactor3u = -1.00
self.loadFactor3v = -1.00
self.loadFactor3w = -1.00
self.loadFactor3x = -1.00
self.loadFactor3y = -1.00
self.loadFactor3z = -1.00
self.loadFactor4a = -1.00
self.loadFactor4b = -1.00
self.loadFactor4c = -1.00
self.loadFactor4d = -1.00
self.loadFactor4e = -1.00
self.loadFactor4f = -1.00
self.loadFactor4g = -1.00
self.loadFactor4h = -1.00
self.loadFactor4i = -1.00
self.loadFactor4j = -1.00
self.loadFactor4k = -1.00
self.loadFactor4l = -1.00
self.loadFactor4m = -1.00
self.loadFactor4n = -1.00
self.loadFactor4o = -1.00
self.loadFactor4p = -1.00
self.loadFactor4q = -1.00
self.loadFactor4r = -1.00
self.loadFactor4s = -1.00
self.loadFactor4t = -1.00
self.loadFactor4u = -1.00
self.loadFactor4v = -1.00
self.loadFactor4w = -1.00
self.loadFactor4x = -1.00
self.loadFactor4y = -1.00
self.loadFactor4z = -1.00

print self.loadPositionDescriptions

if self.loadPositionDescriptions == ['Varma_SPtc-2'] or self.loadFactor1a==34:
    self.L = 5638.80/1000.0
    self.bamb = 762.00/1000.0
    self.bottom = 19.10/1000.0
    self.topt = 19.10/1000.0
    self.plateFyBottom = 419.20
    self.plateFyTop = 419.20
    self.plateFulBottom = 469.20
    self.plateFulTop = 469.20
    self.concretef1c = 42.60
    self.studh1h = 152.40/1000.0
    self.studh1Dia = 12.70/1000.0
    self.studh1FullHeight = False
    self.studh1Resistance = 42.97*1000.0
    self.studh1fy = 374.00
    self.studh1fu = 424.00
    self.studh2h = 381.00/1000.0
    self.studh2Dia = 15.46/1000.0
    self.studh2FullHeight = True
    self.studh2Resistance = 70.01*1000.0
    self.studh2fy = 416.40
    self.studh2fu = 466.40
    self.studh1h = 127.00/1000.0
    self.studh1Dia = 12.70/1000.0
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studFinalHeight = False
self.stud1Resistance = 42.97+1000.0
self.stud1fy = 374.00
self.stud1fu = 424.00

self.stud2h = 0.08/1000.0
self.stud2Diam = 0.09/1000.0
self.stud2FinalHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 762.00/1000.0
self.support1a = 4876.80/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.loadDefPoints = []
self.loadFactor1g = -self.loadFactor1f = self.loadFactor1e = self.loadFactor1d = self.loadFactor1c = self.loadFactor1b = self.loadFactor1a = 3352.80/1000.0
self.load1g = -self.load1f = -self.load1e = self.load1d = -self.load1c = self.load1b = self.load1a = 2286.00/1000.0
self.support1c = self.support1a = 4876.80/1000.0
self.support1 = 762.00/1000.0
self.appliedDeflection = 0.05
self.CyclicLoad = 50.00
self.CyclicLoad = 50.00*studt2fyy = 1.00
self.studt2Resistance = 0.00
self.studt2Dia = 0.00/1000.0
self.studt2h = 0.00/1000.0
self.studt1fyy = 424.00
self.studt1fyy = 374.00
self.studt1Resistance = 42.97
self.studt1Dia = 0.00/1000.0
self.studt1h = 0.00/1000.0

print self.Case_Name
if self.Case_Name == 'Varma_SP2c\-3' or self.beamCaseNo==35:

self.L = 7162.80/1000.0
self.beamnb = 762.00/1000.0
self.beamnb = 381.00/1000.0
self.bottomt = 19.10/1000.0

433
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.topt = 19.1/1000.0
self.plateFyBottom = 419.20
self.plateFyTop = 419.20
self.plateFuBottom = 469.20
self.plateFuTop = 469.20
self.concretefc = 35.70

self.studb1h = 152.40/1000.0
self.studb1Dia = 12.70/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 42.97*1000.0
self.studb1fy = 374.00
self.studb1fu = 424.00

self.studb2h = 381.00/1000.0
self.studb2Dia = 15.46/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 70.01*1000.0
self.studb2fy = 416.40
self.studb2fu = 466.40

self.studt1h = 127.00/1000.0
self.studt1Dia = 12.70/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 42.97*1000.0
self.studt1fy = 374.00
self.studt1fu = 424.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 762.00/1000.0
self.support1a = 6400.80/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 3048.00/1000.0
self.load1a = 4114.80/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((3581.4/1000,0.0,0.0,)))

self.studPositionDescriptions.extend(((0.0889,'b1',3,(3)),(0.2159,'b1',3,(3)),(0.3429,'b1',3,(3)),(0.4699,'b1',3,(3)),(0.5969,'b1',3,(3)),(0.7239,'b1',3,(3)),(0.8509,'b1',3,(3)),(0.9779,'b1',3,(3)),(1.1049,'b1',3,(3)),
(1.2319,'b1',3,(3)),(1.3589,'b1',3,(3)),(1.4859,'b1',3,(3)),(1.6129,'b1',3,(3)),(1.7399,'b1',3,(3)),(1.8669,'b1',3,(3)),(1.9939,'b1',3,(3)),(2.1209,'b1',3,(3)),(2.2479,'b1',3,(3)),(2.3749,'b1',3,(3)),(2.5019,'b1',3,(3)),
(2.6289,'b1',3,(3)),(2.7559,'b1',3,(3)),(2.8829,'b1',3,(3)),(3.0099,'b1',3,(3)),(3.1369,'b1',3,(3)),(3.2639,'b1',3,(3)),(3.3909,'b1',3,(3)),(3.5179,'b1',3,(3)),(3.6449,'b1',3,(3)),(3.7719,'b1',3,(3)),(3.8989,'b1',3,(3)))
```

APPENDIX B: Test Database in Python code format

```python
self.positionDescriptions.extend(((0.3429, 'b2', 3, (3)), (0.7239, 'b2', 3, (3)), (1.1049, 'b2', 3, (3)), (1.4859, 'b2', 3, (3)), (1.8669, 'b2', 3, (3)), (2.2479, 'b2', 3, (3)), (2.6289, 'b2', 3, (3)), (3.0099, 'b2', 3, (3)), (3.3909, 'b2', 3, (3)), (3.7719, 'b2', 3, (3)), (4.1529, 'b2', 3, (3)), (4.5339, 'b2', 3, (3)), (4.9149, 'b2', 3, (3)), (5.2959, 'b2', 3, (3)), (5.6769, 'b2', 3, (3)), (6.0579, 'b2', 3, (3)), (6.4389, 'b2', 3, (3)), (6.8199, 'b2', 3, (3)), ))
self.positionDescriptions.extend(((0.0889, 't1', 3, (3)), (0.2159, 't1', 3, (3)), (0.3429, 't1', 3, (3)), (0.4699, 't1', 3, (3)), (0.5969, 't1', 3, (3)), (0.7239, 't1', 3, (3)), (0.8509, 't1', 3, (3)), (0.9779, 't1', 3, (3)), (1.0499, 't1', 3, (3)), (1.2319, 't1', 3, (3)), (1.3589, 't1', 3, (3)), (1.4859, 't1', 3, (3)), (1.6129, 't1', 3, (3)), (1.7399, 't1', 3, (3)), (1.8669, 't1', 3, (3)), (2.0099, 't1', 3, (3)), (2.1209, 't1', 3, (3)), (2.2479, 't1', 3, (3)), (2.3749, 't1', 3, (3)), (2.5019, 't1', 3, (3)), (2.6289, 't1', 3, (3)), (2.7559, 't1', 3, (3)), (2.8829, 't1', 3, (3)), (3.0099, 't1', 3, (3)), (3.1369, 't1', 3, (3)), (3.2639, 't1', 3, (3)), (3.3909, 't1', 3, (3)), (3.5179, 't1', 3, (3)), (3.6449, 't1', 3, (3)), (3.7719, 't1', 3, (3)), (3.8989, 't1', 3, (3)), (4.0259, 't1', 3, (3)), (4.1529, 't1', 3, (3)), (4.2799, 't1', 3, (3)), (4.4069, 't1', 3, (3)), (4.5339, 't1', 3, (3)), (4.6609, 't1', 3, (3)), (4.7749, 't1', 3, (3)), (4.9149, 't1', 3, (3)), (5.0419, 't1', 3, (3)), (5.1689, 't1', 3, (3)), (5.2959, 't1', 3, (3)), (5.4229, 't1', 3, (3)), (5.5499, 't1', 3, (3)), (5.6769, 't1', 3, (3)), (5.8039, 't1', 3, (3)), (5.9309, 't1', 3, (3)), (6.0579, 't1', 3, (3)), (6.1849, 't1', 3, (3)), (6.3119, 't1', 3, (3)), (6.4389, 't1', 3, (3)), (6.5659, 't1', 3, (3)), (6.6929, 't1', 3, (3)), (6.8199, 't1', 3, (3)), (6.9469, 't1', 3, (3)), (7.0739, 't1', 3, (3)), )
print self.Case_Name
if self.Case_Name == 'Varma_SP2c-4' or self.beamCaseNo==36:
    self.L = 7162.80/1000.0
    self.bamb = 762.00/1000.0
    self.bam = 381.00/1000.0
    self.bottom = 19.10/1000.0
    self.top = 39.10/1000.0
    self.plateFyBottom = 419.20
    self.plateFyTop = 419.20
    self.plateFuBottom = 469.20
    self.plateFuTop = 469.20
    self.concretec = 36.20
    self.studb1h = 152.40/1000.0
    self.studb1Dia = 12.70/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 42.97+1000.0
    self.studb1fY = 374.00
    self.studb1fu = 424.00
    self.studb2h = 381.00/1000.0
    self.studb2Dia = 15.46/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 70.01+1000.0
    self.studb2fy = 416.40
    self.studb2fu = 466.40
    self.studt1h = 127.00/1000.0
    self.studt1Dia = 12.70/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 42.97+1000.0
    self.studt1fy = 374.00
    self.studt1fu = 424.00
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00+1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05
    self.supportL = 762.00/1000.0
    self.supportF1 = 6400.80/1000.0
    self.supportF2 = -100.00/1000.0
```

435
A

s

self.load1 = 3049.00/1000.0
self.load2a = 4114.80/1000.0
self.load2b = -100.00/1000.0
self.load3a = -100.00/1000.0
self.load4a = -100.00/1000.0
self.load5a = -100.00/1000.0
self.load6a = -100.00/1000.0
self.load7a = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor2 = -100.0
self.loadFactor3 = -100.0
self.loadFactor4 = -100.0
self.loadFactor5 = -100.0
self.loadDefPoints = []
self.loadDefPoints = ((3581.4/1000.0, 0.0))

self.studPositionDescriptions = extend(((0.0889, 'b1', 3, 3), (0.2159, 'b1', 3, 3), (0.3429, 'b1', 3, 3), (0.4699, 'b1', 3, 3), (0.5969, 'b1', 3, 3), (0.7239, 'b1', 3, 3), (0.8509, 'b1', 3, 3), (0.9779, 'b1', 3, 3), (1.1049, 'b1', 3, 3), (1.2319, 'b1', 3, 3), (1.3589, 'b1', 3, 3), (1.4859, 'b1', 3, 3), (1.6129, 'b1', 3, 3), (1.7399, 'b1', 3, 3), (1.8669, 'b1', 3, 3), (1.9939, 'b1', 3, 3), (2.1209, 'b1', 3, 3), (2.2479, 'b1', 3, 3), (2.3749, 'b1', 3, 3), (2.5019, 'b1', 3, 3), (2.6289, 'b1', 3, 3), (2.7559, 'b1', 3, 3), (2.8829, 'b1', 3, 3), (3.0099, 'b1', 3, 3), (3.1369, 'b1', 3, 3), (3.2639, 'b1', 3, 3), (3.3909, 'b1', 3, 3), (3.5179, 'b1', 3, 3), (3.6449, 'b1', 3, 3), (3.7719, 'b1', 3, 3), (3.8989, 'b1', 3, 3), (4.0259, 'b1', 3, 3), (4.1529, 'b1', 3, 3), (4.2799, 'b1', 3, 3), (4.4069, 'b1', 3, 3), (4.5339, 'b1', 3, 3), (4.6609, 'b1', 3, 3), (4.7879, 'b1', 3, 3), (4.9149, 'b1', 3, 3), (5.0419, 'b1', 3, 3), (5.1689, 'b1', 3, 3), (5.2959, 'b1', 3, 3), (5.4229, 'b1', 3, 3), (5.5499, 'b1', 3, 3), (5.6769, 'b1', 3, 3), (5.8039, 'b1', 3, 3), (5.9309, 'b1', 3, 3), (6.0579, 'b1', 3, 3), (6.1849, 'b1', 3, 3), (6.3119, 'b1', 3, 3), (6.4389, 'b1', 3, 3), (6.5659, 'b1', 3, 3), (6.6929, 'b1', 3, 3), (6.8199, 'b1', 3, 3), (6.9469, 'b1', 3, 3), (7.0739, 'b1', 3, 3), (7.1969, 'b1', 3, 3),


print self.Case_Name
if self.Case_Name == 'Varma_SP2c--5' or self.beamCaseNo==37:

self.L = 6592.60/1000.0
self.beamb = 1219.00/1000.0
self.beamb = 609.50/1000.0
self.beamb = 12.70/1000.0
self.beamb = 12.70/1000.0

self.plateFyBottom = 439.90
self.plateFyTop = 439.90
self.plateFyBottom = 489.90
self.plateFyTop = 489.90
self.concrete = 37.00

self.studb1h = 152.40/1000.0
self.studb1Dha = 12.00/1000.0
self.studb1Dha = 12.00/1000.0
self.studb1Dha = True
self.studb1Dha = False
self.studb1Resistance = 42.97×1000.0
self.studb1Dha = 374.00
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb1fu = 424.00
self.studb2h = 609.50/1000.0
self.studb2Dia = 24.79/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 180.07*1000.0
self.studb2fy = 416.40
self.studb2fu = 466.40
self.stud1h = 203.17/1000.0
self.stud1Dia = 12.70/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 42.97*1000.0
self.stud1fy = 374.00
self.stud1fu = 424.00

self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00*1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 609.50/1000.0
self.support1a = 5973.10/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.support1d = -100.00/1000.0
self.support1e = -100.00/1000.0
self.support1f = -100.00/1000.0
self.support1g = -100.00/1000.0
self.support1h = -100.00/1000.0
self.support1i = -100.00/1000.0
self.support1j = -100.00/1000.0

self.loadDefPoints = []
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadFactor1h = -100.00

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadFactor1h = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((3291.3,1000.0,0.0),))

self.studPositionDescriptions.extend(((0.1650666666666667, 'b1', 3, (3)), (0.3682333333333334, 'b1', 3, (3)), (0.5714000000000001, 'b1', 3, (3)), (0.7745666666666668, 'b1', 3, (3)), (0.9777333333333335, 'b1', 3, (3)),
(1.1809000000000002, 'b1', 3, (3)), (1.3840666666666669, 'b1', 3, (3)), (1.5872333333333336, 'b1', 3, (3)),
(1.7940400000000003, 'b1', 3, (3)), (1.9935666666666667, 'b1', 3, (3)), (2.1967333333333337, 'b1', 3, (3)),
(2.3999000000000004, 'b1', 3, (3)), (2.6030666666666671, 'b1', 3, (3)), (2.8062333333333338, 'b1', 3, (3)),
(3.0094000000000005, 'b1', 3, (3)), (3.2125666666666672, 'b1', 3, (3)), (3.4157333333333339, 'b1', 3, (3)),
(3.618900000000006, 'b1', 3, (3)), (3.8220666666666673, 'b1', 3, (3)), (4.0252333333333334, 'b1', 3, (3)),
(4.2284000000000007, 'b1', 3, (3)), (4.4315666666666674, 'b1', 3, (3)), (4.6347333333333341, 'b1', 3, (3)),
(4.8379900000000008, 'b1', 3, (3)), (5.0410666666666675, 'b1', 3, (3)), (5.2442333333333342, 'b1', 3, (3)),
(5.4474000000000009, 'b1', 3, (3)), (5.6505666666666676, 'b1', 3, (3)), (5.8537333333333343, 'b1', 3, (3)),
(6.0569000000000001, 'b1', 3, (3)), (6.2600666666666677, 'b1', 3, (3)), (6.4632333333333344, 'b1', 3, (3)),
self.studPositionDescriptions.extend(((0.5714, 'b2', 3, (3)), (1.1809, 'b2', 3, (3)), (1.7904, 'b2', 3, (3)), (2.3999, 'b2', 3, (3)), (3.0094, 'b2', 3, (3)), (3.6189, 'b2', 3, (3)), (4.2284, 'b2', 3, (3)), (4.8379, 'b2', 3, (3)), (5.4474, 'b2', 3, (3)),
(6.0569, 'b2', 3, (3)),

self.studPositionDescriptions.extend(((0.1650666666666667, 't1', 3, (3)), (0.3682333333333334, 't1', 3, (3)),
(0.5714000000000001, 't1', 3, (3)), (0.7745666666666668, 't1', 3, (3)), (0.9777333333333335, 't1', 3, (3)),
(1.1809000000000002, 't1', 3, (3)), (1.3840666666666669, 't1', 3, (3)), (1.5872333333333336, 't1', 3, (3)),
(1.7904000000000003, 't1', 3, (3)), (1.9935666666666667, 't1', 3, (3)), (2.1967333333333337, 't1', 3, (3)),
(2.3999000000000004, 't1', 3, (3)), (2.6030666666666671, 't1', 3, (3)), (2.8062333333333338, 't1', 3, (3)),
```

437
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```
print self.Case_Name
if self.Case_Name == 'Varma_SP2c-6' or self.beamCaseNo==38:
    self.L = 6582.60/1000.0
    self.beamb = 1219.00/1000.0
    self.bottomt = 12.70/1000.0
    self.topt = 12.70/1000.0

    self.plateFyBottom = 439.90
    self.plateFyTop = 439.90
    self.plateFuBottom = 439.90
    self.plateFuTop = 439.90
    self.concretefc = 37.00

    self.studb1h = 152.40/1000.0
    self.studb1Dia = 12.70/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 42.97*1000.0
    self.studb1fy = 374.00
    self.studb1fu = 424.00

    self.studb2h = 609.50/1000.0
    self.studb2Dia = 24.79/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 180.07*1000.0
    self.studb2fy = 416.40
    self.studb2fu = 466.40

    self.studt1h = 203.17/1000.0
    self.studt1Dia = 12.70/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 42.97*1000.0
    self.studt1fy = 374.00
    self.studt1fu = 424.00

    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00*1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50

    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05

    self.support1 = 609.50/1000.0
    self.support1a = 5973.10/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0

    self.load1 = 3047.50/1000.0
    self.load1a = 3535.10/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0

    self.loadFactor1 = 1.00
    self.loadFactor1a = 1.00
    self.loadFactor1b = -100.00
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1x = -100.0
self.loadFactorId = -100.0
self.loadFactor1y = -100.0
self.loadFactor1z = -100.0
self.loadDefPoints = []
self.loadFactor1g = self.loadFactor1f = self.loadFactor1e = self.loadFactor1d = False

self.loadDefPoints.extend(((3291.3/1000.0, 0.0)))
self.loadDefPoints.extend(((5.44740000009, 't1', 3, (3)), (5.650566666667, 'b1', 3, (3)), (5.85373333334, 'b1', 3, (3))))
self.loadDefPoints.extend(((6.05690000001, 'b1', 3, (3)), (6.643233333344, 'b1', 3, (3))))
self.loadDefPoints.extend(((6.05690000001, 'b1', 3, (3)), (6.643233333344, 'b1', 3, (3))))
self.loadDefPoints.extend(((6.05690000001, 'b1', 3, (3)), (6.643233333344, 'b1', 3, (3))))

print self.Case_Name
if self.Case_Name == 'Matsumoto1' or self.beamCaseNo==46:
    L = 2650.0/1000.0
    beamb = 225.0/1000.0
    beamb = 600.0/1000.0
    bottomt = 4.5/1000.0
    topt = 4.5/1000.0
    plateFyBottom = 343.20
    plateFyTop = 343.20
    plateFullBottom = 393.20
    plateFullTop = 393.20
    concrete = 35.90
    studbh = 72.00/1000.0
    studbd1Dia = 9.00/1000.0
    studb1FullHeight = False
    studb1Resistance = 22.90*1000.0
    studb1 fy = 400.00
    studb1 fu = 450.00
    studbd2h = 0.00/1000.0
    studbd2 Dia = 0.00/1000.0
    studbd2FullHeight = False
    studbd2 Resistance = 0.00*1000.0
    studbd2 fy = 1.00
    studbd2 fu = 1.50

    self.studth = 72.00/1000.0
    self.studt1Dia = 9.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 22.90*1000.0
    self.studt1 fy = 400.00
    self.studt1fu = 450.00
```

439
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 650.00/1000.0
self.support1a = 2000.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 200.00/1000.0
self.load1a = 1100.00/1000.0
self.load1b = 1500.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = 2.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1325/10000.0, 0.0),))

self.studyPositionDescriptions.extend(((0.18, 'b1', 4, (4)), (0.315, 'b1', 4, (4)), (0.45, 'b1', 4, (4)), (0.585, 'b1', 4, (4)), (0.72, 'b1', 4, (4)), (0.855, 'b1', 4, (4)), (0.99, 'b1', 4, (4)), (1.125, 'b1', 4, (4)), (1.26, 'b1', 4, (4)), (1.395, 'b1', 4, (4)), (1.53, 'b1', 4, (4)), (1.645, 'b1', 4, (4)), (1.78, 'b1', 4, (4)), (1.915, 'b1', 4, (4)), (2.07, 'b1', 4, (4)), (2.205, 'b1', 4, (4)), (2.34, 'b1', 4, (4)), (2.475, 'b1', 4, (4)))))

self.studyPositionDescriptions.extend(((0.18, 't1', 4, (4)), (0.315, 't1', 4, (4)), (0.45, 't1', 4, (4)), (0.585, 't1', 4, (4)), (0.72, 't1', 4, (4)), (0.855, 't1', 4, (4)), (0.99, 't1', 4, (4)), (1.125, 't1', 4, (4)), (1.26, 't1', 4, (4)), (1.395, 't1', 4, (4)), (1.53, 't1', 4, (4)), (1.665, 't1', 4, (4)), (1.8, 't1', 4, (4)), (1.935, 't1', 4, (4)), (2.07, 't1', 4, (4)), (2.205, 't1', 4, (4)), (2.34, 't1', 4, (4)), (2.475, 't1', 4, (4)))))

print self.Case_Name
if self.Case_Name == 'Matsumoto3' or self.beamCaseNo==47:

self.L = 2655.00/1000.0
self.beamh = 450.00/1000.0
self.beamb = 600.00/1000.0
self.bottomt = 4.50/1000.0
self.top = 4.50/1000.0

self.plateFyBottom = 343.20
self.plateFyTop = 343.20
self.plateFuBottom = 393.20
self.plateFuTop = 393.20
self.concretefc = 35.70

self.studb1h = 72.00/1000.0
self.studb1Dia = 9.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 22.90+1000.0
self.studb1fy = 400.00
self.studb1fu = 450.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
```

440
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2FullHeight = False
self.studb2Resistance = 0.00 * 1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studth = 72.00/1000.0
self.stud1Dia = 9.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 22.90*1000.0
self.stud1fy = 400.00
self.stud1fu = 450.00

self.stud2h = 0.08/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 650.00/1000.0
self.support1a = 2000.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 200.00/1000.0
self.load1a = 1100.00/1000.0
self.load1b = 1550.00/1000.0
self.load1c = 2450.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = 2.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1327.5/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.18, 'b1',4,(4)),(0.315, 'b1',4,(4)),(0.45, 'b1',4,(4)),(0.585, 'b1',4,(4)) ,
(0.72, 'b1',4,(4)),(0.855, 'b1',4,(4)),(0.99, 'b1',4,(4)),(1.125, 'b1',4,(4)),(1.26, 'b1',4,(4)),(1.395, 'b1',4,(4)) ,
(1.53, 'b1',4,(4)),(1.665, 'b1',4,(4)),(1.8, 'b1',4,(4)),(1.935, 'b1',4,(4)),(2.07, 'b1',4,(4)),(2.205, 'b1',4,(4)) ,
(2.34, 'b1',4,(4)),(2.475, 'b1',4,(4))))
self.studPositionDescriptions.extend(((0.18, 't1',4,(4)),(0.315, 't1',4,(4)),(0.45, 't1',4,(4)),(0.585, 't1',4,(4)) ,
(0.72, 't1',4,(4)),(0.855, 't1',4,(4)),(0.99, 't1',4,(4)),(1.125, 't1',4,(4)),(1.26, 't1',4,(4)),(1.395, 't1',4,(4)) ,
(1.53, 't1',4,(4)),(1.665, 't1',4,(4)),(1.8, 't1',4,(4)),(1.935, 't1',4,(4)),(2.07, 't1',4,(4)),(2.205, 't1',4,(4)) ,
(2.34, 't1',4,(4)),(2.475, 't1',4,(4))))

print self.Case_Name
if self.Case_Name == 'Matsumoto4' or self.beamCaseNo==48:

self.L = 2655.00/1000.0
self.beambh = 450.00/1000.0
self.beamb = 600.00/1000.0
self.bottomt = 4.50/1000.0
self.topt = 4.50/1000.0

self.plateFyBottom = 375.60
self.plateFyTop = 375.60
self.plateFuBottom = 425.60
self.plateFuTop = 425.60
```

441
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

self.concretef = 36.10
self.studb1h = 72.00/1000.0
self.studb1Dia = 9.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 22.90/1000.0
self.studb1fy = 400.00
self.studb1fu = 450.00
self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00/1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50
self.studt1h = 72.00/1000.0
self.studt1Dia = 9.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 22.90/1000.0
self.studt1fy = 400.00
self.studt1fu = 450.00
self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00/1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50
self.CyclicLoad = 50.00/1000.0
self.appliedDeflection = 0.05
self.support1 = 650.00/1000.0
self.support1a = 2000.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 200.00/1000.0
self.load1a = 1100.00/1000.0
self.load1b = 1550.00/1000.0
self.load1c = 245.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = 2.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((1327.5/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.18, 'b1', 8, (8)), (0.2475, 'b1', 8, (8)), (0.315, 'b1', 8, (8)), (0.3825, 'b1', 8, (8)), (0.45, 'b1', 8, (8)), (0.5175, 'b1', 8, (8)), (0.585, 'b1', 8, (8)), (0.6525, 'b1', 8, (8)), (0.72, 'b1', 8, (8)), (0.7875, 'b1', 8, (8)), (0.855, 'b1', 8, (8)), (0.9225, 'b1', 8, (8)), (0.99, 'b1', 8, (8)), (1.0575, 'b1', 8, (8)), (1.125, 'b1', 8, (8)), (1.1925, 'b1', 8, (8)), (1.26, 'b1', 8, (8)), (1.3275, 'b1', 8, (8)), (1.395, 'b1', 8, (8)), (1.4625, 'b1', 8, (8)), (1.53, 'b1', 8, (8)), (1.5975, 'b1', 8, (8)), (1.665, 'b1', 8, (8)), (1.7325, 'b1', 8, (8)), (1.8, 'b1', 8, (8)), (1.8675, 'b1', 8, (8)), (1.935, 'b1', 8, (8)), (2.0025, 'b1', 8, (8)), (2.07, 'b1', 8, (8)), (2.1375, 'b1', 8, (8)), (2.205, 'b1', 8, (8)), (2.2725, 'b1', 8, (8)), (2.34, 'b1', 8, (8)), (2.4075, 'b1', 8, (8)), (2.475, 'b1', 8, (8)),))
self.studPositionDescriptions.extend(((0.18, 't1', 8, (8)), (0.2475, 't1', 8, (8)), (0.315, 't1', 8, (8)), (0.3825, 't1', 8, (8)), (0.45, 't1', 8, (8)), (0.5175, 't1', 8, (8)), (0.585, 't1', 8, (8)), (0.6525, 't1', 8, (8)), (0.72, 't1', 8, (8)), (0.7875, 't1', 8, (8)), (0.855, 't1', 8, (8)), (0.9225, 't1', 8, (8)), (0.99, 't1', 8, (8)), (1.0575, 't1', 8, (8)), (1.125, 't1', 8, (8)), (1.1925, 't1', 8, (8)), (1.26, 't1', 8, (8)), (1.3275, 't1', 8, (8)), (1.395, 't1', 8, (8)), (1.4625, 't1', 8, (8)), (1.53, 't1', 8, (8)), (1.5975, 't1', 8, (8)), (1.665, 't1', 8, (8)), (1.7325, 't1', 8, (8)), (1.8, 't1', 8, (8)), (1.8675, 't1', 8, (8)), (1.935, 't1', 8, (8)), (2.0025, 't1', 8, (8)), (2.07, 't1', 8, (8)), (2.1375, 't1', 8, (8)), (2.205, 't1', 8, (8)), (2.2725, 't1', 8, (8)), (2.34, 't1', 8, (8)), (2.4075, 't1', 8, (8)), (2.475, 't1', 8, (8)),))
print self.Case_Name
if self.Case_Name == 'Matsumoto5' or self.beamCaseNo==49:

self.L = 4905.00/1000.0
self.beambh = 450.00/1000.0
self.beamb = 600.00/1000.0
self.bottomt = 4.50/1000.0
self.topt = 4.50/1000.0

self.plateFyBottom = 343.20
self.plateFyTop = 343.20
self.plateFuBottom = 393.20
self.plateFuTop = 393.20
self.concretefc = 35.90

self.studb1h = 72.00/1000.0
self.studb1Dia = 9.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 22.90+1000.0
self.studb1fy = 400.00
self.studb1fu = 450.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00+1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 72.00/1000.0
self.studt1Dia = 9.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 22.90+1000.0
self.studt1fy = 400.00
self.studt1fu = 450.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00+1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 1100.00/1000.0
self.support1a = 3800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 200.00/1000.0
self.load1a = 2000.00/1000.0
self.load1b = 2900.00/1000.0
self.load1c = 4700.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = 2.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
APPENDIX B: TEST DATABASE IN PYTHON Code Format

```python
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((2452.5/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.22,'b1','4,4),(0.355,'b1','4,4),(0.49,'b1','4,4),(0.625,'b1','4,4)
,(0.76,'b1','4,4),(0.895,'b1','4,4),(1.03,'b1','4,4),(1.165,'b1','4,4),(1.3,'b1','4,4),(1.435,'b1','4,4)
,(1.57,'b1','4,4),(1.705,'b1','4,4),(1.84,'b1','4,4),(1.975,'b1','4,4),(2.11,'b1','4,4),(2.245,'b1','4,4)
,(2.38,'b1','4,4),(2.515,'b1','4,4),(2.65,'b1','4,4),(2.785,'b1','4,4),(2.92,'b1','4,4),(3.055,'b1'
,'4,4),(3.19,'b1','4,4),(3.325,'b1','4,4),(3.46,'b1','4,4),(3.595,'b1','4,4),(3.73,'b1','4,4),(3.865,'b1'
,'4,4),(4.135,'b1','4,4),(4.27,'b1','4,4),(4.405,'b1','4,4),(4.54,'b1','4,4),(4.675,'b1'
,'4,4),(4.81,4,4)))

self.studPositionDescriptions.extend(((0.22,'t1','4,4),(0.355,'t1','4,4),(0.49,'t1','4,4),(0.625,'t1','4,4)
,(0.76,'t1','4,4),(0.895,'t1','4,4),(1.03,'t1','4,4),(1.165,'t1','4,4),(1.3,'t1','4,4),(1.435,'t1','4,4)
,(1.57,'t1','4,4),(1.705,'t1','4,4),(1.84,'t1','4,4),(1.975,'t1','4,4),(2.11,'t1','4,4),(2.245,'t1','4,4)
,(2.38,'t1','4,4),(2.515,'t1','4,4),(2.65,'t1','4,4),(2.785,'t1','4,4),(2.92,'t1','4,4),(3.055,'t1'
,'4,4),(3.19,'t1','4,4),(3.325,'t1','4,4),(3.46,'t1','4,4),(3.595,'t1','4,4),(3.73,'t1','4,4),(3.865,'t1'
,'4,4),(4.135,'t1','4,4),(4.27,'t1','4,4),(4.405,'t1','4,4),(4.54,'t1','4,4),(4.675,'t1'
,'4,4),(4.81,4,4)))

print self.Case_Name
if self.Case_Name == 'Matsumoto8' or self.beamCaseNo==50:

self.L = 6420.00/1000.0
self.beamb = 600.00/1000.0
self.beamh = 600.00/1000.0
self.bottomt = 12.00/1000.0
self.top = 12.00/1000.0

self.plateFyBottom = 343.20
self.plateFyTop = 343.20
self.plateFuBottom = 393.20
self.plateFuTop = 393.20
self.concretefc = 37.20

self.studb1h = 128.00/1000.0
self.studb1Dia = 16.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 72.38*1000.0
self.studb1fy = 400.00
self.studb1fu = 450.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 128.00/1000.0
self.studt1Dia = 16.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 72.38*1000.0
self.studt1fy = 400.00
self.studt1fu = 450.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 1400.00/1000.0
self.support1a = 5000.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
```

444
Appendix B: Test Database in Python Code Format

```python
self.load1 = 200.00/1000.0
self.load1a = 2600.00/1000.0
self.load1b = 3800.00/1000.0
self.load1c = 6200.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = 100.00/1000.0

self.loadDefPoints = []
self.loadDefPoints.extend(((3210/1000.0, 0.0),))

self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = 2.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00/1000.0
self.loadFactor1e = -100.00/1000.0
self.loadFactor1f = -100.00/1000.0
self.loadFactor1g = -100.00/1000.0

print self.Case_Name
if self.Case_Name == 'Matsumoto9' or self.beamCaseNo==51:
    self.L = 6420.00/1000.0
    self.beamh = 600.00/1000.0
    self.beamb = 500.00/1000.0
    self.bottomt = 12.00/1000.0
    self.topt = 12.00/1000.0

self.plateFyBottom = 399.10
self.plateFyTop = 399.10
self.plateFuBottom = 449.10
self.plateFuTop = 449.10
self.concretefc = 39.20

self.studh1b = 128.00/1000.0
self.studDia1 = 16.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 72.38*1000.0
self.stud1fy = 400.00
self.stud1fu = 450.00

self.studh2b = 300.00/1000.0
self.stud2Dia = 16.00/1000.0
self.stud2FullHeight = True
self.stud2Resistance = 59.47*1000.0
self.stud2fy = 319.70
self.stud2fu = 369.70

self.stud1h = 128.00/1000.0
self.stud1Dia = 16.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 72.38*1000.0
self.stud1fy = 400.00
self.stud1fu = 450.00
self.stud2h = 0.00/1000.0
```

APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2f = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 1400.00/1000.0
self.support1a = 5000.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 200.00/1000.0
self.load1a = 2600.00/1000.0
self.load1b = 3800.00/1000.0
self.load1c = 6200.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = 2.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((3210/1000.0, 0.0),))
self.loadPositionDescriptions.extend(((0.22, 'b1', 4,(4)), (0.47, 'b1', 4,(4)), (0.72, 'b1', 4,(4)), (0.97, 'b1', 4,(4)), (1.22, 'b1', 4,(4)), (1.47, 'b1', 4,(4)), (1.72, 'b1', 4,(4)), (1.97, 'b1', 4,(4)), (2.22, 'b1', 4,(4)), (2.47, 'b1', 4,(4)), (2.72, 'b1', 4,(4)), (2.97, 'b1', 4,(4)), (3.22, 'b1', 4,(4)), (3.47, 'b1', 4,(4)), (3.72, 'b1', 4,(4)), (3.97, 'b1', 4,(4)), (4.22, 'b1', 4,(4)), (4.47, 'b1', 4,(4)), (4.72, 'b1', 4,(4)), (4.97, 'b1', 4,(4)), (5.22, 'b1', 4,(4)), (5.47, 'b1', 4,(4)), (5.72, 'b1', 4,(4)), (6.02, 'b1', 4,(4)),))
self.loadPositionDescriptions.extend(((0.22, 'b2', 2,(2)), (0.47, 'b2', 2,(2)), (0.72, 'b2', 2,(2)), (0.97, 'b2', 2,(2)), (1.22, 'b2', 2,(2)), (1.47, 'b2', 2,(2)), (1.72, 'b2', 2,(2)), (1.97, 'b2', 2,(2)), (2.22, 'b2', 2,(2)), (2.47, 'b2', 2,(2)), (2.72, 'b2', 2,(2)), (2.97, 'b2', 2,(2)), (3.22, 'b2', 2,(2)), (3.47, 'b2', 2,(2)), (3.72, 'b2', 2,(2)), (3.97, 'b2', 2,(2)), (4.22, 'b2', 2,(2)), (4.47, 'b2', 2,(2)), (4.72, 'b2', 2,(2)), (4.97, 'b2', 2,(2)), (5.22, 'b2', 2,(2)), (5.47, 'b2', 2,(2)), (5.72, 'b2', 2,(2)), (6.02, 'b2', 2,(2))))
self.loadPositionDescriptions.extend(((0.22, 't1', 4,(4)), (0.47, 't1', 4,(4)), (0.72, 't1', 4,(4)), (0.97, 't1', 4,(4)), (1.22, 't1', 4,(4)), (1.47, 't1', 4,(4)), (1.72, 't1', 4,(4)), (1.97, 't1', 4,(4)), (2.22, 't1', 4,(4)), (2.47, 't1', 4,(4)), (2.72, 't1', 4,(4)), (2.97, 't1', 4,(4)), (3.22, 't1', 4,(4)), (3.47, 't1', 4,(4)), (3.72, 't1', 4,(4)), (3.97, 't1', 4,(4)), (4.22, 't1', 4,(4)), (4.47, 't1', 4,(4)), (4.72, 't1', 4,(4)), (4.97, 't1', 4,(4)), (5.22, 't1', 4,(4)), (5.47, 't1', 4,(4)), (5.72, 't1', 4,(4)), (6.02, 't1', 4,(4))))

print self.Case_Name
if self.Case_Name == "Matsumoto10" or self.beamCaseNo==52:

self.L = 6420.00/1000.0
self.beambm = 600.00/1000.0
self.beambmb = 500.00/1000.0
self.bottomt = 12.00/1000.0
self.topt = 12.00/1000.0

self.plateFyBottom = 390.30
self.plateFyTop = 390.30
self.plateFubottom = 440.30
self.plateFutop = 440.30
self.concrete = 38.10

self.stud1 = 128.00/1000.0
self.stud1Dia = 16.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 72.38+1000.0
```

446
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb1fy = 400.00
self.studb1fu = 450.00
self.studb2h = 300.00/1000.0
self.studb2Dia = 22.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 112.12*1000.0
self.studb2fy = 318.70
self.studb2fu = 368.70
self.studt1h = 128.00/1000.0
self.studt1Dia = 16.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 72.38*1000.0
self.studt1fy = 400.00
self.studt1fu = 450.00
self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50
self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05
self.support1 = 1400.00/1000.0
self.support1a = 5000.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 200.00/1000.0
self.load1a = 2600.00/1000.0
self.load1b = 3800.00/1000.0
self.load1c = 6200.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = 2.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((3210/10000.0,0.0),))
self.loadPositionDescriptions.extend(((0.22,'b1',4,4),(0.47,'b1',4,4),(0.72,'b1',4,4),(0.97,'b1',4,4))
(1.22,'b1',4,4),(1.47,'b1',4,4),(1.72,'b1',4,4),(1.97,'b1',4,4))
(2.22,'b1',4,4),(2.47,'b1',4,4)
(3.72,'b1',4,4),(3.97,'b1',4,4))
(4.42,'b1',4,4),(4.67,'b1',4,4),(4.92,'b1',4,4),(5.17,'b1',4,4)
(5.22,'b1',4,4),(5.47,'b1',4,4)
(5.72,'b1',4,4),(5.97,'b1',4,4))
self.loadPositionDescriptions.extend(((0.22,'b2',2,2),(0.47,'b2',2,2),(0.72,'b2',2,2),(0.97,'b2',2,2)
(1.22,'b2',2,2),(1.47,'b2',2,2),(1.72,'b2',2,2),(1.97,'b2',2,2)
(2.22,'b2',2,2),(2.47,'b2',2,2)
(3.22,'b2',2,2),(3.47,'b2',2,2),(3.72,'b2',2,2),(3.97,'b2',2,2)
(4.22,'b2',2,2),(4.47,'b2',2,2),(4.72,'b2',2,2),(5.22,'b2',2,2)
(5.72,'b2',2,2),(5.97,'b2',2,2),))
self.loadPositionDescriptions.extend(((0.22,'t1',4,4),(0.47,'t1',4,4),(0.72,'t1',4,4),(0.97,'t1',4,4)
(1.22,'t1',4,4),(1.47,'t1',4,4),(1.72,'t1',4,4),(1.97,'t1',4,4)
(2.22,'t1',4,4),(2.47,'t1',4,4)
(3.72,'t1',4,4),(3.97,'t1',4,4))
(4.42,'t1',4,4),(4.67,'t1',4,4),(4.92,'t1',4,4),(5.17,'t1',4,4)
(5.22,'t1',4,4),(5.47,'t1',4,4)
(5.72,'t1',4,4),(5.97,'t1',4,4))
print self.Case_Name
```

447
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
if self.Case_Name == 'MatsumotoS1' or self.beamCaseNo==53:
    self.L = 3200.00/1000.0
    self.beamb = 500.00/1000.0
    self.beamb = 750.00/1000.0
    self.beammt = 9.00/1000.0
    self.topt = 9.00/1000.0
    self.plateFyBottom = 377.60
    self.plateFyTop = 377.60
    self.plateFuBottom = 427.60
    self.plateFuTop = 427.60
    self.concretefc = 32.70
    self.studb1h = 128.00/1000.0
    self.studb1Dia = 16.00/1000.0
    self.studb1Resistance = 72.38+1000.0
    self.studb1fy = 400.00
    self.studb1fu = 450.00
    self.studb2h = 0.00/1000.0
    self.studb2Dia = 0.00/1000.0
    self.studb2Resistance = 0.00+1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50
    self.studt1h = 128.00/1000.0
    self.studt1Dia = 16.00/1000.0
    self.studt1Resistance = 72.38+1000.0
    self.studt1fy = 400.00
    self.studt1fu = 450.00
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2Resistance = 0.00+1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05
    self.support1 = 800.00/1000.0
    self.support1a = 2800.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 300.00/1000.0
    self.load1a = 2300.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = 2.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
    self.loadFactor1d = -100.00
    self.loadFactor1e = -100.00
    self.loadFactor1f = -100.00
    self.loadFactor1g = -100.00
    self.loadDefPoints = []
    self.loadDefPoints.extend(((1600/1000,0.0),))
```

448
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studPositionDescriptions.extend(((0.3, 'b1', 3, (3)), (0.55, 'b1', 3, (3)), (0.8, 'b1', 3, (3)), (1.05, 'b1', 3, (3)), (1.3, 'b1', 3, (3)), (1.55, 'b1', 3, (3)), (1.8, 'b1', 3, (3)), (2.05, 'b1', 3, (3)), (2.3, 'b1', 3, (3)), (2.55, 'b1', 3, (3)), (2.8, 'b1', 3, (3)),))

self.studPositionDescriptions.extend(((0.3, 't1', 3, (3)), (0.55, 't1', 3, (3)), (0.8, 't1', 3, (3)), (1.05, 't1', 3, (3)), (1.3, 't1', 3, (3)), (1.55, 't1', 3, (3)), (1.8, 't1', 3, (3)), (2.05, 't1', 3, (3)), (2.3, 't1', 3, (3)), (2.55, 't1', 3, (3)), (2.8, 't1', 3, (3)),))

print(self.Case_Name)

if self.Case_Name == 'MatsumotoS2' or self.beamCaseNo==54:
    self.L = 5600.00/1000.0
    self.beamb = 500.00/1000.0
    self.beamb = 750.00/1000.0
    self.bottomt = 9.00/1000.0
    self.topt = 9.00/1000.0
    self.plateFyBottom = 391.30
    self.plateFyTop = 391.30
    self.plateFuBottom = 441.30
    self.plateFuTop = 441.30
    self.concretefc = 32.50
    self.studb1h = 128.00/1000.0
    self.studb1Dia = 16.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 72.38*1000.0
    self.studb1fy = 400.00
    self.studb1fu = 450.00
    self.studb2h = 0.00/1000.0
    self.studb2Dia = 0.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 0.00*1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50
    self.studt1h = 128.00/1000.0
    self.studt1Dia = 16.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 72.38*1000.0
    self.studt1fy = 400.00
    self.studt1fu = 450.00
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00*1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05
    self.support1 = 1300.00/1000.0
    self.support1a = 5300.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 300.00/1000.0
    self.load1a = 4300.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = 2.00
    self.loadFactor1b = -100.00
```

449
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((2800/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.3, 'b1', 3, (3)), (0.55, 'b1', 3, (3)), (0.8, 'b1', 3, (3)), (1.05, 'b1', 3, (3)), (1.3, 'b1', 3, (3)), (1.55, 'b1', 3, (3)), (1.8, 'b1', 3, (3)), (2.05, 'b1', 3, (3)), (2.3, 'b1', 3, (3)), (2.55, 'b1', 3, (3)),
(2.8, 'b1', 3, (3)), (3.05, 'b1', 3, (3)), (3.3, 'b1', 3, (3)), (3.55, 'b1', 3, (3)), (3.8, 'b1', 3, (3)), (4.05, 'b1', 3, (3)), (4.3, 'b1', 3, (3)), (4.55, 'b1', 3, (3)), (4.8, 'b1', 3, (3)), (5.05, 'b1', 3, (3)), (5.3, 'b1', 3, (3)),))
self.studPositionDescriptions.extend(((0.3, 't1', 3, (3)), (0.55, 't1', 3, (3)), (0.8, 't1', 3, (3)), (1.05, 't1', 3, (3)), (1.3, 't1', 3, (3)), (1.55, 't1', 3, (3)), (1.8, 't1', 3, (3)), (2.05, 't1', 3, (3)), (2.3, 't1', 3, (3)), (2.55, 't1', 3, (3)), (2.8, 't1', 3, (3)), (3.05, 't1', 3, (3)), (3.3, 't1', 3, (3)), (3.55, 't1', 3, (3)), (3.8, 't1', 3, (3)), (4.05, 't1', 3, (3)), (4.3, 't1', 3, (3)), (4.55, 't1', 3, (3)), (4.8, 't1', 3, (3)), (5.05, 't1', 3, (3)), (5.3, 't1', 3, (3)),))

print self.Case_Name
if self.Case_Name == 'MatsumotoS3' or self.beamCaseNo==55:

    self.L = 3200.00/1000.0
    self.beamb = 500.00/1000.0
    self.beamb = 750.00/1000.0
    self.bottomt = 9.00/1000.0
    self.topt = 9.00/1000.0
    self.plateFyBottom = 387.40
    self.plateFyTop = 387.40
    self.plateFyBottom = 387.40
    self.plateFuBottom = 437.40
    self.plateFuTop = 437.40
    self.concretefc = 35.40
    self.studb1h = 128.00/1000.0
    self.studb1Dia = 16.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 72.38*1000.0
    self.studb1fy = 400.00
    self.studb1fu = 450.00
    self.studb2h = 0.00/1000.0
    self.studb2Dia = 0.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 0.00*1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50
    self.studt1h = 128.00/1000.0
    self.studt1Dia = 16.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 72.38*1000.0
    self.studt1fy = 400.00
    self.studt1fu = 450.00
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00*1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05
    self.support1 = 800.00/1000.0
    self.support1a = 2800.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 300.00/1000.0
```

450
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.load1a = 2300.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend([(1600/1000.0,0.0),])

selfstudPositionDescriptions.extend(((0.3,'b1',4,(4)),(0.55,'b1',4,(4)),(0.8,'b1',4,(4)),(1.05,'b1',4,(4)),(1.3,'b1',4,(4)),(1.55,'b1',4,(4)),(2.05,'b1',4,(4)),(2.3,'b1',4,(4)),(2.55,'b1',4,(4)),(2.8,'b1',4,(4)),))

selfstudPositionDescriptions.extend(((0.3,'t1',4,(4)),(0.55,'t1',4,(4)),(0.8,'t1',4,(4)),(1.05,'t1',4,(4)),(1.3,'t1',4,(4)),(1.55,'t1',4,(4)),(2.05,'t1',4,(4)),(2.3,'t1',4,(4)),(2.55,'t1',4,(4)),(2.8,'t1',4,(4)),))

if self.Case_Name == 'MatsumotoS4' or self.beamCaseNo==56:
    self.L = 5600.00/1000.0
    self.beamh = 500.00/1000.0
    self.beamb = 750.00/1000.0
    self.bottomt = 9.00/1000.0
    self.top = 9.00/1000.0
    self.plateFyBottom = 362.80
    self.plateFyTop = 362.80
    self.plateFuBottom = 412.80
    self.plateFuTop = 412.80
    self.concretefc = 32.80

self.studh1 = 128.00/1000.0
self.studDia1 = 16.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 72.38*1000.0
self.studbfy = 400.00
self.studbfu = 450.00

self.studh2 = 0.00/1000.0
self.studDia2 = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00*1000.0
self.studbfy = 1.00
self.studbfu = 1.50

self.stud1h = 128.00/1000.0
self.stud1Dia = 16.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 72.38*1000.0
self.stud1fy = 400.00
self.stud1fu = 450.00

self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00*1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50
```

451
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05
self.support1 = 1300.00/1000.0
self.support1a = 5300.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 300.00/1000.0
self.load1a = 4300.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((2800/1000.0, 0.0), ))
self.loadDefPoints.extend(((0.22, 'b1', 4,(4)), (0.47, 'b1', 4,(4)), (0.72, 'b1', 4,(4)), (0.97, 'b1', 4,(4)), (1.22, 'b1', 4,(4)), (1.47, 'b1', 4,(4)), (1.72, 'b1', 4,(4)), (1.97, 'b1', 4,(4)), (2.22, 'b1', 4,(4)), (2.47, 'b1', 4,(4)), (2.72, 'b1', 4,(4)), (2.97, 'b1', 4,(4)), (3.22, 'b1', 4,(4)), (3.47, 'b1', 4,(4)), (3.72, 'b1', 4,(4)), (3.97, 'b1', 4,(4)), (4.22, 'b1', 4,(4)), (4.47, 'b1', 4,(4)), (4.72, 'b1', 4,(4)), (4.97, 'b1', 4,(4)), (5.22, 'b1', 4,(4)), (5.47, 'b1', 4,(4)), )
self.loadPositionDescriptions.extend(((0.22, 't1', 4,(4)), (0.47, 't1', 4,(4)), (0.72, 't1', 4,(4)), (0.97, 't1', 4,(4)), (1.22, 't1', 4,(4)), (1.47, 't1', 4,(4)), (1.72, 't1', 4,(4)), (1.97, 't1', 4,(4)), (2.22, 't1', 4,(4)), (2.47, 't1', 4,(4)), (2.72, 't1', 4,(4)), (2.97, 't1', 4,(4)), (3.22, 't1', 4,(4)), (3.47, 't1', 4,(4)), (3.72, 't1', 4,(4)), (3.97, 't1', 4,(4)), (4.22, 't1', 4,(4)), (4.47, 't1', 4,(4)), (4.72, 't1', 4,(4)), (4.97, 't1', 4,(4)), (5.22, 't1', 4,(4)), (5.47, 't1', 4,(4)), )

print self.Case_Name
if self.Case_Name == 'MatsumotoS5' or self.beamCaseNo==57:

self.L = 3200.00/1000.0
self.beamh = 500.00/1000.0
self.beamb = 750.00/1000.0
self.bottomt = 9.00/1000.0
self.topt = 9.00/1000.0
self.plateFyBottom = 376.60
self.plateFyTop = 376.60
self.plateFuBottom = 426.60
self.plateFuTop = 426.60
self.concretef = 36.30
self.studb1h = 128.00/1000.0
self.studb1Dia = 16.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 72.38*1000.0
self.studb1fy = 400.00
self.studb1fu = 450.00
self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50
self.studt1h = 128.00/1000.0
```

452
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

self.stud1Dia = 16.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 72.38+1000.0
self.stud1fy = 400.00
self.stud1fu = 450.00

self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 800.00/1000.0
self.support1a = 2800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 300.00/1000.0
self.load1a = 2300.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []

if self.Case_Name == 'MatsumotoS6' or self.beamCaseNo==58:
    self.L = 5600.00/1000.0
    self.beamh = 500.00/1000.0
    self.beamb = 750.00/1000.0
    self.bottomt = 9.00/1000.0
    self.top = 9.00/1000.0
    self.plateFyBottom = 384.40
    self.plateFyTop = 384.40
    self.plateFuBottom = 434.40
    self.plateFuTop = 454.40
    self.concretefc = 32.80

    self.studbh = 128.00/1000.0
    self.studbDia = 16.00/1000.0
    self.studbFullHeight = False
    self.studbResistance = 72.38+1000.0
    self.studbf = 400.00
    self.studbfu = 450.00
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00+1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50
self.studt1h = 128.00/1000.0
self.studt1Dia = 16.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 72.38+1000.0
self.studt1fy = 400.00
self.studt1fu = 450.00
self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05
self.support1 = 1300.00/1000.0
self.support1a = 5300.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 300.00/1000.0
self.load1a = 4300.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((2800/1000.0,0.0),))
self.setPositionDescriptions.extend(((0.22,'b1',4,(4)),(0.47,'b1',4,(4))),(0.72,'b1',4,(4))),(0.97,'b1',4,(4))
(1.22,'b1',4,(4))),(1.47,'b1',4,(4))),(1.72,'b1',4,(4))),(1.97,'b1',4,(4))
(2.22,'b1',4,(4))),(2.47,'b1',4,(4))),(2.72,'b1',4,(4))),(2.97,'b1',4,(4))
(3.22,'b1',4,(4))),(3.47,'b1',4,(4))),(3.72,'b1',4,(4))),(3.97,'b1',4,(4))
(4.22,'b1',4,(4))),(4.47,'b1',4,(4))),(4.72,'b1',4,(4))),(4.97,'b1',4,(4))
(5.22,'b1',4,(4))),(5.47,'b1',4,(4))
)
self.setPositionDescriptions.extend(((0.22,'t1',4,(4))),(0.47,'t1',4,(4))),(0.72,'t1',4,(4))),(0.97,'t1',4,(4))
(1.22,'t1',4,(4))),(1.47,'t1',4,(4))),(1.72,'t1',4,(4))),(1.97,'t1',4,(4))
(2.22,'t1',4,(4))),(2.47,'t1',4,(4))),(2.72,'t1',4,(4))),(2.97,'t1',4,(4))
(3.22,'t1',4,(4))),(3.47,'t1',4,(4))),(3.72,'t1',4,(4))),(3.97,'t1',4,(4))
(4.22,'t1',4,(4))),(4.47,'t1',4,(4))),(4.72,'t1',4,(4))),(4.97,'t1',4,(4))
(5.22,'t1',4,(4))),(5.47,'t1',4,(4))
)

if self.Case_Name == 'TP0069_1' or self.beamCaseNo==59:
    self.L = 8400.00/1000.0
    self.beamb = 500.00/1000.0
    self.beamb = 800.00/1000.0
    self.bottomt = 9.00/1000.0
    self.topt = 9.00/1000.0
```

APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.plateFyBottom = 303.00
self.plateFyTop = 303.00
self.plateFuBottom = 428.00
self.plateFuTop = 428.00
self.concretefc = 30.00

self.studb1h = 103.00/1000.0
self.studb1Dia = 16.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 71.12*1000.0
self.studb1fy = 400.00
self.studb1fu = 450.00

self.studb2h = 250.00/1000.0
self.studb2Dia = 16.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 66.11*1000.0
self.studb2fy = 361.00
self.studb2fu = 411.00

self.studt1h = 103.00/1000.0
self.studt1Dia = 16.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 71.12*1000.0
self.studt1fy = 400.00
self.studt1fu = 450.00

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 1800.00/1000.0
self.support1a = 6400.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 200.00/1000.0
self.load1a = 3400.00/1000.0
self.load1b = 5000.00/1000.0
self.load1c = 8200.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 2.00
self.loadFactor1b = 2.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((4200/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.1,'b1',4),(0.3,'b1',4),(0.5,'b1',4),(0.7,'b1',4),(0.9,'b1',4),(1.1,'b1',4),(1.3,'b1',4),(1.5,'b1',4),(1.7,'b1',4),(1.9,'b1',4),(2.1,'b1',4),(2.3,'b1',4),(2.5,'b1',4),(2.7,'b1',4),(2.9,'b1',4),(3.1,'b1',4),(3.3,'b1',4),(3.5,'b1',4),(3.7,'b1',4),(3.9,'b1',4),(4.1,'b1',4),(4.3,'b1',4),(4.4,'b1',4),(4.5,'b1',4),(4.7,'b1',4),(4.9,'b1',4),(5.1,'b1',4),(5.3,'b1',4),(5.5,'b1',4),(5.7,'b1',4),(5.9,'b1',4),(6.1,'b1',4),(6.3,'b1',4),(6.5,'b1',4),(6.7,'b1',4),(6.9,'b1',4),(7.1,'b1',4),(7.3,'b1',4),(7.5,'b1',4),(7.7,'b1',4),(7.9,'b1',4),(8.1,'b1',4),(8.3,'b1',4),(8.5,'b1',4),(8.7,'b1',4),(8.9,'b1',4),(9.1,'b1',4),(9.3,'b1',4),(9.5,'b1',4),(9.7,'b1',4),(9.9,'b1',4)))
```

APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studPositionDescriptions.extend(((0.4, 'b2', 3, (3)), (0.8, 'b2', 3, (3)), (1.2, 'b2', 3, (3)), (1.6, 'b2', 3, (3)), (2., 'b2', 3, (3)), (2.4, 'b2', 3, (3)), (2.8, 'b2', 3, (3)), (3.2, 'b2', 3, (3)), (3.6, 'b2', 3, (3)), (4., 'b2', 3, (3)), (4.4, 'b2', 3, (3)), (4.8, 'b2', 3, (3)), (5.2, 'b2', 3, (3)), (5.6, 'b2', 3, (3)), (6., 'b2', 3, (3)), (6.4, 'b2', 3, (3)), (6.8, 'b2', 3, (3)), (7.2, 'b2', 3, (3)), (7.6, 'b2', 3, (3)), (8., 'b2', 3, (3))))

self.studPositionDescriptions.extend(((0.1, 't1', 4, (4)), (0.3, 't1', 4, (4)), (0.5, 't1', 4, (4)), (0.7, 't1', 4, (4)), (0.9, 't1', 4, (4)), (1.1, 't1', 4, (4)), (1.3, 't1', 4, (4)), (1.5, 't1', 4, (4)), (1.7, 't1', 4, (4)), (1.9, 't1', 4, (4)), (2.1, 't1', 4, (4)), (2.3, 't1', 4, (4)), (2.5, 't1', 4, (4)), (2.7, 't1', 4, (4)), (2.9, 't1', 4, (4)), (3.1, 't1', 4, (4)), (3.3, 't1', 4, (4)), (3.5, 't1', 4, (4)), (3.7, 't1', 4, (4)), (3.9, 't1', 4, (4)), (4.1, 't1', 4, (4)), (4.3, 't1', 4, (4)), (4.5, 't1', 4, (4)), (4.7, 't1', 4, (4)), (4.9, 't1', 4, (4)), (5.1, 't1', 4, (4)), (5.3, 't1', 4, (4)), (5.5, 't1', 4, (4)), (5.7, 't1', 4, (4)), (5.9, 't1', 4, (4)), (6.1, 't1', 4, (4)), (6.3, 't1', 4, (4)), (6.5, 't1', 4, (4)), (6.7, 't1', 4, (4)), (6.9, 't1', 4, (4)), (7.1, 't1', 4, (4)), (7.3, 't1', 4, (4)), (7.5, 't1', 4, (4)), (7.7, 't1', 4, (4)), (7.9, 't1', 4, (4)), (8.1, 't1', 4, (4)), (8.3, 't1', 4, (4))))

print self.Case_Name

if self.Case_Name == 'TP0069_2' or self.beamCaseNo==60:

self.L = 8400.00/1000.0
self.beamb = 500.00/1000.0
self.bottomt = 9.00/1000.0
self.top t = 9.00/1000.0

self.plateFyBottom = 303.00
self.plateFyTop = 303.00
self.plateFuBottom = 428.00
self.plateFuTop = 428.00
self.concretefc = 30.00

self.studb1h = 103.00/1000.0
self.studb1Dia = 16.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 71.12*1000.0
self.studb1fy = 400.00
self.studb1fu = 450.00

self.studb2h = 250.00/1000.0
self.studb2Dia = 16.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 66.11*1000.0
self.studb2fy = 361.00
self.studb2fu = 411.00

self.studt1h = 103.00/1000.0
self.studt1Dia = 16.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 71.12*1000.0
self.studt1fy = 400.00
self.studt1fu = 450.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 1800.00/1000.0
self.support1a = 6800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 200.00/1000.0
self.load1a = 3400.00/1000.0
self.load1b = 5000.00/1000.0
self.load1c = 8200.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
```

456
if self.Case_Name == 'TP0069_5' or self.beamCaseNo==63:

self.loadDefPoints = []
self.loadDefPoints.extend(((4200/1000,0,0),))

self.studyPositionDescriptions.extend(((0.1, 't1', 4), (0.3, 't1', 4), (0.5, 't1', 4), (0.7, 't1', 4), (0.9, 't1', 4), (1.1, 'b1', 4), (1.3, 'b1', 4), (1.5, 'b1', 4), (1.7, 'b1', 4), (1.9, 'b1', 4), (2.1, 'b1', 4), (2.3, 'b1', 4), (2.5, 'b1', 4), (2.7, 'b1', 4), (2.9, 'b1', 4), (3.1, 'b1', 4), (3.3, 'b1', 4), (3.5, 'b1', 4), (3.7, 'b1', 4), (3.9, 'b1', 4), (4.1, 'b1', 4), (4.3, 'b1', 4), (4.5, 'b1', 4), (4.7, 'b1', 4), (4.9, 'b1', 4), (5.1, 'b1', 4), (5.3, 'b1', 4), (5.5, 'b1', 4), (5.7, 'b1', 4), (5.9, 'b1', 4), (6.1, 'b1', 4), (6.3, 'b1', 4), (6.5, 'b1', 4), (6.7, 'b1', 4), (6.9, 'b1', 4), (7.1, 'b1', 4), (7.3, 'b1', 4), (7.5, 'b1', 4), (7.7, 'b1', 4), (7.9, 'b1', 4), (8.1, 'b1', 4), (8.3, 'b1', 4), (4), (4.2, 'b2', 4), (4.4), (4.6, 'b2', 4), (4.8, 'b2', 4), (5), (5.2, 'b2', 4), (5.4, 'b2', 4), (5.6, 'b2', 4), (5.8, 'b2', 4), (6, 'b2', 4), (6.2, 'b2', 4), (6.4, 'b2', 4), (6.6, 'b2', 4), (6.8, 'b2', 4), (7, 'b2', 4), (7.6, 'b2', 4), (8, 'b2', 4), (4),))

self.studyPositionDescriptions.extend(((0.1, 't1', 4), (0.3, 't1', 4), (0.5, 't1', 4), (0.7, 't1', 4), (0.9, 't1', 4), (1.1, 't1', 4), (1.3, 't1', 4), (1.5, 't1', 4), (1.7, 't1', 4), (1.9, 't1', 4), (2.1, 't1', 4), (2.3, 't1', 4), (2.5, 't1', 4), (2.7, 't1', 4), (2.9, 't1', 4), (3.1, 't1', 4), (3.3, 't1', 4), (3.5, 't1', 4), (3.7, 't1', 4), (3.9, 't1', 4), (4.1, 't1', 4), (4.3, 't1', 4), (4.5, 't1', 4), (4.7, 't1', 4), (4.9, 't1', 4), (5.1, 't1', 4), (5.3, 't1', 4), (5.5, 't1', 4), (5.7, 't1', 4), (5.9, 't1', 4), (6.1, 't1', 4), (6.3, 't1', 4), (6.5, 't1', 4), (6.7, 't1', 4), (6.9, 't1', 4), (7.1, 't1', 4), (7.3, 't1', 4), (7.5, 't1', 4), (7.7, 't1', 4), (7.9, 't1', 4), (8.1, 't1', 4), (8.3, 't1', 4), (4), (4.2, 'b1', 4), (4.4), (4.6, 'b1', 4), (4.8, 'b1', 4), (5), (5.2, 'b1', 4), (5.4, 'b1', 4), (5.6, 'b1', 4), (5.8, 'b1', 4), (6, 'b1', 4), (6.2, 'b1', 4), (6.4, 'b1', 4), (6.6, 'b1', 4), (6.8, 'b1', 4), (7, 'b1', 4), (7.6, 'b1', 4), (8, 'b1', 4), (4),))

print self.Case_Name

if self.Case_Name == 'TP0069_5' or self.beamCaseNo==63:

self.L = 8400.00/1000.0
self.beam = 500.00/1000.0
self.beamb = 800.00/1000.0
self.bottomt = 6.00/1000.0
self.top = 6.00/1000.0

self.plateFyBottom = 303.00
self.plateFyTop = 303.00
self.plateFuBottom = 428.00
self.plateFuTop = 428.00
self.concretefc = 38.00

self.studh1 = 103.00/1000.0
self.studhDia = 13.00/1000.0
self.studhFullHeight = False
self.studhResistance = 47.78*1000.0
self.studhf1y = 400.00
self.studhf1u = 450.00

self.studh2h = 250.00/1000.0
self.studh2Dia = 16.00/1000.0
self.studh2FullHeight = True
self.studh2Resistance = 66.11*1000.0
self.studhf2y = 361.00
self.studhf2u = 411.00
self.studh1 = 103.00/1000.0
self.studh1Dia = 13.00/1000.0
self.studh1FullHeight = False
self.studh1Resistance = 47.78*1000.0
self.studhf1y = 400.00
self.studhf1u = 450.00
self.studh2h = 0.00/1000.0

457
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2Dia = 0.00/1000.0
self.stud2FullPath = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 600.00/1000.0
self.support1a = 7800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 2400.00/1000.0
self.load1a = 6000.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((2400/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.1, 'b1', 4, (4)), (0.3, 'b1', 4, (4)), (0.5, 'b1', 4, (4)), (0.7, 'b1', 4, (4)), (0.9, 'b1', 4, (4)), (1.1, 'b1', 4, (4)), (1.3, 'b1', 4, (4)), (1.5, 'b1', 4, (4)), (1.7, 'b1', 4, (4)), (1.9, 'b1', 4, (4)), (2.1, 'b1', 4, (4)), (2.3, 'b1', 4, (4)), (2.5, 'b1', 4, (4)), (2.7, 'b1', 4, (4)), (2.9, 'b1', 4, (4)), (3.1, 'b1', 4, (4)), (3.3, 'b1', 4, (4)), (3.5, 'b1', 4, (4)), (3.7, 'b1', 4, (4)), (3.9, 'b1', 4, (4)), (4.1, 'b1', 4, (4)), (4.3, 'b1', 4, (4)), (4.5, 'b1', 4, (4)), (4.7, 'b1', 4, (4)), (4.9, 'b1', 4, (4)), (5.1, 'b1', 4, (4)), (5.3, 'b1', 4, (4)), (5.5, 'b1', 4, (4)), (5.7, 'b1', 4, (4)), (5.9, 'b1', 4, (4)), (6.1, 'b1', 4, (4)), (6.3, 'b1', 4, (4)), (6.5, 'b1', 4, (4)), (6.7, 'b1', 4, (4)), (6.9, 'b1', 4, (4)), (7.1, 'b1', 4, (4)), (7.3, 'b1', 4, (4)), (7.5, 'b1', 4, (4)), (7.7, 'b1', 4, (4)), (7.9, 'b1', 4, (4)), (8.1, 'b1', 4, (4)), (8.3, 'b1', 4, (4))))

self.studPositionDescriptions.extend(((0.4, 'b2', 2, (2)), (0.8, 'b2', 2, (2)), (1.2, 'b2', 2, (2)), (1.6, 'b2', 2, (2)), (2.0, 'b2', 2, (2)), (2.4, 'b2', 2, (2)), (2.8, 'b2', 2, (2)), (3.2, 'b2', 2, (2)), (3.6, 'b2', 2, (2)), (4.0, 'b2', 2, (2)), (4.4, 'b2', 2, (2)), (4.8, 'b2', 2, (2)), (5.2, 'b2', 2, (2)), (5.6, 'b2', 2, (2)), (6.0, 'b2', 2, (2)), (6.4, 'b2', 2, (2)), (6.8, 'b2', 2, (2)), (7.2, 'b2', 2, (2)), (7.6, 'b2', 2, (2)), (8.0, 'b2', 2, (2))))

self.studPositionDescriptions.extend(((0.1, 't1', 4, (4)), (0.3, 't1', 4, (4)), (0.5, 't1', 4, (4)), (0.7, 't1', 4, (4)), (0.9, 't1', 4, (4)), (1.1, 't1', 4, (4)), (1.3, 't1', 4, (4)), (1.5, 't1', 4, (4)), (1.7, 't1', 4, (4)), (1.9, 't1', 4, (4)), (2.1, 't1', 4, (4)), (2.3, 't1', 4, (4)), (2.5, 't1', 4, (4)), (2.7, 't1', 4, (4)), (2.9, 't1', 4, (4)), (3.1, 't1', 4, (4)), (3.3, 't1', 4, (4)), (3.5, 't1', 4, (4)), (3.7, 't1', 4, (4)), (3.9, 't1', 4, (4)), (4.1, 't1', 4, (4)), (4.3, 't1', 4, (4)), (4.5, 't1', 4, (4)), (4.7, 't1', 4, (4)), (4.9, 't1', 4, (4)), (5.1, 't1', 4, (4)), (5.3, 't1', 4, (4)), (5.5, 't1', 4, (4)), (5.7, 't1', 4, (4)), (5.9, 't1', 4, (4)), (6.1, 't1', 4, (4)), (6.3, 't1', 4, (4)), (6.5, 't1', 4, (4)), (6.7, 't1', 4, (4)), (6.9, 't1', 4, (4)), (7.1, 't1', 4, (4)), (7.3, 't1', 4, (4)), (7.5, 't1', 4, (4)), (7.7, 't1', 4, (4)), (7.9, 't1', 4, (4)), (8.1, 't1', 4, (4)), (8.3, 't1', 4, (4))))

print(self.Case_Name)
if self.Case_Name == 'TP0069_6' or self.beamCaseNo==64:

self.L = 8400.00/1000.0
self.beamL = 300.00/1000.0
self.beamB = 800.00/1000.0
self.beambot = 6.00/1000.0
self.beamTop = 6.00/1000.0
self.beamFyb = 303.00
self.beamFyTop = 303.00
self.beamFyBottom = 428.00
self.beamFubottom = 38.00
self.studBottom = 103.00/1000.0
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studDi = 13.00/1000.0
self.studFullHeight = False
self.studResistance = 47.78*1000.0
self.stud1y = 400.0
self.stud1fu = 450.0

self.stud2h = 250.00/1000.0
self.stud2Di = 16.00/1000.0
self.stud2FullHeight = True
self.stud2Resistance = 66.11*1000.0
self.stud2fy = 361.00
self.stud2fu = 411.00

self.stud1h = 103.00/1000.0
self.stud1Di = 13.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 47.78*1000.0
self.stud1fy = 400.0
self.stud1fu = 450.0

self.stud2h = 0.00/1000.0
self.stud2Di = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00*1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CycleLoad = 50.00/1000.0
self.appliedDeflection = 0.05

self.support1 = 600.00/1000.0
self.support1a = 600.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 2400.00/1000.0
self.load1a = 7800.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 2.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend((2400/1000.0, 0.0))

self.studPositionDescriptions.extend(((0.1, 't1', 4, (4)), (0.3, 't1', 4, (4)), (0.5, 't1', 4, (4)), (0.7, 't1', 4, (4)), (0.9, 't1', 4, (4)), (1.1, 't1', 4, (4)), (1.3, 't1', 4, (4)), (1.5, 't1', 4, (4)), (1.7, 't1', 4, (4)), (1.9, 't1', 4, (4)), (2.1, 't1', 4, (4)), (2.3, 't1', 4, (4)), (2.5, 't1', 4, (4)), (2.7, 't1', 4, (4)), (2.9, 't1', 4, (4)), (3.1, 't1', 4, (4)), (3.3, 't1', 4, (4)), (3.5, 't1', 4, (4)), (3.7, 't1', 4, (4)), (3.9, 't1', 4, (4)), (4.1, 't1', 4, (4)), (4.3, 't1', 4, (4)), (4.5, 't1', 4, (4)), (4.7, 't1', 4, (4)), (4.9, 't1', 4, (4)), (5.1, 't1', 4, (4)), (5.3, 't1', 4, (4)), (5.5, 't1', 4, (4)), (5.7, 't1', 4, (4)), (5.9, 't1', 4, (4)), (6.1, 't1', 4, (4)), (6.3, 't1', 4, (4)), (6.5, 't1', 4, (4)), (6.7, 't1', 4, (4)), (6.9, 't1', 4, (4)), (7.1, 't1', 4, (4)), (7.3, 't1', 4, (4)), (7.5, 't1', 4, (4)), (7.7, 't1', 4, (4)), (7.9, 't1', 4, (4)), (8.1, 't1', 4, (4)), (8.3, 't1', 4, (4)).))

self.studPositionDescriptions.extend(((0.4, 'b2', 2, (2)), (0.8, 'b2', 2, (2)), (1.2, 'b2', 2, (2)), (1.6, 'b2', 2, (2)), (2.0, 'b2', 2, (2)), (2.4, 'b2', 2, (2)), (2.8, 'b2', 2, (2)), (3.2, 'b2', 2, (2)), (3.6, 'b2', 2, (2)), (4.0, 'b2', 2, (2)), (4.4, 'b2', 2, (2)), (4.8, 'b2', 2, (2)), (5.2, 'b2', 2, (2)), (5.6, 'b2', 2, (2)), (6.0, 'b2', 2, (2)), (6.4, 'b2', 2, (2)), (6.8, 'b2', 2, (2)), (7.2, 'b2', 2, (2)), (7.6, 'b2', 2, (2)), (8.0, 'b2', 2, (2)).))

self.studPositionDescriptions.extend(((0.1, 't1', 4, (4)), (0.3, 't1', 4, (4)), (0.5, 't1', 4, (4)), (0.7, 't1', 4, (4)), (0.9, 't1', 4, (4)), (1.1, 't1', 4, (4)), (1.3, 't1', 4, (4)), (1.5, 't1', 4, (4)), (1.7, 't1', 4, (4)), (1.9, 't1', 4, (4)), (2.1, 't1', 4, (4)), (2.3, 't1', 4, (4)), (2.5, 't1', 4, (4)), (2.7, 't1', 4, (4)), (2.9, 't1', 4, (4)), (3.1, 't1', 4, (4)), (3.3, 't1', 4, (4)), (3.5, 't1', 4, (4)), (3.7, 't1', 4, (4)), (3.9, 't1', 4, (4)), (4.1, 't1', 4, (4)), (4.3, 't1', 4, (4)), (4.5, 't1', 4, (4)), (4.7, 't1', 4, (4)), (4.9, 't1', 4, (4)).))
```

459
## Appendix B: Test Database in Python Code Format

```python
print(self.Case_Name)

if self.Case_Name == 'Chu_WS3.0' or self.beamCaseNo==71:

    self.L = 7815.00/1000.0
    self.beamh = 734.00/1000.0
    self.beamb = 762.00/1000.0
    self.bottomt = 14.00/1000.0
    self.top t = 14.00/1000.0
    self.plateFyBottom = 295.00
    self.plateFyTop = 295.00
    self.plateFuBottom = 450.00
    self.plateFuTop = 450.00
    self.concretefc = 34.50
    self.studb1h = 150.00/1000.0
    self.studb1Dia = 19.04/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 95.21*1000.0
    self.studb1fy = 368.00
    self.studb1fu = 418.00
    self.studb2h = 0.00/1000.0
    self.studb2Dia = 0.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 0.00*1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50
    self.studt1h = 150.00/1000.0
    self.studt1Dia = 19.04/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 95.21*1000.0
    self.studt1fy = 368.00
    self.studt1fu = 418.00
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 26.07/1000.0
    self.studt2FullHeight = True
    self.studt2Resistance = 115.31*1000.0
    self.studt2fy = 220.00
    self.studt2fu = 270.00
    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05
    self.support1 = 625.00/1000.0
    self.support1a = 7190.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 2425.00/1000.0
    self.load1a = 5390.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = 1.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
    self.loadFactor1d = -100.00
    self.loadFactor1e = -100.00
```

460
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1g = -100.00
self.loadFactor2g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((3907.5/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.11, 'b1', 4,(4)), (0.2626, 'b1', 4,(4)), (0.4152, 'b1', 4,(4)), (0.5678, 'b1', 4,(4)), (0.7204, 'b1', 4,(4)), (0.873, 'b1', 4,(4)), (1.0256, 'b1', 4,(4)), (1.1782, 'b1', 4,(4)), (1.3308, 'b1', 4,(4)), (1.4834, 'b1', 4,(4)), (1.636, 'b1', 4,(4)), (1.7866, 'b1', 4,(4)), (1.9412, 'b1', 4,(4)), (2.0938, 'b1', 4,(4)), (2.2464, 'b1', 4,(4)), (2.399, 'b1', 4,(4)), (2.5516, 'b1', 4,(4)), (2.7042, 'b1', 4,(4)), (2.8568, 'b1', 4,(4)), (3.0094, 'b1', 4,(4)), (3.162, 'b1', 4,(4)), (3.3146, 'b1', 4,(4)), (3.4672, 'b1', 4,(4)), (3.6198, 'b1', 4,(4)), (3.7724, 'b1', 4,(4)), (3.925, 'b1', 4,(4)), (4.0776, 'b1', 4,(4)), (4.2302, 'b1', 4,(4)), (4.3828, 'b1', 4,(4)), (4.5354, 'b1', 4,(4)), (4.688, 'b1', 4,(4)), (4.8406, 'b1', 4,(4)), (4.9932, 'b1', 4,(4)), (5.1458, 'b1', 4,(4)), (5.2994, 'b1', 4,(4)), (5.451, 'b1', 4,(4)), (5.6036, 'b1', 4,(4)), (5.7562, 'b1', 4,(4)), (5.9088, 'b1', 4,(4)), (6.0614, 'b1', 4,(4)), (6.214, 'b1', 4,(4)), (6.3666, 'b1', 4,(4)), (6.5192, 'b1', 4,(4)), (6.6718, 'b1', 4,(4)), (6.8244, 'b1', 4,(4)), (6.9771, 'b1', 4,(4)), (7.1296, 'b1', 4,(4)), (7.2822, 'b1', 4,(4)), (7.4348, 'b1', 4,(4)), (7.5874, 'b1', 4,(4)), (7.74, 'b1', 4,(4)), ))

self.studPositionDescriptions.extend(((0.11, 't1', 4,(4)), (0.2626, 't1', 4,(4)), (0.4152, 't1', 4,(4)), (0.5678, 't1', 4,(4)), (0.7204, 't1', 4,(4)), (0.873, 't1', 4,(4)), (1.0256, 't1', 4,(4)), (1.1782, 't1', 4,(4)), (1.3308, 't1', 4,(4)), (1.4834, 't1', 4,(4)), (1.636, 't1', 4,(4)), (1.7866, 't1', 4,(4)), (1.9412, 't1', 4,(4)), (2.0938, 't1', 4,(4)), (2.2464, 't1', 4,(4)), (2.399, 't1', 4,(4)), (2.5516, 't1', 4,(4)), (2.7042, 't1', 4,(4)), (2.8568, 't1', 4,(4)), (3.0094, 't1', 4,(4)), (3.162, 't1', 4,(4)), (3.3146, 't1', 4,(4)), (3.4672, 't1', 4,(4)), (3.6198, 't1', 4,(4)), (3.7724, 't1', 4,(4)), (3.925, 't1', 4,(4)), (4.0776, 't1', 4,(4)), (4.2302, 't1', 4,(4)), (4.3828, 't1', 4,(4)), (4.5354, 't1', 4,(4)), (4.688, 't1', 4,(4)), (4.8406, 't1', 4,(4)), (4.9932, 't1', 4,(4)), (5.1458, 't1', 4,(4)), (5.2994, 't1', 4,(4)), (5.451, 't1', 4,(4)), (5.6036, 't1', 4,(4)), (5.7562, 't1', 4,(4)), (5.9088, 't1', 4,(4)), (6.0614, 't1', 4,(4)), (6.214, 't1', 4,(4)), (6.3666, 't1', 4,(4)), (6.5192, 't1', 4,(4)), (6.6718, 't1', 4,(4)), (6.8244, 't1', 4,(4)), (6.9771, 't1', 4,(4)), (7.1296, 't1', 4,(4)), (7.2822, 't1', 4,(4)), (7.4348, 't1', 4,(4)), (7.5874, 't1', 4,(4)), (7.74, 't1', 4,(4)), ))

self.loadDefPoints = (3907.5/1000.0, 0.0)

if self.Case_Name == 'Chu_WS2.0' or self.beamCaseNo==72:
    self.L = 4148.00/1000.0
    self.bamb = 734.00/1000.0
    self.bamb = 762.00/1000.0
    self.bottomt = 14.00/1000.0
    self.topt = 14.00/1000.0
    self.plateFyBottom = 290.00
    self.plateFyTop = 290.00
    self.plateFyBottom = 440.00
    self.plateFyTop = 440.00
    self.concretef = 33.70
    self.studh1 = 150.00/1000.0
    self.studb1Dia = 19.04/1000.0
    self.studb1FullHeight = False
    self.studh1Resistance = 95.21/1000.0
    self.studh1f = 368.00
    self.studh1fu = 418.00
    self.studh2 = 0.00/1000.0
    self.studh2Dia = 0.00/1000.0
    self.studh2FullHeight = False
    self.studh2Resistance = 0.00/1000.0
    self.studh2f = 1.00
    self.studh2fu = 1.50
    self.studh1 = 150.00/1000.0
    self.studb1D = 19.04/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 95.21/1000.0
    self.studb1f = 368.00
    self.studb1fu = 418.00
    self.studh2 = 0.00/1000.0
    self.studh2Dia = 26.07/1000.0
    self.studh2FullHeight = True
    self.studh2Resistance = 115.31/1000.0
    self.studh2f = 220.00
```

461
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2fu = 270.00

self.CyclicLoad = 50.00/1000.0
self.appliedDeflection = 0.05

self.support1 = 625.00/1000.0
self.supporta = 3523.00/1000.0
self.supportb = -100.00/1000.0
self.supportc = -100.00/1000.0

self.load1 = 2074.00/1000.0
self.loada = -100.00/1000.0
self.loadb = -100.00/1000.0
self.loadc = -100.00/1000.0
self.loadd = -100.00/1000.0
self.loade = -100.00/1000.0
self.loade = -100.00/1000.0
self.loadf = -100.00/1000.0
self.loadg = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactora = -100.00
self.loadFactorb = -100.00
self.loadFactorc = -100.00
self.loadFactord = -100.00
self.loadFactore = -100.00
self.loadFactorf = -100.00
self.loadFactorg = -100.00

self.loadDefPoints = []
sel.loadDefPoints.extend(((2074/1000.0, 0.0),))

self.loadPositionDescriptions.extend(((0.09, 'b1', 4, (4)),(0.2426, 'b1', 4, (4)),(0.3952, 'b1', 4, (4)),(0.5478, 'b1', 4, (4)),(0.7004, 'b1', 4, (4)),(0.853, 'b1', 4, (4)),(1.0058, 'b1', 4, (4)),(1.1582, 'b1', 4, (4)),(1.3108, 'b1', 4, (4)),
self.loadPositionDescriptions.extend(((0.09, 't1', 4, (4)),(0.2426, 't1', 4, (4)),(0.3952, 't1', 4, (4)),(0.5478, 't1', 4, (4)),(0.7004, 't1', 4, (4)),(0.853, 't1', 4, (4)),(1.0058, 't1', 4, (4)),(1.1582, 't1', 4, (4)),(1.3108, 't1', 4, (4)),(1.4634, 't1', 4, (4)),(1.616, 't1', 4, (4)),(1.7686, 't1', 4, (4)),(1.9212, 't1', 4, (4)),(2.0738, 't1', 4, (4)),(2.2264, 't1', 4, (4)),(2.3799, 't1', 4, (4)),(2.5316, 't1', 4, (4)),(2.6842, 't1', 4, (4)),(2.8368, 't1', 4, (4)),(2.9894, 't1', 4, (4)),(3.142, 't1', 4, (4)),(3.2946, 't1', 4, (4)),(3.4472, 't1', 4, (4)),(3.5998, 't1', 4, (4)),(3.7524, 't1', 4, (4)),(3.905, 't1', 4, (4)),(4.0576, 't1', 4, (4)),))
self.loadPositionDescriptions.extend(((0.2426, 't2', 1, (1)),(1.4606, 't2', 1, (1)),(2.6786, 't2', 1, (1)),(3.8966, 't2', 1, (1)),))

print self.Case_Name

if self.Case_Name == 'Chu_WS1.5' or self.beamCaseNo==73:

self.L = 3386.00/1000.0
self.beamb = 734.00/1000.0
self.beambottomt = 14.00/1000.0
self.topt = 14.00/1000.0

self.plateFyBottom = 305.00
self.plateFyTop = 305.00
self.plateFuBottom = 445.00
self.plateFuTop = 445.00
self.concretesf = 38.80

self.load1b = 150.00/1000.0
self.load1Da = 19.04/1000.0
self.load1FullHeight = False
self.load1Resistance = 95.21+1000.0
self.load1yf = 368.00
self.load1fu = 418.00

self.load2h = 0.00/1000.0
self.load2Da = 0.00/1000.0
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2FullHeight = False
self.studb2Resistance = 0.00  # 1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 150.00 / 1000.0
self.studt1Dia = 19.04 / 1000.0
self.studt1FullHeight = False
self.studt1Resistance = 95.21 * 1000.0
self.studt1fy = 1.00
self.studt1fu = 1.50

self.studt2h = 0.00 / 1000.0
self.studt2Dia = 26.07 / 1000.0
self.studt2FullHeight = True
self.studt2Resistance = 115.31 * 1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00 * 1000.0
self.appliedDeflection = 0.05

self.support1 = 625.00 / 1000.0
self.support1a = 2761.00 / 1000.0
self.support1b = -100.00 / 1000.0
self.support1c = -100.00 / 1000.0

self.load1 = 1693.00 / 1000.0
self.load1a = -100.00 / 1000.0
self.load1b = -100.00 / 1000.0
self.load1c = -100.00 / 1000.0
self.load1d = -100.00 / 1000.0
self.load1e = -100.00 / 1000.0
self.load1f = -100.00 / 1000.0
self.load1g = -100.00 / 1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1693 / 1000.0, 0.0),))

for (x, y, z, ...) in self.loadPositionDescriptions:
    print(x, y, z, ...)

print(self.Case_Name)
if self.Case_Name == 'Chu_WS_1.5' or self.beamCaseNo==74:

self.L = 3386.00 / 1000.0
self.beamh = 734.00 / 1000.0
self.beamb = 762.00 / 1000.0
self.bottomt = 14.00 / 1000.0
self.topt = 14.00 / 1000.0
self.plateFyBottom = 300.00

print(self.Case_Name)
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.plateFyTop = 300.00
self.plateFuBottom = 400.00
self.plateFuTop = 400.00
self.concreteFc = 37.70

self.studb1h = 150.00/1000.0
self.studb1Dia = 19.04/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 95.21*1000.0
self.studb1fy = 368.00
self.studb1fu = 418.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 150.00/1000.0
self.studt1Dia = 19.04/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 95.21*1000.0
self.studt1fy = 368.00
self.studt1fu = 418.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 26.07/1000.0
self.studt2FullHeight = True
self.studt2Resistance = 115.31*1000.0
self.studt2fy = 220.00
self.studt2fu = 270.00

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 625.00/1000.0
self.support1a = 2761.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1693.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1693/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.09, 'b1', 4, (4)), (0.2426, 'b1', 4, (4)), (0.3952, 'b1', 4, (4)), (0.5478, 'b1', 4, (4)), (0.7004, 'b1', 4, (4)), (0.853, 'b1', 4, (4)), (1.0056, 'b1', 4, (4)), (1.1582, 'b1', 4, (4)), (1.3108, 'b1', 4, (4)), (1.4634, 'b1', 4, (4)), (1.616, 'b1', 4, (4)), (1.7686, 'b1', 4, (4)), (1.9212, 'b1', 4, (4)), (2.0738, 'b1', 4, (4)), (2.2264, 'b1', 4, (4)), (2.379, 'b1', 4, (4)), (2.5316, 'b1', 4, (4)), (2.6842, 'b1', 4, (4)), (2.8368, 'b1', 4, (4)), (3.9894, 'b1', 4, (4)), (3.142, 'b1', 4, (4)), (3.2946, 'b1', 4, (4)),))
self.studPositionDescriptions.extend(((0.09, 't1', 4, (4)), (0.2426, 't1', 4, (4)), (0.3952, 't1', 4, (4)), (0.5478, 't1', 4, (4)), (0.7004, 't1', 4, (4)), (0.853, 't1', 4, (4)), (1.0056, 't1', 4, (4)), (1.1582, 't1', 4, (4)), (1.3108, 't1', 4, (4)), (1.4634, 't1', 4, (4)), (1.616, 't1', 4, (4)), (1.7686, 't1', 4, (4)), (1.9212, 't1', 4, (4)), (2.0738, 't1', 4, (4)), (2.2264, 't1', 4, (4)), (2.379, 't1', 4, (4)), (2.5316, 't1', 4, (4)), (2.6842, 't1', 4, (4)), (2.8368, 't1', 4, (4)), (3.9894, 't1', 4, (4)),))
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studyDescription = ((3.142, 't1', 4, (4)), (3.2946, 't1', 4, (4)),
                        (3.295, 't1', 4, (4)), (2.9075, 't2', 1, (1))
self.studyPositionDescription.extend(((0.4715, 't2', 1, (1)), (1.0805, 't2', 1, (1)), (1.6895, 't2', 1, (1)), (2.2985, 't2', 1, (1))))

print self.Case_Name
if self.Case_Name == 'Chu_WS_2.0' or self.beamCaseNo==75:
    self.L = 4148.00/1000.0
    self.beamh = 734.00/1000.0
    self.beamb = 762.00/1000.0
    self.bottomt = 14.00/1000.0
    self.topb = 14.00/1000.0
    self.plateFyBottom = 280.00
    self.plateFyTop = 280.00
    self.plateFuBottom = 430.00
    self.plateFuTop = 430.00
    self.concreteFc = 45.20
    self.studb1h = 150.00/1000.0
    self.studb1Dia = 19.04/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 95.21*1000.0
    self.studb1fy = 368.00
    self.studb1fu = 418.00
    self.studb2h = 0.00/1000.0
    self.studb2Dia = 0.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 0.00*1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50
    self.studt1h = 150.00/1000.0
    self.studt1Dia = 19.04/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 95.21*1000.0
    self.studt1fy = 368.00
    self.studt1fu = 418.00
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 26.07/1000.0
    self.studt2FullHeight = True
    self.studt2Resistance = 115.31*1000.0
    self.studt2fy = 220.00
    self.studt2fu = 270.00
    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05
    self.support1 = 0.00/1000.0
    self.support1a = 3523.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 2074.00/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = -100.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
    self.loadFactor1d = -100.00
    self.loadFactor1e = -100.00
```

465
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((2074/1000.0, 0.0),))

self.studPositionDescriptions.extend(((0.09, 'b1', 4, (4)), (0.2426, 'b1', 4, (4)), (0.3952, 'b1', 4, (4)), (0.5478, 'b1', 4, (4)), (0.7004, 'b1', 4, (4)), (1.4634, 'b1', 4, (4)), (1.616, 'b1', 4, (4)), (1.7686, 'b1', 4, (4)), (1.9212, 'b1', 4, (4)), (2.0738, 'b1', 4, (4)), (2.2264, 'b1', 4, (4)), (2.379, 'b1', 4, (4)), (2.5316, 'b1', 4, (4)), (2.6842, 'b1', 4, (4)), (2.8368, 'b1', 4, (4)), (2.9894, 'b1', 4, (4)), (3.142, 'b1', 4, (4)), (3.2946, 'b1', 4, (4)), (3.4472, 'b1', 4, (4)), (3.5998, 'b1', 4, (4)), (3.7524, 'b1', 4, (4)), (3.905, 'b1', 4, (4)), (4.0576, 'b1', 4, (4)),))

self.studPositionDescriptions.extend(((0.09, 't1', 4, (4)), (0.2426, 't1', 4, (4)), (0.3952, 't1', 4, (4)), (0.5478, 't1', 4, (4)), (0.7004, 't1', 4, (4)), (1.4634, 't1', 4, (4)), (1.616, 't1', 4, (4)), (1.7686, 't1', 4, (4)), (1.9212, 't1', 4, (4)), (2.0738, 't1', 4, (4)), (2.2264, 't1', 4, (4)), (2.379, 't1', 4, (4)), (2.5316, 't1', 4, (4)), (2.6842, 't1', 4, (4)), (2.8368, 't1', 4, (4)), (2.9894, 't1', 4, (4)), (3.142, 't1', 4, (4)), (3.2946, 't1', 4, (4)), (3.4472, 't1', 4, (4)), (3.5998, 't1', 4, (4)), (3.7524, 't1', 4, (4)), (3.905, 't1', 4, (4)), (4.0576, 't1', 4, (4)),))

self.studPositionDescriptions.extend(((0.2426, 't2', 1, (1)), (0.8516, 't2', 1, (1)), (1.4606, 't2', 1, (1)), (2.0696, 't2', 1, (1)), (2.6786, 't2', 1, (1)), (3.2878, 't2', 1, (1)), (3.8966, 't2', 1, (1)),))
```

print self.Case_Name

if self.Case_Name == 'Chu_WS_3.0T' or self.beamCaseNo==76:

```
self.L = 9586.0/1000.0
self.beamh = 734.0/1000.0
self.beamb = 762.0/1000.0
self.bottomt = 14.0/1000.0
self.top = 14.0/1000.0
self.plateFyBottom = 315.0
self.plateFyTop = 315.0
self.plateFuBottom = 440.0
self.plateFuTop = 440.0
self.concreteFc = 36.40

self.studb1h = 150.0/1000.0
self.studb1Dia = 19.04/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 95.21*1000.0
self.studb1fy = 368.00
self.studb1fu = 418.00

self.studb2h = 0.0/1000.0
self.studb2Dia = 0.0/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.0*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 150.0/1000.0
self.studt1Dia = 19.04/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 95.21*1000.0
self.studt1fy = 368.00
self.studt1fu = 418.00

self.studt2h = 0.0/1000.0
self.studt2Dia = 26.07/1000.0
self.studt2FullHeight = True
self.studt2Resistance = 115.31*1000.0
self.studt2fy = 220.0
self.studt2fu = 270.0
```

print self.Case_Name

if self.Case_Name == 'Chu_WS_3.0T' or self.beamCaseNo==76:

```
```
Appendix B: Test Database in Python Code Format

```python
# self.supportle = -100.00/1000.0
self.load1 = 2425.00/1000.0
self.loada = 7161.00/1000.0
self.loadb = -100.00/1000.0
self.loadc = -100.00/1000.0
self.loadd = -100.00/1000.0
self.loade = -100.00/1000.0
self.loadf = -100.00/1000.0
self.loadg = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor2 = 1.00
self.loadFactor3 = -100.00
self.loadFactor4 = -100.00
self.loadFactor5 = -100.00
self.loadFactor6 = -100.00
self.loadFactor7 = 1
self.loadDefPoints = []
self.loadDefPoints.extend(((4793/1000.0,0.0),))

self.studyPositionDescriptions.extend(((0.08,'b1','4','4'),(0.2326,'b1','4','4'),(0.3852,'b1','4','4'),(0.53578,'b1','4','4'),(0.6904,'b1','4','4'),(0.843,'b1','4','4'),(0.9956,'b1','4','4'),(1.1482,'b1','4','4'),(1.3008,'b1','4','4'),(1.4534,'b1','4','4'),(1.606,'b1','4','4'),(1.7586,'b1','4','4'),(1.9112,'b1','4','4'),(2.0638,'b1','4','4'),(2.2164,'b1','4','4'),(2.369,'b1','4','4'),(2.5216,'b1','4','4'),(2.6742,'b1','4','4'),(2.8268,'b1','4','4'),(2.9794,'b1','4','4'),(3.132,'b1','4','4'),(3.2846,'b1','4','4'),(3.4372,'b1','4','4'),(3.5898,'b1','4','4'),(3.7424,'b1','4','4'),(3.895,'b1','4','4'),(4.0476,'b1','4','4'),(4.2002,'b1','4','4'),(4.3528,'b1','4','4'),(4.5054,'b1','4','4'),(4.658,'b1','4','4'),(4.8106,'b1','4','4'),(4.9632,'b1','4','4'),(5.1158,'b1','4','4'),(5.2684,'b1','4','4'),(5.421,'b1','4','4'),(5.5736,'b1','4','4'),(5.7262,'b1','4','4'),(5.8788,'b1','4','4'),(6.0314,'b1','4','4'),(6.184,'b1','4','4'),(6.3366,'b1','4','4'),(6.4892,'b1','4','4'),(6.6418,'b1','4','4'),(6.7944,'b1','4','4'),(6.947,'b1','4','4'),(7.0996,'b1','4','4'),(7.2522,'b1','4','4'),(7.4048,'b1','4','4'),(7.5574,'b1','4','4'),(7.71,'b1','4','4'),(7.8626,'b1','4','4'),(8.0152,'b1','4','4'),(8.1678,'b1','4','4'),(8.3204,'b1','4','4'),(8.473,'b1','4','4'),(8.6256,'b1','4','4'),(8.7782,'b1','4','4'),(8.9308,'b1','4','4'),(9.0834,'b1','4','4'),(9.236,'b1','4','4'),(9.3886,'b1','4','4'),(9.5412,'b1','4','4'),(9.6938,'b1','4','4'),(9.8467,'b1','4','4'),(10.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00)

self.studyPositionDescriptions.extend(((0.08,'t1','4','4'),(0.2326,'t1','4','4'),(0.3852,'t1','4','4'),(0.53578,'t1','4','4'),(0.6904,'t1','4','4'),(0.843,'t1','4','4'),(0.9956,'t1','4','4'),(1.1482,'t1','4','4'),(1.3008,'t1','4','4'),(1.4534,'t1','4','4'),(1.606,'t1','4','4'),(1.7586,'t1','4','4'),(1.9112,'t1','4','4'),(2.0638,'t1','4','4'),(2.2164,'t1','4','4'),(2.369,'t1','4','4'),(2.5216,'t1','4','4'),(2.6742,'t1','4','4'),(2.8268,'t1','4','4'),(2.9794,'t1','4','4'),(3.132,'t1','4','4'),(3.2846,'t1','4','4'),(3.4372,'t1','4','4'),(3.5898,'t1','4','4'),(3.7424,'t1','4','4'),(3.895,'t1','4','4'),(4.0476,'t1','4','4'),(4.2002,'t1','4','4'),(4.3528,'t1','4','4'),(4.5054,'t1','4','4'),(4.658,'t1','4','4'),(4.8106,'t1','4','4'),(4.9632,'t1','4','4'),(5.1158,'t1','4','4'),(5.2684,'t1','4','4'),(5.421,'t1','4','4'),(5.5736,'t1','4','4'),(5.7262,'t1','4','4'),(5.8788,'t1','4','4'),(6.0314,'t1','4','4'),(6.184,'t1','4','4'),(6.3366,'t1','4','4'),(6.4892,'t1','4','4'),(6.6418,'t1','4','4'),(6.7944,'t1','4','4'),(6.947,'t1','4','4'),(7.0996,'t1','4','4'),(7.2522,'t1','4','4'),(7.4048,'t1','4','4'),(7.5574,'t1','4','4'),(7.71,'t1','4','4'),(7.8626,'t1','4','4'),(8.0152,'t1','4','4'),(8.1678,'t1','4','4'),(8.3204,'t1','4','4'),(8.473,'t1','4','4'),(8.6256,'t1','4','4'),(8.7782,'t1','4','4'),(8.9308,'t1','4','4'),(9.0834,'t1','4','4'),(9.236,'t1','4','4'),(9.3886,'t1','4','4'),(9.5412,'t1','4','4'),(9.6938,'t1','4','4'),(9.8467,'t1','4','4'),(10.00,1.00,1.00,1.00,1.00,1.00,1.00,1.00)

print self.Case_Name
if self.Case_Name == 'Chu_WS_1.5T' or self.beamCaseNo==77:

self.L = 7400.00/1000.0
self.beamb = 734.00/1000.0
self.bamb = 762.00/1000.0
self.bottom = 14.00/1000.0
self.topt = 14.00/1000.0
self.plateFyBottom = 285.00
self.plateFyTop = 285.00
self.plateFbBottom = 425.00
self.plateFbTop = 425.00
self.concretef = 27.50
self.studbh = 150.00/1000.0
self.studbhDa = 19.04/1000.0
self.studbfFullHeight = False
self.studbResistance = 95.17*1000.0
self.studbfly = 368.00

467
self.studbfu = 418.00
self.studbfh = 418.00/1000.0
self.stud12h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50
self.stud1th = 150.00/1000.0
self.stud1Dia = 19.04/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 95.17+1000.0
self.stud1fy = 368.00
self.stud1fu = 418.00
self.stud12h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50
self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05
self.supportl = 625.00/1000.0
self.supportra = 6775.00/1000.0
self.supportlb = -100.00/1000.0
self.supportlc = -100.00/1000.0
self.load1 = 1726.00/1000.0
self.loadla = 5674.00/1000.0
self.load1b = -100.00/1000.0
self.loadlc = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1f = -100.00/1000.0
self.loadlg = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactorla = 1.00
self.loadFactorlb = -100.00
self.loadFactorlc = -100.00
self.loadFactorlf = -100.00
self.loadFactorlg = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((3701/1000.0,0.0),))
self.studPositionDescriptions.extend((self,))
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
print self.Case_Name
if self.Case_Name == 'SCIENCE_SP1' or self.beamCaseNo==78:

    self.L = 9540.00/1000.0
    self.beamh = 800.00/1000.0
    self.beamb = 640.00/1000.0
    self.bottomt = 15.00/1000.0
    self.topt = 10.00/1000.0

    self.plateFyBottom = 396.20
    self.pateFyTop = 376.00
    self.pateFubottom = 550.40
    self.pateFuTop = 531.00
    self.concrete = 37.30

    self.stud1h = 100.00/1000.0
    self.stud1Dia = 19.00/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 113.18+1000.0
    self.stud1fy = 399.00
    self.stud1fu = 499.00

    self.stud2h = 400.00/1000.0
    self.stud2Dia = 24.00/1000.0
    self.stud2FullHeight = True
    self.stud2Resistance = 184.35+1000.0
    self.stud2fy = 385.80
    self.stud2fu = 540.60

    self.stud1th = 100.00/1000.0
    self.stud1Dia = 19.00/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 113.18+1000.0
    self.stud1fy = 399.00
    self.stud1fu = 499.00

    self.stud2h = 0.00/1000.0
    self.stud2Dia = 0.00/1000.0
    self.stud2FullHeight = False
    self.stud2Resistance = 0.00+1000.0
    self.stud2fy = 1.00
    self.stud2fu = 1.50

    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05

    self.support1 = 270.00/1000.0
    self.support1a = 9270.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0

    self.load1 = 3870.00/1000.0
    self.load1a = 5670.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0

    self.loadFactor1 = 1.00
    self.loadFactor1a = 1.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
    self.loadFactor1d = -100.00
    self.loadFactor1e = -100.00
    self.loadFactor1f = -100.00
    self.loadFactor1g = -100.00
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadDefPoints = []
self.loadDefPoints.extend(((4770/1000.0,0.0),))

self.studyPositionDescriptions.extend(((0.08, 'b1', 3,(3)),(0.68, 'b1', 3,(3)),(1.28, 'b1', 3,(3)),(1.88, 'b1', 3,(3)),
  (2.48, 'b1', 3,(3)),(3.08, 'b1', 3,(3)),(3.68, 'b1', 3,(3)),(4.28, 'b1', 3,(3)),(4.88, 'b1', 3,(3)),(5.48, 'b1', 3,(3)),
  (6.08, 'b1', 3,(3)),(6.68, 'b1', 3,(3)),(7.28, 'b1', 3,(3)),(7.88, 'b1', 3,(3)),(8.48, 'b1', 3,(3)),(9.08, 'b1', 3,(3)),
  (0.48, 'b1', 3,(3)),(1.08, 'b1', 3,(3)),(1.68, 'b1', 3,(3)),(2.28, 'b1', 3,(3)),(2.88, 'b1', 3,(3)),(3.48, 'b1', 3,(3)),
  (4.08, 'b1', 3,(3)),(4.68, 'b1', 3,(3)),(5.28, 'b1', 3,(3)),(5.88, 'b1', 3,(3)),(6.48, 'b1', 3,(3)),(7.08, 'b1', 3,(3)),
  (7.68, 'b1', 3,(3)),(8.28, 'b1', 3,(3)),(8.88, 'b1', 3,(3)),(9.48, 'b1', 3,(3))))

self.studyPositionDescriptions.extend(((0.28, 'b2', 3,(3)),(0.88, 'b2', 3,(3)),(1.48, 'b2', 3,(3)),(2.08, 'b2', 3,(3)),
  (2.68, 'b2', 3,(3)),(3.28, 'b2', 3,(3)),(3.88, 'b2', 3,(3)),(4.48, 'b2', 3,(3)),(5.08, 'b2', 3,(3)),(5.68, 'b2', 3,(3)),
  (6.28, 'b2', 3,(3)),(6.88, 'b2', 3,(3)),(7.48, 'b2', 3,(3)),(8.08, 'b2', 3,(3)),(8.68, 'b2', 3,(3)),(9.28, 'b2', 3,(3)),))

self.studyPositionDescriptions.extend(((0.08, 't1', 3,(3)),(0.68, 't1', 3,(3)),(1.28, 't1', 3,(3)),(1.88, 't1', 3,(3)),
  (2.48, 't1', 3,(3)),(3.08, 't1', 3,(3)),(3.68, 't1', 3,(3)),(4.28, 't1', 3,(3)),(4.88, 't1', 3,(3)),(5.48, 't1', 3,(3)),
  (6.08, 't1', 3,(3)),(6.68, 't1', 3,(3)),(7.28, 't1', 3,(3)),(7.88, 't1', 3,(3)),(8.48, 't1', 3,(3)),(9.08, 't1', 3,(3)),
  (3.48, 't1', 3,(3)),(4.08, 't1', 3,(3)),(4.68, 't1', 3,(3)),(5.28, 't1', 3,(3)),(5.88, 't1', 3,(3)),(6.48, 't1', 3,(3)),
  (7.08, 't1', 3,(3)),(7.68, 't1', 3,(3)),(8.28, 't1', 3,(3)),(8.88, 't1', 3,(3)),(9.48, 't1', 3,(3))))

print self.Case_Name
if self.Case_Name == 'SCIENCE_SP2' or self.beamCaseNo==79:
  self.L = 9540.00/1000.0
  self.beamh = 800.00/1000.0
  self.beamb = 640.00/1000.0
  self.bottomt = 15.00/1000.0
  self.topt = 10.00/1000.0

  self.plateFyBottom = 396.20
  self.plateFyTop = 309.40
  self.plateFuBottom = 550.40
  self.plateFuTop = 609.60
  self.concretefc = 36.70

  self.studb1h = 100.00/1000.0
  self.studb1Dia = 19.00/1000.0
  self.studb1FullHeight = False
  self.studb1Resistance = 113.18*1000.0
  self.studb1fy = 399.00
  self.studb1fu = 499.00

  self.studb2h = 400.00/1000.0
  self.studb2Dia = 24.00/1000.0
  self.studb2FullHeight = True
  self.studb2Resistance = 182.42*1000.0
  self.studb2fy = 385.80
  self.studb2fu = 540.60

  self.studt1h = 100.00/1000.0
  self.studt1Dia = 19.00/1000.0
  self.studt1FullHeight = False
  self.studt1Resistance = 113.18*1000.0
  self.studt1fy = 399.00
  self.studt1fu = 499.00

  self.studt2h = 0.00/1000.0
  self.studt2Dia = 0.00/1000.0
  self.studt2FullHeight = False
  self.studt2Resistance = 0.00*1000.0
  self.studt2fy = 1.00
  self.studt2fu = 1.50

  self.CyclicLoad = 50.00*1000.0
  self.appliedDeflection = 0.05

  self.support1 = 270.00/1000.0
  self.support1a = 9270.00/1000.0
  self.support1b = -100.00/1000.0
  self.support1c = -100.00/1000.0

470
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.load1 = 3870.00 / 1000.0
self.load1a = 5670.00 / 1000.0
self.load1b = -100.00 / 1000.0
self.load1c = -100.00 / 1000.0
self.load1d = -100.00 / 1000.0
self.load1e = -100.00 / 1000.0
self.load1f = -100.00 / 1000.0
self.load1g = -100.00 / 1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((4770 / 1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.08, 'b1', 3, (3)), (0.68, 'b1', 3, (3)), (1.28, 'b1', 3, (3)), (1.88, 'b1', 3, (3)),
                                      (2.48, 'b1', 3, (3)), (3.08, 'b1', 3, (3)), (3.68, 'b1', 3, (3)), (4.28, 'b1', 3, (3)), (4.88, 'b1', 3, (3)), (5.48, 'b1', 3, (3)),
                                      (6.08, 'b1', 3, (3)), (6.68, 'b1', 3, (3)), (7.28, 'b1', 3, (3)), (7.88, 'b1', 3, (3)), (8.48, 'b1', 3, (3)), (9.08, 'b1', 3, (3)),
                                      (9.68, 'b1', 3, (3)), (1.48, 'b1', 3, (3)), (2.08, 'b1', 3, (3)), (2.68, 'b1', 3, (3)), (3.28, 'b1', 3, (3)), (3.88, 'b1', 3, (3)), (4.48, 'b1', 3, (3)),
                                      (5.08, 'b1', 3, (3)), (5.68, 'b1', 3, (3)), (6.28, 'b1', 3, (3)), (6.88, 'b1', 3, (3)), (7.48, 'b1', 3, (3)), (8.08, 'b1', 3, (3)), (8.68, 'b1', 3, (3)), (9.28, 'b1', 3, (3)))
self.studPositionDescriptions.extend(((0.08, 't1', 3, (3)), (0.68, 't1', 3, (3)), (1.28, 't1', 3, (3)), (1.88, 't1', 3, (3)),
                                      (2.48, 't1', 3, (3)), (3.08, 't1', 3, (3)), (3.68, 't1', 3, (3)), (4.28, 't1', 3, (3)), (4.88, 't1', 3, (3)), (5.48, 't1', 3, (3)),
                                      (6.08, 't1', 3, (3)), (6.68, 't1', 3, (3)), (7.28, 't1', 3, (3)), (7.88, 't1', 3, (3)), (8.48, 't1', 3, (3)), (9.08, 't1', 3, (3)),
                                      (9.68, 't1', 3, (3)), (1.48, 't1', 3, (3)), (2.08, 't1', 3, (3)), (2.68, 't1', 3, (3)), (3.28, 't1', 3, (3)), (3.88, 't1', 3, (3)), (4.48, 't1', 3, (3)),
                                      (5.08, 't1', 3, (3)), (5.68, 't1', 3, (3)), (6.28, 't1', 3, (3)), (6.88, 't1', 3, (3)), (7.48, 't1', 3, (3)), (8.08, 't1', 3, (3)), (8.68, 't1', 3, (3)), (9.28, 't1', 3, (3))))

print self.Case_Name
if self.Case_Name == 'SCIENCE_SP3_E2' or self.beamCaseNo==81:
    self.L = 8340.00 / 1000.0
    self.beamb = 800.00 / 1000.0
    self.beamb = 800.00 / 1000.0
    self.bottomt = 15.00 / 1000.0
    self.top = 15.00 / 1000.0
    self.plateFyBottom = 396.20
    self.plateFyTop = 396.20
    self.plateFuBottom = 550.40
    self.plateFuTop = 550.40
    self.concretefc = 38.10
    self.studb1h = 100.00 / 1000.0
    self.studb1Dia = 19.00 / 1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 113.18 * 1000.0
    self.studb1fy = 399.00
    self.studb1fu = 499.00
    self.studb2h = 400.00 / 1000.0
    self.studb2Dia = 18.00 / 1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 105.14 * 1000.0
    self.studb2fy = 624.00
    self.studb2fu = 673.20
    self.studt1h = 100.00 / 1000.0
    self.studt1Dia = 19.00 / 1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 113.18 * 1000.0
    self.studt1fy = 399.00
```

471
```python
if self.Case_Name == 'SCIENCE_SP4_E2' or self.beamCaseNo==83:
    self.L = 8340.00/1000.0
    self.bamb = 800.00/1000.0
    self.bottomt = 15.00/1000.0
    self.toppt = 15.00/1000.0

    self.plateFyBottom = 396.20
    self.plateFyTop = 396.20
    self.plateFullBottom = 550.40
```

472
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.plateFuTop = 550.40
self.concretefC = 43.60
self.studb1h = 100.00/1000.0
self.studb1Dia = 16.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 113.18*1000.0
self.studb1fy = 931.80
self.studb1fu = 1026.00
self.studt1h = 100.00/1000.0
self.studt1Dia = 19.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 113.18*1000.0
self.studt1fy = 399.00
self.studt1fu = 499.00
self.studb2h = 400.00/1000.0
self.studb2Dia = 19.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 90.68*1000.0
self.studb2fy = 931.80
self.studb2fu = 1026.00
self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05
self.support1 = 270.00/1000.0
self.support1a = 8070.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 3270.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((4170/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.08, 'b1', 5,(5)),(0.68, 'b1', 5,(5)),(1.28, 'b1', 5,(5)),(1.88, 'b1', 5,(5))
  , (2.48, 'b1', 5,(5)),(3.08, 'b1', 5,(5)),(3.68, 'b1', 5,(5)),(4.28, 'b1', 5,(5)),(4.88, 'b1', 5,(5)), (5.48, 'b1', 5,(5))
  , (6.08, 'b1', 5,(5)),(6.68, 'b1', 5,(5)),(7.28, 'b1', 5,(5)),(7.88, 'b1', 5,(5)),(8.48, 'b1', 5,(5)),(9.08, 'b1', 5,(5))
  , (1.68, 'b1', 5,(5)),(2.28, 'b1', 5,(5)),(2.88, 'b1', 5,(5)),(3.48, 'b1', 5,(5)),(4.08, 'b1', 5,(5)),(4.68, 'b1', 5,(5))
  , (5.28, 'b1', 5,(5)),(5.88, 'b1', 5,(5)),(6.48, 'b1', 5,(5)),(7.08, 'b1', 5,(5)),(7.68, 'b1', 5,(5)),(8.28, 'b1', 5,(5))
  , (0.28, 'b2', 2,(2)),(0.88, 'b2', 2,(2)),(1.48, 'b2', 3,(3)),(2.08, 'b2', 3,(3)),(2.68, 'b2', 3,(3)),(3.28, 'b2', 3,(3))
  , (3.88, 'b2', 3,(3)),(4.48, 'b2', 3,(3)),(5.08, 'b2', 3,(3)),(5.68, 'b2', 3,(3)),(6.28, 'b2', 3,(3))
  , (7.48, 'b2', 2,(2)),(8.08, 'b2', 2,(2)),))
self.studPositionDescriptions.extend(((0.08, 'b1', 3,(3)),(0.68, 'b1', 3,(3)),(1.28, 'b1', 3,(3)),(1.88, 'b1', 3,(3))
  , (2.48, 'b1', 3,(3)),(3.08, 'b1', 3,(3)),(3.68, 'b1', 3,(3)),(4.28, 'b1', 3,(3)),(4.88, 'b1', 3,(3)), (5.48, 'b1', 3,(3))
  , (6.08, 'b1', 3,(3)),(6.68, 'b1', 3,(3)),(7.28, 'b1', 3,(3)),(7.88, 'b1', 3,(3)),(8.48, 'b1', 3,(3))
  , (0.28, 'b2', 2,(2)),(0.88, 'b2', 2,(2)),(1.48, 'b2', 3,(3)),(2.08, 'b2', 3,(3)),(2.68, 'b2', 3,(3)),(3.28, 'b2', 3,(3))
  , (3.88, 'b2', 3,(3)),(4.48, 'b2', 3,(3)),(5.08, 'b2', 3,(3)),(5.68, 'b2', 3,(3)),(6.28, 'b2', 3,(3))
  , (7.48, 'b2', 2,(2)),(8.08, 'b2', 2,(2)),))
```

473
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studPositionDescriptions.extend(((0.08, 't1', 5, (5)), (1.28, 't1', 5, (5)), (1.88, 't1', 5, (5)), (2.48, 't1', 5, (5)), (3.08, 't1', 5, (5)), (3.68, 't1', 5, (5)), (4.28, 't1', 5, (5)), (4.88, 't1', 5, (5)), (5.48, 't1', 5, (5)), (6.08, 't1', 5, (5)), (6.68, 't1', 5, (5)), (7.28, 't1', 5, (5)), (7.88, 't1', 5, (5)), (8.48, 't1', 5, (5)), (9.08, 't1', 5, (5)), (9.68, 't1', 5, (5)), (10.28, 't1', 5, (5)), (10.88, 't1', 5, (5)), (11.48, 't1', 5, (5)), (12.08, 't1', 5, (5)), (12.68, 't1', 5, (5)), (13.28, 't1', 5, (5)), (13.88, 't1', 5, (5)), (14.48, 't1', 5, (5)), (15.08, 't1', 5, (5)), (15.68, 't1', 5, (5)), (16.28, 't1', 5, (5)), (16.88, 't1', 5, (5)), (17.48, 't1', 5, (5)), (18.08, 't1', 5, (5)), (18.68, 't1', 5, (5)), (19.28, 't1', 5, (5)), (19.88, 't1', 5, (5)), (20.48, 't1', 5, (5)), (21.08, 't1', 5, (5)), (21.68, 't1', 5, (5)), (22.28, 't1', 5, (5)), (22.88, 't1', 5, (5)), (23.48, 't1', 5, (5))))

print self.Case_Name  
if self.Case_Name == 'SCIENCE_SP5_E2' or self.beamCaseNo==5:

    self.L = 8340.00/1000.0
    self.beamb = 800.00/1000.0
    self.bottomt = 12.00/1000.0
    self.topt = 12.00/1000.0

    self.plateFyBottom = 428.60
    self.plateFuTop = 428.60
    self.concretefc = 39.50

    self.studb1h = 100.00/1000.0
    self.studb1Dia = 19.00/1000.0
    self.studb1Resistance = 113.18  
    self.studb1fy = 399.00
    self.studb1fu = 499.00

    self.studb2h = 400.00/1000.0
    self.studb2Dia = 18.00/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 107.63+1000.0
    self.studb2fy = 624.00
    self.studb2fu = 673.20

    self.studt1h = 100.00/1000.0
    self.studt1Dia = 19.00/1000.0
    self.studt1FullHeight = True
    self.studt1Resistance = 113.18+1000.0
    self.studt1fy = 399.00
    self.studt1fu = 499.00

    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00+1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50

    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05

    self.support1 = 270.00/1000.0
    self.support1a = 8070.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0

    self.load1 = 3270.00/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00

474
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1a = -100.0
self.loadFactor1b = -100.0
self.loadFactor1c = -100.0
self.loadFactor1d = -100.0
self.loadFactor1e = -100.0
self.loadFactor1f = -100.0
self.loadFactor1g = -100.0
self.loadDefPoints = []
self.loadDefPoints.extend([(4170/1000.0,0.0),])
self.studPositionDescriptions.extend([(0.27,'b1',2,(2)),(1.27,'b1',2,(2)),(2.27,'b1',3,(3)),(3.27,'b1',3,(3)),(4.27,'b1',5,(5)),(5.02,'b1',5,(5)),(6.02,'b1',5,(5)),(6.95,'b1',5,(5)),(7.85,'b1',5,(5)),(8.75,'b1',5,(5)),])
self.studPositionDescriptions.extend([(0.27,'b2',3,(3)),(1.27,'b2',3,(3)),(2.27,'b2',2,(2)),(3.27,'b2',2,(2)),(4.17,'b2',2,(2)),(5.07,'b2',2,(2)),(5.97,'b2',2,(2)),(6.97,'b2',2,(2)),(7.87,'b2',2,(2)),(8.77,'b2',2,(2)),(9.67,'b2',2,(2)),])
self.studPositionDescriptions.extend([(0.27,'t1',2,(2)),(1.27,'t1',2,(2)),(2.27,'t1',3,(3)),(3.27,'t1',3,(3)),(4.17,'t1',3,(3)),(5.07,'t1',3,(3)),(6.07,'t1',3,(3)),(7.07,'t1',3,(3)),(8.07,'t1',3,(3)),(9.07,'t1',3,(3)),])
self.studPositionDescriptions.extend([(0.27,'t2',5,(5)),(1.27,'t2',5,(5)),(2.27,'t2',5,(5)),(3.27,'t2',5,(5)),(4.17,'t2',5,(5)),(5.07,'t2',5,(5)),(6.07,'t2',5,(5)),(7.07,'t2',5,(5)),(8.07,'t2',5,(5)),(9.07,'t2',5,(5)),])

print self.Case_Name if self.Case_Name == 'SCIENCE_SP6_E2' or self.beamCaseNo==87:
    self.L = 8340.00/1000.0
    self.beamh = 800.00/1000.0
    self.beamb = 800.00/1000.0
    self.bottomt = 12.00/1000.0
    self.top = 12.00/1000.0
    self.plateFyBottom = 428.60
    self.plateFyTop = 428.60
    self.plateFuBottom = 556.60
    self.plateFuTop = 556.60
    self.concretefc = 37.70
    self.stud1h = 100.00/1000.0
    self.stud1Dia = 19.00/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 113.18*1000.0
    self.stud1fy = 399.00
    self.stud1fu = 499.00
    self.stud2h = 400.00/1000.0
    self.stud2Dia = 18.00/1000.0
    self.stud2FullHeight = True
    self.stud2Resistance = 104.42*1000.0
    self.stud2fy = 424.00
    self.stud2fu = 673.20
    self.stud3h = 100.00/1000.0
    self.stud3Dia = 19.00/1000.0
    self.stud3FullHeight = False
    self.stud3Resistance = 113.18*1000.0
    self.stud3fy = 399.00
    self.stud3fu = 499.00
    self.stud4h = 0.00/1000.0
    self.stud4Dia = 0.00/1000.0
    self.stud4FullHeight = False
    self.stud4Resistance = 0.00*1000.0
    self.stud4fy = 1.00
    self.stud4fu = 1.50
```

475
self.CyclicLoad = 50.00/1000.0
self.appliedDeflection = 0.05
self.support1 = 270.00/1000.0
self.support1a = 8070.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 3270.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadDefPoints = []
self.loadDefPoints.extend(((4170/1000.0,0.0),))

self.loadFactor1a = -self.loadFactor1b = -self.loadFactor1c = -self.loadFactor1d = -self.loadFactor1e = -
self.loadFactor1g = -self.loadFactor1f = -
self.loadFactor1 = 1.00

self.studPositionDescriptions.extend(((0.08, 'b1', 5,(5)),(0.68, 'b1', 5,(5)),(1.88, 'b1', 5,(5))
   ,(2.48, 'b1', 5,(5)),(3.08, 'b1', 5,(5)),(3.68, 'b1', 5,(5)),(4.28, 'b1', 5,(5)),(4.88, 'b1', 5,(5)),(5.48, 'b1', 5,(5))
   ,(6.08, 'b1', 5,(5)),(6.68, 'b1', 5,(5)),(7.28, 'b1', 5,(5)),(7.88, 'b1', 5,(5)),(8.48, 'b1', 5,(5)),(9.08, 'b1', 5,(5))
   ,(1.68, 'b1', 5,(5)),(2.28, 'b1', 5,(5)),(2.88, 'b1', 5,(5)),(3.48, 'b1', 5,(5)),(4.08, 'b1', 5,(5)),(4.68, 'b1', 5,(5))
   ,(5.28, 'b1', 5,(5)),(5.88, 'b1', 5,(5)),(6.48, 'b1', 5,(5)),(7.08, 'b1', 5,(5)),(7.68, 'b1', 5,(5)),(8.28, 'b1', 5,(5))
   ,(0.28, 'b1', 2,(2)),(0.88, 'b1', 5,(5)),(1.48, 'b1', 3,(3)),(2.08, 'b1', 3,(3)),(2.68, 'b1', 3,(3)),(3.28, 'b1', 3,(3))
   ,(3.88, 'b1', 3,(3)),(4.48, 'b1', 3,(3)),(5.08, 'b1', 3,(3)),(5.68, 'b1', 3,(3)),(6.28, 'b1', 3,(3)),(6.88, 'b1', 3,(3))
   ,(7.48, 'b1', 2,(2)),(8.08, 'b1', 2,(2)))

self.loadPositionDescriptions.extend(((0.28, 'b2', 3,(3)),(0.88, 'b2', 3,(3)),(1.48, 'b2', 3,(3)),(2.08, 'b2', 3,(3))
   ,(2.68, 'b2', 3,(3)),(3.28, 'b2', 3,(3)),(3.88, 'b2', 3,(3)),(4.48, 'b2', 3,(3)),(5.08, 'b2', 3,(3)),(5.68, 'b2', 3,(3))
   ,(6.28, 'b2', 3,(3)),(6.88, 'b2', 3,(3)),(7.48, 'b2', 3,(3)),(8.08, 'b2', 3,(3)))

self.loadPositionDescriptions.extend(((0.08, 't1', 5,(5)),(0.68, 't1', 5,(5)),(1.28, 't1', 5,(5)),(1.88, 't1', 5,(5))
   ,(2.48, 't1', 5,(5)),(3.08, 't1', 5,(5)),(3.68, 't1', 5,(5)),(4.28, 't1', 5,(5)),(4.88, 't1', 5,(5)),(5.48, 't1', 5,(5))
   ,(6.08, 't1', 5,(5)),(6.68, 't1', 5,(5)),(7.28, 't1', 5,(5)),(7.88, 't1', 5,(5)),(8.48, 't1', 5,(5)),(9.08, 't1', 5,(5))
   ,(1.68, 't1', 5,(5)),(2.28, 't1', 5,(5)),(2.88, 't1', 5,(5)),(3.48, 't1', 5,(5)),(4.08, 't1', 5,(5)),(4.68, 't1', 5,(5))
   ,(5.28, 't1', 5,(5)),(5.88, 't1', 5,(5)),(6.48, 't1', 5,(5)),(7.08, 't1', 5,(5)),(7.68, 't1', 5,(5)),(8.28, 't1', 5,(5))
   ,(0.28, 't1', 2,(2)),(0.88, 't1', 5,(5)),(1.48, 't1', 3,(3)),(2.08, 't1', 3,(3)),(2.68, 't1', 3,(3)),(3.28, 't1', 3,(3))
   ,(3.88, 't1', 3,(3)),(4.48, 't1', 3,(3)),(5.08, 't1', 3,(3)),(5.68, 't1', 3,(3)),(6.28, 't1', 3,(3)),(6.88, 't1', 3,(3))
   ,(7.48, 't1', 2,(2)),(8.08, 't1', 2,(2)))

self.beamCaseNo==89:

if self.Case_Name == 'R,E&N_B1' or self.beamCaseNo==89:
    self.L = 1700.00/1000.0
    self.beamb = 166.08/1000.0
    self.beamb = 400.00/1000.0
    self.beamb = 804.00/1000.0
    self.teopt = 8.04/1000.0

    self.plateFyBottom = 202.00
    self.plateFyTop = 202.00
    self.plateFullBottom = 252.00
    self.plateFullTop = 252.00
    self.concretefc = 34.50

    self.studbh = 150.00/1000.0
    self.stud3Dia = 10.00/1000.0
    self.studFullHeight = False
    self.studResistance = 28.27+1000.0
    self.studPy = 400.00

print self.Case_Name
self.studb1fu = 450.00
self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00+1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.stud1h = 65.00/1000.0
self.stud1Dia = 6.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 10.18+1000.0
self.stud1fy = 400.00
self.stud1fu = 450.00

self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

cyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 100.00/1000.0
self.support2a = 1600.00/1000.0
self.support2b = -100.00/1000.0
self.support2c = -100.00/1000.0

self.load1 = 300.00/1000.0
self.load1a = 650.00/1000.0
self.load1b = 1050.00/1000.0
self.load1c = 1400.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = 1.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((850/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.045, 'b1', 4, (4)), (0.14, 'b1', 4, (4)), (0.235, 'b1', 4, (4)), (0.33, 'b1', 4, (4)), (0.425, 'b1', 4, (4)), (0.52, 'b1', 4, (4)), (0.615, 'b1', 4, (4)), (0.71, 'b1', 4, (4)), (0.805, 'b1', 4, (4)), (0.895, 'b1', 4, (4)), (0.99, 'b1', 4, (4)), (1.085, 'b1', 4, (4)), (1.18, 'b1', 4, (4)), (1.275, 'b1', 4, (4)), (1.37, 'b1', 4, (4)), (1.465, 'b1', 4, (4)), (1.56, 'b1', 4, (4)), (1.655, 'b1', 4, (4))))
self.studPositionDescriptions.extend(((0.045, 't1', 4, (4)), (0.13, 't1', 4, (4)), (0.215, 't1', 4, (4)), (0.3, 't1', 4, (4)), (0.385, 't1', 4, (4)), (0.47, 't1', 4, (4)), (0.555, 't1', 4, (4)), (0.64, 't1', 4, (4)), (0.725, 't1', 4, (4)), (0.81, 't1', 4, (4)), (0.89, 't1', 4, (4)), (0.975, 't1', 4, (4)), (1.06, 't1', 4, (4)), (1.145, 't1', 4, (4)), (1.23, 't1', 4, (4)), (1.315, 't1', 4, (4)), (1.4, 't1', 4, (4)), (1.485, 't1', 4, (4)), (1.57, 't1', 4, (4)), (1.655, 't1', 4, (4))))

print self.Case_Name
if self.Case_Name == 'R,J & N_B2' or self.beamCaseNo==90:

self.L = 1700.00/1000.0
self.beamb = 162.13/1000.0
self.beamb = 400.00/1000.0
self.bottom = 8.10/1000.0
self.topt = 4.03/1000.0
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.plateFyBottom = 325.00
self.plateFyTop = 224.00
self.plateFuBottom = 375.00
self.plateFuTop = 274.00
self.concretefc = 37.20

self.studb1h = 150.00/1000.0
self.studb1Dia = 10.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 28.27 * 1000.0
self.studb1fy = 400.00
self.studb1fu = 450.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 65.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 10.18*1000.0
self.studt1fy = 400.00
self.studt1fu = 450.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00 * 1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 100.00/1000.0
self.support1a = 1600.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 300.00/1000.0
self.load1a = 650.00/1000.0
self.load1b = 1050.00/1000.0
self.load1c = 1400.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = 1.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((850/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.045, 'b1', 4, (4)), (0.14, 'b1', 4, (4)), (0.235, 'b1', 4, (4)), (0.33, 'b1', 4, (4)),
(0.425, 'b1', 4, (4)), (0.52, 'b1', 4, (4)), (0.615, 'b1', 4, (4)), (0.71, 'b1', 4, (4)), (0.805, 'b1', 4, (4)), (0.895, 'b1', 4, (4)), (0.99, 'b1', 4, (4)), (1.085, 'b1', 4, (4)), (1.18, 'b1', 4, (4)), (1.275, 'b1', 4, (4)), (1.37, 'b1', 4, (4)), (1.465, 'b1', 4, (4)), (1.56, 'b1', 4, (4)), (1.655, 'b1', 4, (4))),)
self.studPositionDescriptions.extend(((0.045, 't1', 2, (2)), (0.125, 't1', 2, (2)), (0.225, 't1', 2, (2)), (0.325, 't1', 2, (2)), (0.425, 't1', 2, (2)), (0.525, 't1', 2, (2)), (0.625, 't1', 2, (2)), (0.725, 't1', 2, (2)), (0.825, 't1', 2, (2)), (0.925, 't1', 2, (2)), (1.025, 't1', 2, (2)), (1.125, 't1', 2, (2)), (1.225, 't1', 2, (2)), (1.325, 't1', 2, (2)), (1.425, 't1', 2, (2)), (1.525, 't1', 2, (2)), (1.625, 't1', 2, (2)), (1.725, 't1', 2, (2)), (1.825, 't1', 2, (2)),)
```

APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
    print self.Case_Name
    if self.Case_Name == 'R,16N_B3' or self.beamCaseNo==91:

        self.L = 1700.00/1000.0
        self.beamh = 162.12/1000.0
        self.beamb = 400.00/1000.0
        self.bottomt = 8.07/1000.0
        self.top = 4.05/1000.0

        self.plateFyBottom = 325.00
        self.plateFyTop = 224.00
        self.plateFuBottom = 375.00
        self.plateFuTop = 274.00
        self.concreteFc = 34.10

        self.studb1h = 150.00/1000.0
        self.studb1Dia = 10.00/1000.0
        self.studb1Resistance = 28.27*1000.0
        self.studb1fy = 400.00
        self.studb1fu = 450.00

        self.studb2h = 0.00/1000.0
        self.studb2Dia = 0.00/1000.0
        self.studb2Resistance = 0.00*1000.0
        self.studb2fy = 1.00
        self.studb2fu = 1.50

        self.studt1h = 65.00/1000.0
        self.studt1Dia = 6.00/1000.0
        self.studt1Resistance = 10.18*1000.0
        self.studt1fy = 400.00
        self.studt1fu = 450.00

        self.studt2h = 0.00/1000.0
        self.studt2Dia = 0.00/1000.0
        self.studt2Resistance = 0.00*1000.0
        self.studt2fy = 1.00
        self.studt2fu = 1.50

        self.CyclicLoad = 50.00*1000.0
        self.appliedDeflection = 0.05

        self.support1 = 100.00/1000.0
        self.support1a = 1600.00/1000.0
        self.support1b = -100.00/1000.0
        self.support1c = -100.00/1000.0

        self.load1 = 300.00/1000.0
        self.load1a = 650.00/1000.0
        self.load1b = 1050.00/1000.0
        self.load1c = 1400.00/1000.0
        self.load1d = -100.00/1000.0
        self.load1e = -100.00/1000.0
        self.load1f = -100.00/1000.0
        self.load1g = -100.00/1000.0

        self.loadFactor1 = 1.00
        self.loadFactor1a = 1.00
        self.loadFactor1b = 1.00
        self.loadFactor1c = 1.00
        self.loadFactor1d = -100.00
        self.loadFactor1e = -100.00
        self.loadFactor1f = -100.00
        self.loadFactor1g = -100.00

        self.loadDefPoints = []
```

479
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadDefPoints.extend(((850/1000, 0.0), ))

self.studyPositionDescriptions.extend(((0.045, 'b1', 2, (2)), (0.14, 'b1', 2, (2)), (0.235, 'b1', 2, (2)), (0.33, 'b1', 2, (2)), (0.425, 'b1', 2, (2)), (0.52, 'b1', 2, (2)), (0.615, 'b1', 2, (2)), (0.71, 'b1', 2, (2)), (0.805, 'b1', 2, (2)), (0.895, 'b1', 2, (2)), (0.99, 'b1', 2, (2)), (1.085, 'b1', 2, (2)), (1.175, 'b1', 2, (2)), (1.275, 'b1', 2, (2)), (1.37, 'b1', 2, (2)), (1.465, 'b1', 2, (2)), (1.56, 'b1', 2, (2)), (1.655, 'b1', 2, (2)), ))

self.studyPositionDescriptions.extend(((0.04, 't1', 4, (4)), (0.22, 't1', 4, (4)), (0.4, 't1', 4, (4)), (0.58, 't1', 4, (4)), (0.76, 't1', 4, (4)), (0.94, 't1', 4, (4)), (1.12, 't1', 4, (4)), (1.3, 't1', 4, (4)), (1.48, 't1', 4, (4)), (1.66, 't1', 4, (4)), ))

print self.Case_Name

if self.Case_Name == 'R,EBN_B4' or self.beamCaseNo==92:
    self.L = 1700.00/1000.0
    self.beamb = 166.05/1000.0
    self.beambottomt = 400.00/1000.0
    self.beambottom = 8.05/1000.0
    self.beambottom = 8.0/1000.0

    self.plateFyBottom = 202.00
    self.plateFyTop = 325.00
    self.plateFuBottom = 252.00
    self.plateFuTop = 375.00
    self.concretefc = 37.00

    self.studb1h = 150.00/1000.0
    self.studb1Dia = 10.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 28.27*1000.0
    self.studb1fy = 400.00
    self.studb1fu = 450.00

    self.studb2h = 0.0/1000.0
    self.studb2Dia = 0.0/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 0.00*1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50

    self.studt1h = 65.00/1000.0
    self.studt1Dia = 6.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 28.27*1000.0
    self.studt1fy = 400.00
    self.studt1fu = 450.00

    self.studt2h = 0.0/1000.0
    self.studt2Dia = 0.0/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00*1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50

    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05

    self.support1 = 100.00/1000.0
    self.support1a = 1600.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0

    self.load1 = 300.00/1000.0
    self.load1a = 650.00/1000.0
    self.load1b = 1050.00/1000.0
    self.load1c = 1400.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0

    self.loadFactor1 = 1.00
```

480
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1a = 1.00
self.loadFactor1b = 1.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((850/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.045, 'b1', 4, (4)), (0.14, 'b1', 4, (4)), (0.235, 'b1', 4, (4)), (0.33, 'b1', 4, (4)), (0.425, 'b1', 4, (4)), (0.52, 'b1', 4, (4)), (0.615, 'b1', 4, (4)), (0.71, 'b1', 4, (4)), (0.805, 'b1', 4, (4)), (0.895, 'b1', 4, (4)), (0.99, 'b1', 4, (4)), (1.085, 'b1', 4, (4)), (1.18, 'b1', 4, (4)), (1.275, 'b1', 4, (4)), (1.37, 'b1', 4, (4)), (1.465, 'b1', 4, (4)), (1.56, 'b1', 4, (4)), (1.655, 'b1', 4, (4)),))
self.studPositionDescriptions.extend(((0.045, 't1', 4, (4)), (0.13, 't1', 4, (4)), (0.215, 't1', 4, (4)), (0.295, 't1', 4, (4)), (0.385, 't1', 4, (4)), (0.47, 't1', 4, (4)), (0.555, 't1', 4, (4)), (0.64, 't1', 4, (4)), (0.725, 't1', 4, (4)), (0.81, 't1', 4, (4)), (0.89, 't1', 4, (4)), (0.975, 't1', 4, (4)), (1.06, 't1', 4, (4)), (1.145, 't1', 4, (4)), (1.23, 't1', 4, (4)), (1.315, 't1', 4, (4)), (1.4, 't1', 4, (4)), (1.485, 't1', 4, (4)), (1.57, 't1', 4, (4)), (1.655, 't1', 4, (4)),))

print self.Case_Name
if self.Case_Name == ('R&N_B5' or self.beamCaseNo == 93):

    self.L = 1700.00/1000.0
self.beamb = 162.15/1000.0
self.beamb = 400.00/1000.0
self.bottomt = 8.10/1000.0
self.top = 4.05/1000.0
self.plateFyBottom = 325.00
self.plateFyTop = 224.00
self.plateFullBottom = 375.00
self.plateTop = 274.00
self.concretef = 36.50
self.studb1h = 150.00/1000.0
self.studb1Dia = 10.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 10.18
self.studb1fy = 400.00
self.studb1fu = 450.00
self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00+1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50
self.studt1h = 65.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 10.18+1000.0
self.studt1fy = 400.00
self.studt1fu = 450.00
self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00+1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50
self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05
self.support1 = 100.00/1000.0
self.support1a = 1600.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
```

481
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

self.load1 = 300.00/1000.0
self.load1a = 650.00/1000.0
self.load1b = 1050.00/1000.0
self.load1c = 1400.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = 1.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((850/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.045,'b1',2,(2)),(0.14,'b1',2,(2)),(0.235,'b1',2,(2)),(0.33,'b1',2,(2)),(0.425,'b1',2,(2)),(0.52,'b1',2,(2)),(0.615,'b1',2,(2)),(0.71,'b1',2,(2)),(0.805,'b1',2,(2)),(0.895,'b1',2,(2)),(0.99,'b1',2,(2)),(1.085,'b1',2,(2)),(1.18,'b1',2,(2)),(1.275,'b1',2,(2)),(1.37,'b1',2,(2)),(1.465,'b1',2,(2)),(1.56,'b1',2,(2)),(1.655,'b1',2,(2)),))
self.studPositionDescriptions.extend(((0.04,'t1',4,(4)),(0.22,'t1',4,(4)),(0.4,'t1',4,(4)),(0.58,'t1',4,(4)),(0.76,'t1',4,(4)),(0.94,'t1',4,(4)),(1.12,'t1',4,(4)),(1.3,'t1',4,(4)),(1.48,'t1',4,(4)),(1.66,'t1',4,(4)),))

print self.Case_Name
if self.Case_Name == 'R,E&N_B6' or self.beamCaseNo==94:

self.L = 1700.00/1000.0
self.beamb = 162.10/1000.0
self.beamb = 400.00/1000.0
self.bottomt = 8.10/1000.0
self.topt = 4.00/1000.0

self.plateFyBottom = 325.00
self.plateFyTop = 224.00
self.plateFbTopBottom = 375.00
self.plateFbTop = 274.00
self.concreteste = 35.70

self.studh1h = 150.00/1000.0
self.studh1Dia = 10.00/1000.0
self.studh1FullHeight = False
self.studh1Resistance = 28.27+1000.0
self.studh1f = 400.00
self.studh1fu = 450.00

self.studh2h = 0.00/1000.0
self.studh2Dia = 0.00/1000.0
self.studh2FullHeight = False
self.studh2Resistance = 0.00+1000.0
self.studh2f = 1.00
self.studh2fu = 1.50

self.studh3h = 65.00/1000.0
self.studh3Dia = 6.00/1000.0
self.studh3FullHeight = False
self.studh3Resistance = 10.18+1000.0
self.studh3f = 400.00
self.studh3fu = 450.00

self.studh4h = 0.00/1000.0
self.studh4Dia = 0.00/1000.0
self.studh4FullHeight = False
self.studh4Resistance = 0.00+1000.0
self.studh4f = 1.00

482
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2fu = 1.50
self.CyclicLoad = 50.00/1000.0
self.appliedDeflection = 0.05

self.support1 = 100.00/1000.0
self.support1a = 1600.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 300.00/1000.0
self.load1a = 650.00/1000.0
self.load1b = 1050.00/1000.0
self.load1c = 1400.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = 1.00
self.loadFactor1c = 1.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((850/1000.0, 0.0),))

self.studPositionDescriptions.extend(((0.045, 'b1', 2, (2)), (0.14, 'b1', 2, (2)), (0.235, 'b1', 2, (2)), (0.33, 'b1', 2, (2)),
                                      (0.425, 'b1', 2, (2)), (0.52, 'b1', 2, (2)), (0.615, 'b1', 2, (2)), (0.71, 'b1', 2, (2)), (0.805, 'b1', 2, (2)), (0.895, 'b1', 2, (2)),
                                      (0.99, 'b1', 2, (2)), (1.085, 'b1', 2, (2)), (1.18, 'b1', 2, (2)), (1.275, 'b1', 2, (2)), (1.37, 'b1', 2, (2)), (1.465, 'b1', 2, (2)),
                                      (1.56, 'b1', 2, (2)), (1.655, 'b1', 2, (2)),))

self.studPositionDescriptions.extend(((0.04, 't1', 4, (4)), (0.22, 't1', 4, (4)), (0.4, 't1', 4, (4)), (0.58, 't1', 4, (4)),
                                      (0.76, 't1', 4, (4)), (0.94, 't1', 4, (4)), (1.12, 't1', 4, (4)), (1.3, 't1', 4, (4)), (1.48, 't1', 4, (4)), (1.66, 't1', 4, (4)),))

print self.Case_Name
if self.Case_Name == 'R,E&N_B9' or self.beamCaseNo==95:
    self.L = 3200.00/1000.0
    self.beambh = 166.09/1000.0
    self.beamb = 400.00/1000.0
    self.bottomt = 8.04/1000.0
    self.topt = 8.05/1000.0

self.plateFyBottom = 281.00
self.plateFyTop = 281.00
self.plateFuBottom = 331.00
self.plateFuTop = 331.00
self.concreteseal = 36.60

self.studb1h = 150.00/1000.0
self.studb1Dia = 10.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 28.27*1000.0
self.studb1fy = 400.00
self.studb1fu = 450.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00/1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 65.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1FullHeight = False
```

483
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studt1Resistance = 10.18*1000.0
self.studt1fy = 400.00
self.studt1fu = 450.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00+1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 100.00/1000.0
self.support1a = 3100.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 565.00/1000.0
self.load1a = 1225.00/1000.0
self.load1b = 1975.00/1000.0
self.load1c = 2635.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = 1.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints . extend(((1600/1000.0,0.0),))
self.loadPositionDescriptions.extend(((0.05, 'b1',4,(4)),(0.23, 'b1',4,(4)),(0.41, 'b1',4,(4)),(0.59, 'b1',4,(4))
                              ,(0.77, 'b1',4,(4)),(0.95, 'b1',4,(4)),(1.13, 'b1',4,(4)),(1.31, 'b1',4,(4)),(1.49, 'b1',4,(4))
                              ,(1.67, 'b1',4,(4)),(1.85, 'b1',4,(4)),(2.03, 'b1',4,(4)),(2.21, 'b1',4,(4)),(2.39, 'b1',4,(4))
                              ,(2.57, 'b1',4,(4)),(2.75, 'b1',4,(4)),(2.93, 'b1',4,(4)),(3.11, 'b1',4,(4)))

self.studPositionDescriptions.extend(((0.05, 't1',8,(8)),(0.21, 't1',8,(8)),(0.37, 't1',8,(8)),(0.53, 't1',8,(8))
                              ,(0.69, 't1',8,(8)),(0.85, 't1',8,(8)),(1.01, 't1',8,(8)),(1.17, 't1',8,(8)),(1.33, 't1',8,(8))
                              ,(1.49, 't1',8,(8)),(1.65, 't1',8,(8)),(1.81, 't1',8,(8)),(1.97, 't1',8,(8)),(2.13, 't1',8,(8))
                              ,(2.29, 't1',8,(8)),(2.45, 't1',8,(8)),(2.61, 't1',8,(8)),(2.77, 't1',8,(8)),(2.93, 't1',8,(8)))
                              ,(3.15, 't1',8,(8)))

print self.Case_Name
if self.Case_Name == 'R,E&N_B10' or self.beamCaseNo==96:

self.L = 3200.00/1000.0
self.beamh = 162.02/1000.0
self.beamb = 400.00/1000.0
self.bottomt = 8.05/1000.0
self.topt = 3.97/1000.0

self.plateFyBottom = 281.00
self.plateFuTop = 358.00
self.plateFuBottom = 531.00
self.plateFuTop = 408.00
self.concretesc = 33.20

self.studh1 = 150.00/1000.0
self.stud1Dia = 10.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 28.27+1000.0
self.stud1fy = 400.00
self.stud1fu = 450.00
```
self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00∗1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.stud1h = 65.00/1000.0
self.stud1Dia = 6.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 10.18+1000.0
self.stud1fy = 400.00
self.stud1fu = 450.00

self.CyclicLoad = 50.00∗1000.0
self.appliedDeflection = 0.05
self.support1 = 100.00/1000.0
self.support1a = 3100.00/1000.0
self.support1b = −100.00/1000.0
self.support1c = −100.00/1000.0
self.load1 = 1225.00/1000.0
self.load1a = 1975.00/1000.0
self.load1b = −100.00/1000.0
self.load1c = −100.00/1000.0
self.load1d = −100.00/1000.0
self.load1e = −100.00/1000.0
self.load1f = −100.00/1000.0
self.load1g = −100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = −100.00
self.loadFactor1c = −100.00
self.loadFactor1d = −100.00
self.loadFactor1e = −100.00
self.loadFactor1f = −100.00
self.loadFactor1g = −100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((1600/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.05, 'b1', 4,(4)), (0.23, 'b1', 4,(4)), (0.41, 'b1', 4,(4)), (0.59, 'b1', 4,(4)), (0.77, 'b1', 4,(4)), (0.95, 'b1', 4,(4)), (1.13, 'b1', 4,(4)), (1.31, 'b1', 4,(4)), (1.49, 'b1', 4,(4)), (1.71, 'b1', 4,(4)), (1.89, 'b1', 4,(4)), (2.07, 'b1', 4,(4)), (2.25, 'b1', 4,(4)), (2.43, 'b1', 4,(4)), (2.61, 'b1', 4,(4)), (2.79, 'b1', 4,(4)), (2.97, 'b1', 4,(4)), (3.15, 'b1', 4,(4)),))
self.studPositionDescriptions.extend(((0.05, 't1', 2,(2)), (0.41, 't1', 2,(2)), (0.77, 't1', 2,(2)), (1.13, 't1', 2,(2)), (1.49, 't1', 2,(2)), (1.71, 't1', 2,(2)), (2.07, 't1', 2,(2)), (2.35, 't1', 2,(2)), (2.61, 't1', 2,(2)), (2.87, 't1', 2,(2)), (3.15, 't1', 2,(2)),))

print self.Case_Name
if self.Case_Name == 'R_E&N_B11' or self.beamCaseNo==97:
    self.L = 3200.00/1000.0
    self.beamh = 161.97/1000.0
    self.beamb = 400.00/1000.0
    self.bottomt = 7.98/1000.0
    self.topb = 3.99/1000.0
    self.plateFyBottom = 281.00
    self.plateFyTop = 358.00
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.plateFuBottom = 331.00
self.plateFuTop = 408.00
self.concretefc = 35.20

self.studb1h = 150.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 28.27*1000.0
self.studb1fy = 400.00
self.studb1fu = 450.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 65.00/1000.0
self.studt1Dia = 6.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 10.18*1000.0
self.studt1fy = 400.00
self.studt1fu = 450.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 100.00/1000.0
self.support1a = 3100.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1225.00/1000.0
self.load1a = 1975.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1600/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.06,'b1',4,(4)),(0.445,'b1',4,(4)),(0.83,'b1',4,(4)),(0.215,'b1',4,(4)),(1.6,'b1',4,(4)),(1.985,'b1',4,(4)),(2.37,'b1',4,(4)),(2.755,'b1',4,(4)),(3.14,'b1',4,(4)),))

self.studPositionDescriptions.extend(((0.05,'t1',2,(2)),(0.21,'t1',2,(2)),(0.37,'t1',2,(2)),(0.53,'t1',2,(2)),(0.69,'t1',2,(2)),(0.85,'t1',2,(2)),(1.01,'t1',2,(2)),(1.17,'t1',2,(2)),(1.33,'t1',2,(2)),(1.49,'t1',2,(2)),(1.65,'t1',2,(2)),(1.81,'t1',2,(2)),(2.03,'t1',2,(2)),(2.19,'t1',2,(2)),(2.35,'t1',2,(2)),(2.51,'t1',2,(2)),(2.67,'t1',2,(2)),(2.83,'t1',2,(2)),(2.99,'t1',2,(2)),(3.15,'t1',2,(2)),))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BSI' or self.beamCaseNo==98:
```

486
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.L = 1600.00/1000.0
self.beamb = 412.40/1000.0
self.bottomt = 6.20/1000.0
self.topt = 6.20/1000.0

self.plateFyBottom = 384.00
self.plateFyTop = 384.00
self.plateFuBottom = 507.00
self.plateFuTop = 507.00
self.concretefc = 58.00

self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 400.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 222.27+1000.0
self.studb2fy = 541.00
self.studb2fu = 566.00

self.studt1h = 0.00/1000.0
self.studt1Dia = 0.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 0.00+1000.0
self.studt1fy = 1.00
self.studt1fu = 1.50

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00+1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 1400.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 800.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((800/1000.0,0.0),))
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studPositionDescriptions.extend(((0.2, 'b2', 2, (2)), (0.4, 'b2', 2, (2)), (0.6, 'b2', 2, (2)), (0.8, 'b2', 2, (2)), (1.0, 'b2', 2, (2)), (1.2, 'b2', 2, (2)), (1.4, 'b2', 2, (2))))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS2' or self.beamCaseNo==99:
    self.L = 1600.00/1000.0
    self.beamb = 415.80/1000.0
    self.beamb = 400.00/1000.0
    self.botmt = 7.99/1000.0
    self.topt = 7.90/1000.0
    self.plateFyBottom = 381.00
    self.plateFyTop = 381.00
    self.plateFuBottom = 518.00
    self.plateFuTop = 518.00
    self.concretefc = 58.00
    self.stub1h = 0.00/1000.0
    self.stub1Dia = 0.00/1000.0
    self.stub1FullHeight = False
    self.stub1Resistance = 0.00+1000.0
    self.stub1fy = 1.00
    self.stub1fu = 1.50
    self.stub2h = 400.00/1000.0
    self.stub2Dia = 25.00/1000.0
    self.stub2FullHeight = True
    self.stub2Resistance = 222.27+1000.0
    self.stub2fy = 541.00
    self.stub2fu = 566.00
    self.stud1h = 0.00/1000.0
    self.stud1Dia = 0.00/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 0.00+1000.0
    self.stud1fy = 1.00
    self.stud1fu = 1.50
    self.stud2h = 0.00/1000.0
    self.stud2Dia = 0.00/1000.0
    self.stud2FullHeight = False
    self.stud2Resistance = 0.00+1000.0
    self.stud2fy = 1.00
    self.stud2fu = 1.50
    self.CyclicLoad = 50.00×1000.0
    self.appliedDeflection = 0.05
    self.support1 = 200.00/1000.0
    self.support1a = 1400.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 600.00/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = -100.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
    self.loadFactor1d = -100.00
    self.loadFactor1e = -100.00
    self.loadFactor1f = -100.00
```

488
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((800/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.2, 'b2', 2, (2)), (0.4, 'b2', 2, (2)), (0.6, 'b2', 2, (2)), (0.8, 'b2', 2, (2)), (1.2, 'b2', 2, (2)), (1.4, 'b2', 2, (2)),))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS4' or self.beamCaseNo==101:
    self.L = 1600.00/1000.0
    self.beamb = 218.00/1000.0
    self.bottomt = 6.10/1000.0
    self.topt = 11.90/1000.0
    self.plateFyBottom = 384.00
    self.plateFyTop = 419.00
    self.plateFuBottom = 507.00
    self.plateFuTop = 563.00
    self.concretefc = 58.00
    self.studb1h = 0.00/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00+1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50
    self.studb2h = 200.00/1000.0
    self.studb2Dia = 25.00/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 222.27+1000.0
    self.studb2fy = 541.00
    self.studb2fu = 566.00
    self.studt1h = 0.00/1000.0
    self.studt1Dia = 0.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 0.00+1000.0
    self.studt1fy = 1.00
    self.studt1fu = 1.50
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00+1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05
    self.supportl = 200.00/1000.0
    self.supportla = 1400.00/1000.0
    self.supportlb = -100.00/1000.0
    self.supportlc = -100.00/1000.0
    self.load1 = 800.00/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1l = 1.00
    self.loadFactor1a = -100.00
```

APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((800/1000.0, 0.0),))

self.studPositionDescriptions.extend(((0.2, 'b2', 2,(2)), (0.4, 'b2', 2,(2)), (0.6, 'b2', 2,(2)), (0.8, 'b2', 2,(2)), (1, 'b2', 2,(2)), (1.2, 'b2', 2,(2)), (1.4, 'b2', 2,(2)),))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS5' or self.beamCaseNo==102:
    self.L = 1600.00/1000.0
    self.beamh = 219.81/1000.0
    self.beamb = 480.00/1000.0
    self.bottomt = 7.91/1000.0
    self.topt = 11.90/1000.0
    self.plateFyBottom = 381.00
    self.plateFyTop = 419.00
    self.plateFuBottom = 518.00
    self.plateFuTop = 563.00
    self.concretefc = 58.00
    self.studb1h = 0.00/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00*1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50
    self.studb2h = 200.00/1000.0
    self.studb2Dia = 25.00/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 222.27+1000.0
    self.studb2fy = 541.00
    self.studb2fu = 566.00
    self.studt1h = 0.00/1000.0
    self.studt1Dia = 0.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 0.00+1000.0
    self.studt1fy = 1.00
    self.studt1fu = 1.50
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00*1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05
    self.support1 = 200.00/1000.0
    self.support1a = 1400.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 800.00/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
```

APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((800/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.2,'b2',2,(2)),(0.4,'b2',2,(2)),(0.6,'b2',2,(2)),(0.8,'b2',2,(2)),(1,'b2',2,(2)),(1.2,'b2',2,(2)),(1.4,'b2',2,(2)),))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS6' or self.beamCaseNo==103:
    self.L = 1600.00/1000.0
    self.beamb = 223.63/1000.0
    self.beamb = 480.00/1000.0
    self.bottomt = 11.83/1000.0
    self.topt = 11.80/1000.0

    self.plateFyBottom = 419.00
    self.plateFyTop = 419.00
    self.plateFuBottom = 563.00
    self.plateFuTop = 563.00
    self.concretefc = 58.00

    self.studb1h = 0.08/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00+1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50

    self.studb2h = 200.00/1000.0
    self.studb2Dia = 25.00/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 222.27+1000.0
    self.studb2fy = 541.00
    self.studb2fu = 566.00

    self.studt1h = 0.08/1000.0
    self.studt1Dia = 0.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 0.00+1000.0
    self.studt1fy = 1.00
    self.studt1fu = 1.50

    self.studt2h = 0.08/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00+1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50

    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05

    self.support1 = 200.00/1000.0
    self.support2a = 1400.00/1000.0
    self.supportb = -100.00/1000.0
    self.supportc = -100.00/1000.0

    self.load1 = 800.00/1000.0
```

491
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((800/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.2, 'b2', 2,(2)),(0.4, 'b2', 2,(2)),(0.6, 'b2', 2,(2)),(0.8, 'b2', 2,(2)),(1, 'b2', 2,(2)),(1.2, 'b2', 2,(2)),(1.4, 'b2', 2,(2)),))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS7' or self.beamCaseNo==184:

self.L = 2200.00/1000.0
self.beamb = 218.10/1000.0
self.bottomt = 6.20/1000.0
self.topt = 11.90/1000.0
self.plateFyBottom = 384.00
self.plateFyTop = 419.00
self.plateFuTop = 563.00
self.concretefc = 58.00
self.stud1h = 0.00/1000.0
self.stud1Dia = 0.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 0.00+1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50
self.stud2h = 200.00/1000.0
self.stud2Dia = 25.00/1000.0
self.stud2FullHeight = True
self.stud2Resistance = 222.27+1000.0
self.stud2fy = 541.00
self.stud2fu = 566.00
self.stud1h = 0.00/1000.0
self.stud1Dia = 0.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 0.00+1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50
self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05
self.support1 = 200.00/1000.0
```

492
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.support1a = 2000.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1100.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.load1 = 1100.00/1000.0
self.load1a = -100.00
self.load1b = -100.00
self.load1c = -100.00
self.load1d = -100.00
self.load1e = -100.00
self.load1f = -100.00
self.load1g = -100.00

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1100/1000,0.0),))

self.studPositionDescriptions.extend(((2.,'b2',2,(2)),(2.,'b2',2,(2)),(2.,'b2',2,(2)),(1.1,'b2',2,(2)),(1.4,'b2',2,(2)),(1.7,'b2',2,(2)),(2.,'b2',2,(2)),))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS8' or self.beamCaseNo==105:
    self.L = 2200.00/1000.0
    self.beamh = 219.78/1000.0
    self.beamb = 400.00/1000.0
    self.bottomt = 7.98/1000.0
    self.topt = 11.80/1000.0
    self.plateFyBottom = 381.00
    self.plateFyTop = 419.00
    self.plateFullBottom = 518.00
    self.plateFullTop = 563.00
    self.concretefc = 58.00
    self.studb1h = 0.09/1000.0
    self.studb1Dia = 0.09/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00+1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50
    self.studb2h = 200.00/1000.0
    self.studb2Dia = 25.00/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 222.27+1000.0
    self.studb2fy = 541.00
    self.studb2fu = 566.00
    self.stud1h = 0.09/1000.0
    self.stud1Dia = 0.09/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 0.00+1000.0
    self.stud1fy = 1.00
    self.stud1fu = 1.50
    self.stud2h = 0.09/1000.0
    self.stud2Dia = 0.09/1000.0
    self.stud2FullHeight = False
    self.stud2Resistance = 0.00+1000.0
    self.stud2fy = 1.00
    self.stud2fu = 1.50

493
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2000.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1100.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1100/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.2, 'b2', 2, (2)), (0.5, 'b2', 2, (2)), (0.8, 'b2', 2, (2)), (1.1, 'b2', 2, (2)), (1.4, 'b2', 2, (2)), (1.7, 'b2', 2, (2)), (2, 'b2', 2, (2)),))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS9' or self.beamCaseNo==106:

    self.L = 2200.00/1000.0
    self.beamh = 223.80/1000.0
    self.beamb = 400.00/1000.0
    self.bottomt = 11.90/1000.0
    self.topt = 11.90/1000.0

    self.plateFyBottom = 419.00
    self.plateFyTop = 419.00
    self.plateFuBottom = 563.00
    self.plateFuTop = 563.00
    self.concretefc = 58.00

    self.studb1h = 0.08/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00*1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50

    self.studb2h = 200.00/1000.0
    self.studb2Dia = 25.00/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 222.27+1000.0
    self.studb2fy = 541.00
    self.studb2fu = 566.00

    self.studt1h = 0.08/1000.0
    self.studt1Dia = 0.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 0.00+1000.0
    self.studt1fy = 1.00
    self.studt1fu = 1.50

    self.studt2h = 0.08/1000.0
```

494
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2000.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1100.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1100/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.2, 'b2', 2,(2) ),(0.5, 'b2', 2,(2) ),(0.8, 'b2', 2,(2) ),(1.1, 'b2', 2,(2) ),(1.4, 'b2', 2,(2) ),(1.7, 'b2', 2,(2) ),(2, 'b2', 2,(2) )))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS10' or self.beamCaseNo==107:

self.L = 2800.00/1000.0
self.beamh = 218.10/1000.0
self.beamb = 400.00/1000.0
self.bottomt = 6.10/1000.0
self.topt = 12.00/1000.0

self.plateFyBottom = 384.00
self.plateFyTop = 419.00
self.plateFuBottom = 507.00
self.plateFuTop = 563.00

self.concretef = 58.00

self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00+1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 0.00/1000.0
self.studt1Dia = 0.00/1000.0
self.studt1FullHeight = False
```

495
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studt1Resistance = 0.00*1000.0
self.studt1fy = 1.00
self.studt1fu = 1.50

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00+1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2600.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1400.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1400/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.2,'b2',2),(0.6,'b2',2),(1.2,'b2',2),(1.8,'b2',2),(2.2,'b2',2),(2.6,'b2',2),(2.8,'b2',2),))

print self.Case_Name
if self.Case_Name == 'Foundoudos_BS11' or self.beamCaseNo==108:

self.L = 2800.00/1000.0
self.beamh = 219.88/1000.0
self.beamb = 400.00/1000.0
self.bottomt = 7.98/1000.0
self.topt = 11.90/1000.0

self.plateFyBottom = 381.00
self.plateFyTop = 419.00
self.plateFuBottom = 518.00
self.plateFuTop = 563.00
self.concretefc = 58.00

self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 222.27+1000.0
self.studb2fy = 541.00
```

496
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2fu = 566.00
self.studth = 0.00/1000.0
self.stud1Dia = 0.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 0.00+1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50
self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50
self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05
self.support1 = 200.00/1000.0
self.support1a = 2600.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 1400.00/1000.0
self.loadla = -100.00/1000.0
self.loadlb = -100.00/1000.0
self.loadlc = -100.00/1000.0
self.loadld = -100.00/1000.0
self.loadle = -100.00/1000.0
self.loadlf = -100.00/1000.0
self.loadlg = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend((1400/1000.0, 0.0, 0.0))
self.studPositionDescriptions.extend(((0.2, 'b2', 2, (2)), (0.6, 'b2', 2, (2)), (1.0, 'b2', 2, (2)), (1.4, 'b2', 2, (2)), (1.8, 'b2', 2, (2)), (2.2, 'b2', 2, (2)), (2.6, 'b2', 2, (2))))

print self.Case_Name
if self.Case_Name == 'Foundoukos_RS12' or self.beamCaseNo==109:
    self.L = 2800.00/1000.0
    self.beamh = 223.82/1000.0
    self.beamb = 400.00/1000.0
    self.bottomt = 11.92/1000.0
    self.topt = 11.90/1000.0
    self.plateFyBottom = 419.00
    self.plateFyTop = 419.00
    self.plateFuBottom = 563.00
    self.plateFuTop = 563.00
    self.concretefc = 58.00
    self.studth = 0.00/1000.0
    self.studbDia = 0.00/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 0.00+1000.0
    self.stud1fy = 1.00
    self.stud1fu = 1.50
```
APPENDIX B: Test database in Python code format

```python
self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 222.27+1000.0
self.studb2fy = 541.00
self.studb2fu = 566.00

self.stud1h = 0.08/1000.0
self.stud1Dia = 0.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 0.00+1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50

self.stud2h = 0.08/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2600.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1400.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1400/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.2,'b2',2,(2)),(0.6,'b2',2,(2)),(1,'b2',2,(2)),(1.4,'b2',2,(2)),(1.8,'b2',2,(2)),(2.2,'b2',2,(2)),(2.6,'b2',2,(2)),(2.8,'b2',2,(2))))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS13' or self.beamCaseNo==110:

self.L = 2000.00/1000.0
self.beamh = 320.50/1000.0
self.beamb = 400.00/1000.0
self.bottomt = 10.30/1000.0
self.top = 10.20/1000.0

self.plateFyBottom = 430.00
self.plateFyTop = 430.00
self.plateFuBottom = 548.00
self.plateFuTop = 548.00
self.concretefc = 40.00

self.stud1h = 0.08/1000.0
self.stud1Dia = 0.00/1000.0
```

498
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb1FullHeight = False
self.studb1Resistance = 0.00*1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 300.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 209.33*1000.0
self.studb2fy = 553.00
self.studb2fu = 586.00

self.studt1h = 0.00/1000.0
self.studt1Dia = 0.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 0.00
self.studt1fy = 1.00
self.studt1fu = 1.50

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.supportla = 1800.00/1000.0
self.supportlb = -100.00/1000.0
self.supportlc = -100.00/1000.0

self.load1 = 1000.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1000/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.2,'b2',2,(2)),(0.4,'b2',2,(2)),(0.6,'b2',2,(2)),(0.8,'b2',2,(2)),(1.0,'b2',2,(2)),(1.2,'b2',2,(2)),(1.4,'b2',2,(2)),(1.6,'b2',2,(2)),(1.8,'b2',2,(2)),))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS14' or self.beamCaseNo==111:

self.L = 2000.00/1000.0
self.beamb = 323.71/1000.0
self.beamb = 400.00/1000.0
self.bottomt = 11.88/1000.0
self.topt = 11.83/1000.0

self.plateFyBottom = 431.00
self.plateFyTop = 431.00
self.plateFuBottom = 571.00
```

499
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.plateFuTop = 571.00
self.concretefc = 40.00

self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 300.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 209.33+1000.0
self.studb2fy = 553.00
self.studb2fu = 586.00

self.studt1h = 0.00/1000.0
self.studt1Dia = 0.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 0.00+1000.0
self.studt1fy = 1.00
self.studt1fu = 1.50

self.studt2h = 0.00/1000.0
self.studt2Dia = 25.00/1000.0
self.studt2FullHeight = True
self.studt2Resistance = 209.33+1000.0
self.studt2fy = 553.00
self.studt2fu = 586.00

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 1800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1000.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1000/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.2, 'b2', 2,(2)),(0.4, 'b2', 2,(2)),(0.6, 'b2', 2,(2)),(0.8, 'b2', 2,(2)),(1.0, 'b2', 2,(2)),(1.2, 'b2', 2,(2)),(1.4, 'b2', 2,(2)),(1.6, 'b2', 2,(2)),(1.8, 'b2', 2,(2)),))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BES15' or self.beamCaseNo==112:

self.L = 2200.00/1000.0
self.beamb = 320.29/1000.0
self.beamb = 400.00/1000.0
self.bottomt = 10.02/1000.0
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.topt = 10.27/1000.0
self.plateFyBottom = 430.00
self.plateFyTop = 430.00
self.plateFuBottom = 548.00
self.plateFuTop = 548.00
self.concretefc = 40.00

self.stud1h = 0.08/1000.0
self.stud1Dia = 0.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 0.00+1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50

self.stud2h = 300.00/1000.0
self.stud2Dia = 25.00/1000.0
self.stud2FullHeight = True
self.stud2Resistance = 209.33+1000.0
self.stud2fy = 553.00
self.stud2fu = 586.00

self.stud1h = 0.08/1000.0
self.stud1Dia = 0.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 0.00+1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50

self.stud2h = 0.08/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2000.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1100.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1100/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.2, 'b2',2,(2)) , (0.5, 'b2',2,(2)) , (0.8, 'b2',2,(2)) , (1.1, 'b2',2,(2)) , (1.4, 'b2',2,(2)) , (1.7, 'b2',2,(2)) , (2, 'b2',2,(2)) ,))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS16' or self.beamCaseNo==113:
```
```
```
APPENDIX B: Test database in Python code format

```python
self.L = 2200.00/1000.0
self.beamb = 323.59/1000.0
self.bottomt = 11.76/1000.0
self.top = 11.85/1000.0

self.plateFyBottom = 431.00
self.plateFyTop = 431.00
self.plateFuBottom = 571.00
self.plateFuTop = 571.00
self.concretefc = 40.00

self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00*1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 300.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 209.33+1000.0
self.studb2fy = 553.00
self.studb2fu = 586.00

self.studt1h = 0.00/1000.0
self.studt1Dia = 0.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 0.00*1000.0
self.studt1fy = 1.00
self.studt1fu = 1.50

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2000.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1100.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1100/1000.0,0.0),))
```

502
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studPositionDescriptions.extend(((0.2, 'b2', 2, (2)), (0.5, 'b2', 2, (2)), (0.8, 'b2', 2, (2)), (1.1, 'b2', 2, (2)), (1.4, 'b2', 2, (2)), (1.7, 'b2', 2, (2)), (2, 'b2', 2, (2)),))

print self.Case_Name
if self.Case_Name == 'Foundoukos_BS17' or self.beamCaseNo==114:
    self.L = 2800.00/1000.0
    self.beamb = 320.39/1000.0
    self.beamb = 320.39/1000.0
    self.bottomt = 10.18/1000.0
    self.topt = 10.21/1000.0
    self.plateFyBottom = 430.00
    self.plateFyTop = 430.00
    self.plateFuBottom = 548.00
    self.plateFuTop = 548.00
    self.concretefc = 40.00
    self.studb1h = 0.00/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00+1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50
    self.studb2h = 300.00/1000.0
    self.studb2Dia = 25.00/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 209.33+1000.0
    self.studb2fy = 553.00
    self.studb2fu = 586.00
    self.studt1h = 0.08/1000.0
    self.studt1Dia = 0.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 0.00+1000.0
    self.studt1fy = 1.00
    self.studt1fu = 1.50
    self.studt2h = 0.08/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00+1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05
    self.support1 = 200.00/1000.0
    self.support1a = 200.00/1000.0
    self.support1b = 200.00/1000.0
    self.support1c = -100.00/1000.0
    self.support1d = -100.00/1000.0
    self.load1 = 1400.00/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = -100.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
    self.loadFactor1d = -100.00
    self.loadFactor1e = -100.00
    self.loadFactor1f = -100.00
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((1400/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.2, 'b2', 2,(2)), (0.5, 'b2', 2,(2)), (0.8, 'b2', 2,(2)), (1.1, 'b2', 2,(2)), (1.4, 'b2', 2,(2)), (1.7, 'b2', 2,(2)), (2, 'b2', 2,(2)), (2.3, 'b2', 2,(2)), (2.6, 'b2', 2,(2)),))
print self.Case_Name
if self.Case_Name == 'Foundoukos_BS18' or self.beamCaseNo==115:
    self.L = 2800.00/1000.0
    self.beamb = 323.78/1000.0
    self.bottomt = 11.89/1000.0
    self.top = 11.89/1000.0
    self.plateFyBottom = 431.00
    self.plateFyTop = 431.00
    self.plateFuBottom = 571.00
    self.plateFuTop = 571.00
    self.concretefc = 40.00
    self.studb1h = 0.00/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00+1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50
    self.studb2h = 300.00/1000.0
    self.studb2Dia = 25.00/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 209.33+1000.0
    self.studb2fy = 553.00
    self.studb2fu = 586.00
    self.studt1h = 0.00/1000.0
    self.studt1Dia = 0.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 0.00+1000.0
    self.studt1fy = 1.00
    self.studt1fu = 1.50
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00+1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05
    self.support1 = 200.00/1000.0
    self.support1a = 2600.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 1400.00/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = -100.00

```

504
**APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT**

```python
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1400/1000.0, 0.0),))

self.studPositionDescriptions.extend(((0.2, 'b2', 2, (2)), (0.5, 'b2', 2, (2)), (0.0, 'b2', 2, (2)), (1.1, 'b2', 2, (2)), (1.4, 'b2', 2, (2)), (1.7, 'b2', 2, (2)), (2, 'b2', 2, (2)), (2.3, 'b2', 2, (2)), (2.6, 'b2', 2, (2)),))

print self.Case_Name
if self.Case_Name == 'sMiRT_SC1 north' or self.beamCaseNo==194:
    self.L = 4572.00/1000.0
    self.beamh = 406.00/1000.0
    self.beamb = 304.00/1000.0
    self.bottomt = 4.76/1000.0
    self.topt = 4.76/1000.0
    self.plateFyBottom = 379.00
    self.plateFyTop = 379.00
    self.plateFuBottom = 429.00
    self.plateFuTop = 429.00
    self.concretef = 56.02

self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 203.00/1000.0
self.studb2Dia = 7.07/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 14.74+1000.0
self.studb2fy = 419.00
self.studb2fu = 469.00

self.studth = 0.08/1000.0
self.stud1Dia = 0.08/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 0.00+1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50

self.stud2h = 0.08/1000.0
self.stud2Dia = 0.08/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 203.20/1000.0
self.support1a = 4368.80/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1218.20/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
```

505
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadIf = -100.00/1000.0
self.loadIg = -100.00/1000.0

self.loadFactorI = 1.00
self.loadFactorIa = -100.00
self.loadFactorIb = -100.00
self.loadFactorId = -100.00
self.loadFactorIe = -100.00
self.loadFactorIf = -100.00
self.loadFactorIg = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1218.2/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.053,'b2',2),(0.256,'b2',2),(0.459,'b2',2),(0.662,'b2',2),(0.865,'b2',2),(1.068,'b2',2),(1.271,'b2',2),(1.474,'b2',2),(1.677,'b2',2),(1.880,'b2',2),(2.083,'b2',2),(2.286,'b2',2),(2.489,'b2',2),(2.692,'b2',2),(2.895,'b2',2),(3.098,'b2',2),(3.301,'b2',2),(3.504,'b2',2),(3.707,'b2',2),(3.910,'b2',2),(4.113,'b2',2),(4.316,'b2',2),(4.519,'b2',2),(4.722,'b2',2),))

print self.Case_Name
if self.Case_Name == 'sMiRT_SC1 south' or self.beamCaseNo==195:
    self.L = 4572.00/1000.0
    self.beamb = 406.00/1000.0
    self.beamb = 304.00/1000.0
    self.bottomt = 4.76/1000.0
    self.top = 4.76/1000.0
    self.plateFyBottom = 379.00
    self.plateFyTop = 379.00
    self.plateFuBottom = 429.00
    self.plateFuTop = 429.00
    self.concretefc = 56.02
    self.studb1h = 0.00/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00+1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50
    self.studb2h = 203.00/1000.0
    self.studb2Dia = 7.07/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 14.74+1000.0
    self.studb2fy = 419.00
    self.studb2fu = 469.00
    self.studt1h = 0.00/1000.0
    self.studt1Dia = 0.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 0.00+1000.0
    self.studt1fy = 1.00
    self.studt1fu = 1.50
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00+1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05
    self.support1 = 203.20/1000.0
    self.support1a = 4368.80/1000.0
    self.support1b = -100.00/1000.0
```
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.support1c = -100.0/1000.0
self.load1 = 1218.20/1000.0
self.load1b = -100.0/1000.0
self.load1c = -100.0/1000.0
self.load1d = -100.0/1000.0
self.load1e = -100.0/1000.0
self.load1f = -100.0/1000.0
self.load1g = -100.0/1000.0
self.loadFactor1 = 1.00
self.loadFactor1b = -100.0
self.loadFactor1c = -100.0
self.loadFactor1d = -100.0
self.loadFactor1e = -100.0
self.loadFactor1f = -100.0
self.loadFactor1g = -100.0
self.loadDefPoints = []
self.loadDefPoints.extend(((1218.2/1000.0,0.0),))
self.studPositionDescriptions.extend([(0.053,'b2',2),(0.256,'b2',2),(0.459,'b2',2),(0.662,'b2',2),(1.068,'b2',2),(1.271,'b2',2),(1.474,'b2',2),(1.677,'b2',2),(1.88,'b2',2),(2.083,'b2',2),(2.286,'b2',2),(2.489,'b2',2),(2.692,'b2',2),(2.895,'b2',2),
(3.098,'b2',2),(3.301,'b2',2),(3.504,'b2',2),(3.707,'b2',2),(3.91,'b2',2),(4.113,'b2',2),(4.316,'b2',2),(4.519,'b2',2),(4.722,'b2',2),
(4.925,'b2',2),(5.128,'b2',2),(5.331,'b2',2),(5.534,'b2',2),(5.737,'b2',2),(5.940,'b2',2),(6.143,'b2',2),(6.346,'b2',2),(6.549,'b2',2),
(6.752,'b2',2),(6.955,'b2',2),(7.158,'b2',2),(7.361,'b2',2),(7.564,'b2',2),(7.767,'b2',2),(7.970,'b2',2),(8.173,'b2',2),(8.376,'b2',2),

print self.Case_Name
if self.Case_Name == 'sMRT_SC2 south' or self.beamCaseNo==196:
    self.L = 4572.00/1000.0
    self.beamb = 406.00/1000.0
    self.bottomt = 4.76/1000.0
    self.topt = 4.76/1000.0
    self.plateFyBottom = 379.00
    self.plateFyTop = 379.00
    self.plateFyBottom = 429.00
    self.plateFuBottom = 429.00
    self.concretefc = 39.96
    self.studb1h = 0.00/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00*1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50
    self.studb2h = 203.00/1000.0
    self.studb2Dia = 6.61/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 12.89*1000.0
    self.studb2fy = 419.00
    self.studb2fu = 469.00
    self.studt1h = 0.00/1000.0
    self.studt1Dia = 0.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 0.00*1000.0
    self.studt1fy = 1.00
    self.studt1fu = 1.50
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 203.20/1000.0
self.support1a = 4368.80/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1218.20/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((1218.2/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.063,'b2',2,2),(0.241,'b2',2,2),(0.419,'b2',2,2),(0.597,'b2',2,2),(0.775,'b2',2,2),(0.953,'b2',2,2),(1.131,'b2',2,2),(1.309,'b2',2,2),(1.487,'b2',2,2),(1.665,'b2',2,2),(1.843,'b2',2,2),(2.021,'b2',2,2),(2.199,'b2',2,2),(2.377,'b2',2,2),(2.555,'b2',2,2),(2.733,'b2',2,2),(2.911,'b2',2,2),(3.089,'b2',2,2),(3.267,'b2',2,2),(3.445,'b2',2,2),(3.623,'b2',2,2),(3.801,'b2',2,2),(3.979,'b2',2,2),(4.157,'b2',2,2),(4.335,'b2',2,2),(4.513,'b2',2,2),))

print self.Case_Name
if self.Case_Name == 'SMART_SC3 north' or self.beamCaseNo==197:

    self.L = 4572.00/1000.0
    self.beamh = 406.00/1000.0
    self.beamb = 304.00/1000.0
    self.bottomt = 4.76/1000.0
    self.top = 4.76/1000.0

    self.plateFyBottom = 379.00
    self.plateFyTop = 379.00
    self.plateFyBottom = 429.00
    self.plateFyTop = 429.00
    self.concretefc = 40.10

    self.studb1h = 0.00/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00*1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50

    self.studb2h = 203.00/1000.0
    self.studb2Dia = 6.12/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 11.05*1000.0
    self.studb2fy = 419.00
    self.studb2fu = 469.00

    self.studt1h = 0.00/1000.0
    self.studt1Dia = 0.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 0.00+1000.0
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studt1fy = 1.00
self.studt1fu = 1.50

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00+1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 203.20/1000.0
self.support1a = 4368.80/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 1218.20/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((1218.2/1000.0,0.0),))
self.loadPositionDescriptions.extend(((0.082,'b2',2,(2)),(0.234,'b2',2,(2)),(0.386,'b2',2,(2)),(0.538,'b2',2,(2)),(0.69,'b2',2,(2)),(0.842,'b2',2,(2)),(0.994,'b2',2,(2)),(1.146,'b2',2,(2)),(1.298,'b2',2,(2)),(1.45,'b2',2,(2)),(1.602,'b2',2,(2)),(1.754,'b2',2,(2)),(1.906,'b2',2,(2)),(2.058,'b2',2,(2)),(2.21,'b2',2,(2)),(2.362,'b2',2,(2)),(2.514,'b2',2,(2)),(2.666,'b2',2,(2)),(2.818,'b2',2,(2)),(2.97,'b2',2,(2)),(3.122,'b2',2,(2)),(3.274,'b2',2,(2)),(3.426,'b2',2,(2)),(3.578,'b2',2,(2)),(3.73,'b2',2,(2)),(3.882,'b2',2,(2)),(4.034,'b2',2,(2)),(4.186,'b2',2,(2)),(4.338,'b2',2,(2)),(4.49,'b2',2,(2)),(4.642,'b2',2,(2)),(4.794,'b2',2,(2)))

print self.Case_Name
if self.Case_Name == 'sMiRT_SC3 south' or self.beamCaseNo==198:
    L = 4572.00/1000.0
    Lbeam = 406.00/1000.0
    Lbeam = 304.00/1000.0
    bottomt = 4.76/1000.0
    topt = 4.76/1000.0
    plateFyBottom = 379.00
    plateFyTop = 379.00
    plateFuBottom = 429.00
    plateFuTop = 429.00
    concretefc = 40.10

self.studbh = 0.00/1000.0
self.studbDia = 0.00/1000.0
self.studbFullHeight = False
self.studbResistance = 0.00+1000.0
self.studbfy = 1.00
self.studbfu = 1.50
self.studb2h = 203.00/1000.0
self.studb2Dia = 6.12/1000.0
```

509
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2FullHeight = True
self.studb2Resistance = 11.05*1000.0
self.studb2fy = 419.00
self.studb2fu = 469.00
self.studb2th = 0.08/1000.0
self.studb2Dia = 0.09/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00+1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50
self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00+1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50
self.studb2th = 0.08/1000.0
self.studb2Dia = 0.09/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00+1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 203.20/1000.0
self.support1a = 4368.80/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 1218.20/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints . extend(((1218.2/1000.0, 0.0),))
self.studPositionDescriptions . extend(((0.82, b2, 2, (2)), (0.234, b2, 2, (2)), (0.386, b2, 2, (2)), (0.538, b2, 2, (2)), (0.69, b2, 2, (2)), (0.842, b2, 2, (2)), (0.994, b2, 2, (2)), (1.146, b2, 2, (2)), (1.298, b2, 2, (2)), (1.45, b2, 2, (2)), (1.602, b2, 2, (2)), (1.754, b2, 2, (2)), (1.906, b2, 2, (2)), (2.058, b2, 2, (2)), (2.21, b2, 2, (2)), (2.362, b2, 2, (2)), (2.514, b2, 2, (2)), (2.666, b2, 2, (2)), (2.818, b2, 2, (2)), (2.97, b2, 2, (2)), (3.122, b2, 2, (2)), (3.274, b2, 2, (2)), (3.426, b2, 2, (2)), (3.578, b2, 2, (2)), (3.73, b2, 2, (2)), (3.882, b2, 2, (2)), (4.034, b2, 2, (2)), (4.186, b2, 2, (2)), (4.338, b2, 2, (2)), (4.49, b2, 2, (2)), (4.642, b2, 2, (2)),))

print self.Case_Name
if self.Case_Name == 'shMRT_SC4 north' or self.beamCaseNo==199:
self.L = 4572.00/1000.0
self.beamb = 406.00/1000.0
self.beamb = 304.00/1000.0
self.beamth = 4.76/1000.0
self.top = 4.76/1000.0
self.plateFyBottom = 379.00
self.plateFyTop = 379.00
self.plateFuBottom = 429.00
self.plateFuTop = 429.00
self.concretefc = 50.78
```

510
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00*1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 203.00/1000.0
self.studb2Dia = 5.59/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 9.21*1000.0
self.studb2fy = 419.00
self.studb2fu = 469.00

self.studt1h = 0.00/1000.0
self.studt1Dia = 0.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 0.00*1000.0
self.studt1fy = 1.00
self.studt1fu = 1.50

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 203.20/1000.0
self.support1a = 4368.80/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1218.20/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1218.2/1000.0,0.0),))

```

print self.Case_Name
if self.Case_Name == 'sMiRT_SC4 south' or self.beamCaseNo==200:

self.L = 4572.00/1000.0
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.beamh = 406.0/1000.0
self.beamb = 304.0/1000.0
self.bottomt = 4.76/1000.0
self.topt = 4.76/1000.0
self.plateFyBottom = 379.0
self.plateFyTop = 379.0
self.plateFuBottom = 379.0
self.plateFuTop = 379.0
self.concretefc = 50.78
self.studb1h = 0.0/1000.0
self.studb1Dia = 0.0/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.0*1000.0
self.studb1fy = 1.0
self.studb1fu = 1.5
self.studb2h = 203.0/1000.0
self.studb2Dia = 5.0/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 7.37*1000.0
self.studb2fy = 419.0
self.studb2fu = 469.0
self.studt1h = 0.0/1000.0
self.studt1Dia = 0.0/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 0.0*1000.0
self.studt1fy = 1.0
self.studt1fu = 1.5
self.studt2h = 0.0/1000.0
self.studt2Dia = 0.0/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.0*1000.0
self.studt2fy = 1.0
self.studt2fu = 1.5
self.CyclicLoad = 50.0*1000.0
self.appliedDeflection = 0.05
self.support1 = 203.2/1000.0
self.support1a = 4368.8/1000.0
self.support1b = -100.0/1000.0
self.support1c = -100.0/1000.0
self.load1 = 1218.2/1000.0
self.load1a = -100.0/1000.0
self.load1b = -100.0/1000.0
self.load1c = -100.0/1000.0
self.load1d = -100.0/1000.0
self.load1e = -100.0/1000.0
self.load1f = -100.0/1000.0
self.load1g = -100.0/1000.0
self.loadFactor1 = 1.0
self.loadFactor1a = -1.0
self.loadFactor1b = -1.0
self.loadFactor1c = -1.0
self.loadFactor1d = -1.0
self.loadFactor1e = -1.0
self.loadFactor1f = -1.0
self.loadFactor1g = -1.0
self.loadDefPoints = []
self.loadDefPoints.extend(((1218.2/1000.0, 0.0,))
self.studPositionDescriptions.extend(((0.127, 'b2', 2, (2)), (0.254, 'b2', 2, (2)), (0.381, 'b2', 2, (2)), (0.508, 'b2', 2, (2)), (0.635, 'b2', 2, (2)), (0.762, 'b2', 2, (2)), (0.889, 'b2', 2, (2)), (1.016, 'b2', 2, (2)), (1.143, 'b2', 2, (2)), (1.27, 'b2', 2, (2)), (1.397, 'b2', 2, (2)), (1.524, 'b2', 2, (2)), (1.651, 'b2', 2, (2)), (1.778, 'b2', 2, (2)), (1.905, 'b2', 2, (2)))
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

# (2.032, 'b2', 2, (2)), (2.159, 'b2', 2, (2)), (2.286, 'b2', 2, (2)), (2.3876, 'b2', 2, (2)), (2.4992, 'b2', 2, (2)), (2.5958, 'b2', 2, (2)), (2.6924, 'b2', 2, (2)), (2.794, 'b2', 2, (2)), (2.8956, 'b2', 2, (2)), (2.9972, 'b2', 2, (2)), (3.0988, 'b2', 2, (2)), (3.2004, 'b2', 2, (2)), (3.302, 'b2', 2, (2)), (3.4036, 'b2', 2, (2)), (3.5052, 'b2', 2, (2)), (3.6068, 'b2', 2, (2)), (3.7084, 'b2', 2, (2)), (3.81, 'b2', 2, (2)), (3.9116, 'b2', 2, (2)), (4.0132, 'b2', 2, (2)), (4.1148, 'b2', 2, (2)), (4.2164, 'b2', 2, (2)), (4.318, 'b2', 2, (2)), (4.4196, 'b2', 2, (2)), (4.5212, 'b2', 2, (2)),)

print self.Case_Name
if self.Case_Name == 'sMRT_SC5 south' or self.beamCaseNo==201:

    self.L = 4572.00/1000.0
    self.beamb = 406.00/1000.0
    self.beamb = 304.00/1000.0
    self.bottomt = 4.76/1000.0
    self.topt = 4.76/1000.0
    self.plateFyBottom = 379.00
    self.plateFuTop = 379.00
    self.concretefc = 55.12
    self.studb1h = 0.00/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00×1000.0
    self.studb1fy = 1.00
    self.studb1fu = 1.50
    self.studb2h = 203.00/1000.0
    self.studb2Dia = 6.12/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 11.05×1000.0
    self.studb2fy = 419.00
    self.studb2fu = 469.00
    self.studt1h = 0.00/1000.0
    self.studt1Dia = 0.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 0.00×1000.0
    self.studt1fy = 1.00
    self.studt1fu = 1.50
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00×1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00×1000.0
    self.appliedDeflection = 0.05
    self.support1 = 203.20/1000.0
    self.support1a = 4368.80/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 812.20/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = -100.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactorId = -100.00
self.loadFactorle = -100.00
self.loadFactorlf = -100.00
self.loadFactorlg = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((812.2/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.152, 'b2', 2, (2)), (0.304, 'b2', 2, (2)), (0.456, 'b2', 2, (2)), (0.608, 'b2', 2, (2)), (0.76, 'b2', 2, (2)), (0.912, 'b2', 2, (2)), (1.064, 'b2', 2, (2)), (1.216, 'b2', 2, (2)), (1.368, 'b2', 2, (2)), (1.52, 'b2', 2, (2)), (1.672, 'b2', 2, (2)), (1.824, 'b2', 2, (2)), (1.976, 'b2', 2, (2)), (2.128, 'b2', 2, (2)), (2.28, 'b2', 2, (2)), (2.415, 'b2', 2, (2)), (2.54, 'b2', 2, (2)), (2.667, 'b2', 2, (2)), (2.794, 'b2', 2, (2)), (2.921, 'b2', 2, (2)), (3.048, 'b2', 2, (2)), (3.175, 'b2', 2, (2)), (3.302, 'b2', 2, (2)), (3.429, 'b2', 2, (2)), (3.556, 'b2', 2, (2)), (3.683, 'b2', 2, (2)), (3.81, 'b2', 2, (2)), (3.937, 'b2', 2, (2)), (4.064, 'b2', 2, (2)), (4.191, 'b2', 2, (2)), (4.318, 'b2', 2, (2)), (4.445, 'b2', 2, (2)),))

print self.Case_Name
if self.Case_Name == 'sMRT_SC5 north' or self.beamCaseNo==202:
    self.L = 4572.00/1000.0
    self.beamb = 406.00/1000.0
    self.beamb = 304.00/1000.0
    self.bottomt = 4.76/1000.0
    self.beamh = 4.76/1000.0
    self.topf = 4.76/1000.0
    self.plateFyBottom = 379.00
    self.plateFyTop = 429.00
    self.plateFyTop = 429.00
    self.concretefc = 55.12

    self.studb1h = 0.00/1000.0
    self.studb1Dia = 0.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 0.00
    self.studb1fy = 1.00
    self.studb1fu = 1.50

    self.studb2h = 203.00/1000.0
    self.studb2Dia = 5.59/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 9.21
    self.studb2fy = 419.00
    self.studb2fu = 469.00

    self.studt1h = 0.00/1000.0
    self.studt1Dia = 0.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 0.00
    self.studt1fy = 1.00
    self.studt1fu = 1.50

    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00
    self.studt2fy = 1.00
    self.studt2fu = 1.50

    self.CyclicLoad = 50.00
    self.appliedDeflection = 0.05

    self.support1 = 203.20/1000.0
    self.support1a = 4366.80/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0

    self.load1 = 812.20/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
```

APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.load1d = -100.00 / 1000.0
self.load1e = -100.00 / 1000.0
self.load1f = -100.00 / 1000.0
self.load1g = -100.00 / 1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((812.2 / 1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.152, 'b2', 2, (2)), (0.304, 'b2', 2, (2)), (0.456, 'b2', 2, (2)), (0.608, 'b2', 2, (2)), (0.76, 'b2', 2, (2)), (0.912, 'b2', 2, (2)), (1.064, 'b2', 2, (2)), (1.216, 'b2', 2, (2)), (1.368, 'b2', 2, (2)), (1.52, 'b2', 2, (2)), (1.672, 'b2', 2, (2)), (1.824, 'b2', 2, (2)), (1.976, 'b2', 2, (2)), (2.128, 'b2', 2, (2)), (2.28, 'b2', 2, (2)), (2.431, 'b2', 2, (2)), (2.54, 'b2', 2, (2)), (2.667, 'b2', 2, (2)), (2.794, 'b2', 2, (2)), (2.921, 'b2', 2, (2)), (3.048, 'b2', 2, (2)), (3.175, 'b2', 2, (2)), (3.302, 'b2', 2, (2)), (3.429, 'b2', 2, (2)), (3.556, 'b2', 2, (2)), (3.683, 'b2', 2, (2)), (3.81, 'b2', 2, (2)), (3.937, 'b2', 2, (2)), (4.064, 'b2', 2, (2)), (4.191, 'b2', 2, (2)), (4.318, 'b2', 2, (2)), (4.445, 'b2', 2, (2)),))

print self.Case_Name
if self.Case_Name == 'smRT_SC6' or self.beamCaseNo==203:

self.L = 4572.00 / 1000.0
self.beamh = 406.00 / 1000.0
self.beamb = 304.00 / 1000.0
self.bottomt = 4.76 / 1000.0
self.top = 4.76 / 1000.0
self.plateFyBottom = 379.00
self.plateFyTop = 379.00
self.plateFuBottom = 429.00
self.plateFuTop = 429.00
self.concretefc = 55.12
self.stud1h = 0.00 / 1000.0
self.stud1Dia = 0.00 / 1000.0
self.stud1FullHeight = False
self.stud1Resistance = 0.00 * 1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50
self.stud2h = 203.00 / 1000.0
self.stud2Dia = 6.12 / 1000.0
self.stud2FullHeight = True
self.stud2Resistance = 11.05 * 1000.0
self.stud2fy = 419.00
self.stud2fu = 469.00
self.stud3h = 0.00 / 1000.0
self.stud3Dia = 0.00 / 1000.0
self.stud3FullHeight = False
self.stud3Resistance = 0.00 * 1000.0
self.stud3fy = 1.00
self.stud3fu = 1.50
self.CyclicLoad = 50.00 * 1000.0
self.appliedDeflection = 0.05
```

515
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.support1 = 203.20/1000.0
self.support1a = 4368.80/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 2314.40/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((2314.4/1000.0, 0.0),))

self.studPositionDescriptions.extend(((0.082, 'b2', 2, (2)), (0.234, 'b2', 2, (2)), (0.386, 'b2', 2, (2)), (0.538, 'b2', 2, (2)), (0.69, 'b2', 2, (2)), (0.842, 'b2', 2, (2)), (1.094, 'b2', 2, (2)), (1.146, 'b2', 2, (2)), (1.298, 'b2', 2, (2)), (1.45, 'b2', 2, (2)), (1.602, 'b2', 2, (2)), (1.754, 'b2', 2, (2)), (1.906, 'b2', 2, (2)), (2.058, 'b2', 2, (2)), (2.21, 'b2', 2, (2)), (2.362, 'b2', 2, (2)), (2.514, 'b2', 2, (2)), (2.666, 'b2', 2, (2)), (2.818, 'b2', 2, (2)), (2.97, 'b2', 2, (2)), (3.122, 'b2', 2, (2)), (3.274, 'b2', 2, (2)), (3.426, 'b2', 2, (2)), (3.578, 'b2', 2, (2)), (3.73, 'b2', 2, (2)), (3.882, 'b2', 2, (2)), (4.034, 'b2', 2, (2)), (4.186, 'b2', 2, (2)), (4.338, 'b2', 2, (2)), (4.49, 'b2', 2, (2)),))

print self.Case_Name

if self.Case_Name == 'Leng_JZ2.5-1' or self.beamCaseNo==206:
    self.L = 2800.00/1000.0
    self.beamh = 300.00/1000.0
    self.beamb = 300.00/1000.0
    self.bottomt = 6.00/1000.0
    self.topt = 6.00/1000.0
    self.plateFyBottom = 350.00
    self.plateFyTop = 350.00
    self.plateFuBottom = 350.00
    self.plateFuTop = 350.00
    self.concretes = 40.80
    self.stud1h = 70.00/1000.0
    self.stud1Dia = 13.00/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 38.76*1000.0
    self.stud1fy = 315.00
    self.stud1fu = 365.00
    self.stud2h = 288.00/1000.0
    self.stud2Dia = 9.50/1000.0
    self.stud2FullHeight = True
    self.stud2Resistance = 25.80*1000.0
    self.stud2fy = 295.00
    self.stud2fu = 455.00
    self.stud1h = 70.00/1000.0
    self.stud1Dia = 13.00/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 38.76*1000.0
    self.stud1fy = 315.00
    self.stud1fu = 365.00
    self.stud2h = 0.00/1000.0

print self.load1
```

516
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 300.00/1000.0
self.support1a = 2500.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1050.00/1000.0
self.load1a = 1750.00/1000.0
self.load1b = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1050/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.1, 'b1', 3,(3)) , (0.2, 'b1', 3,(3)) , (0.3, 'b1', 3,(3)) , (0.4, 'b1', 3,(3)) , (0.5, 'b1', 3,(3)) , (0.6, 'b1', 3,(3)) , (0.7, 'b1', 3,(3)) , (0.8, 'b1', 3,(3)) , (0.9, 'b1', 3,(3)) , (1.0, 'b1', 3,(3)) , (1.1, 'b1', 3,(3)) , (1.2, 'b1', 3,(3)) , (1.3, 'b1', 3,(3)) , (1.4, 'b1', 3,(3)) , (1.5, 'b1', 3,(3)) , (1.6, 'b1', 3,(3)) , (1.7, 'b1', 3,(3)) , (1.8, 'b1', 3,(3)) , (1.9, 'b1', 3,(3)) , (2.0, 'b1', 3,(3)) , (2.1, 'b1', 3,(3)) , (2.2, 'b1', 3,(3)) , (2.3, 'b1', 3,(3)) , (2.4, 'b1', 3,(3)) , (2.5, 'b1', 3,(3)) , (2.6, 'b1', 3,(3)) , (2.7, 'b1', 3,(3)) , (2.8, 'b1', 3,(3))
)

self.studPositionDescriptions.extend(((0.2, 'b2', 2,(2)) , (0.4, 'b2', 2,(2)) , (0.6, 'b2', 2,(2)) , (0.8, 'b2', 2,(2)) , (1.0, 'b2', 2,(2)) , (1.2, 'b2', 2,(2)) , (1.4, 'b2', 2,(2)) , (1.6, 'b2', 2,(2)) , (1.8, 'b2', 2,(2)) , (2.0, 'b2', 2,(2)) , (2.2, 'b2', 2,(2))
)

self.studPositionDescriptions.extend(((0.1, 't1', 3,(3)) , (0.2, 't1', 3,(3)) , (0.3, 't1', 3,(3)) , (0.4, 't1', 3,(3)) , (0.5, 't1', 3,(3)) , (0.6, 't1', 3,(3)) , (0.7, 't1', 3,(3)) , (0.8, 't1', 3,(3)) , (0.9, 't1', 3,(3)) , (1.0, 't1', 3,(3)) , (1.1, 't1', 3,(3)) , (1.2, 't1', 3,(3)) , (1.3, 't1', 3,(3)) , (1.4, 't1', 3,(3)) , (1.5, 't1', 3,(3)) , (1.6, 't1', 3,(3)) , (1.7, 't1', 3,(3)) , (1.8, 't1', 3,(3)) , (1.9, 't1', 3,(3)) , (2.0, 't1', 3,(3)) , (2.1, 't1', 3,(3)) , (2.2, 't1', 3,(3)) , (2.3, 't1', 3,(3)) , (2.4, 't1', 3,(3))
)

print self.Case_Name
if self.Case_Name == 'Long_JZ3.0-1' or self.beamCaseNo==207:

self.L = 3020.00/1000.0
self.beamh = 300.00/1000.0
self.beamb = 300.00/1000.0
self.bottomt = 6.00/1000.0
self.top = 6.00/1000.0
self.plateFyBottom = 350.00
self.plateFyTop = 350.00
self.plateFuBottom = 440.00
self.plateFuTop = 440.00
self.concreteFc = 40.00

self.studb1h = 70.00/1000.0
self.studb1Dia = 13.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 38.76+1000.0
self.studb1fy = 315.00
self.studb1fu = 365.00
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2h = 288.00/1000.0
self.studb2Dia = 9.50/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 25.80+1000.0
self.studb2fy = 295.00
self.studb2fu = 455.00

self.stud1h = 70.00/1000.0
self.stud1Dia = 13.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 38.78+1000.0
self.stud1fy = 315.00
self.stud1fu = 365.00

self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 360.00/1000.0
self.support1a = 2660.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 1260.00/1000.0
self.load1a = 1760.00/1000.0
self.load1b = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((1260/1000.0, 0.0),))

self.setPositionDescriptions.extend(((0.07, 'b1', 3, (3)), (0.19, 'b1', 3, (3)), (0.31, 'b1', 3, (3)), (0.43, 'b1', 3, (3)), (0.55, 'b1', 3, (3)), (0.67, 'b1', 3, (3)), (0.79, 'b1', 3, (3)), (0.91, 'b1', 3, (3)), (1.03, 'b1', 3, (3)), (1.15, 'b1', 3, (3)), (1.27, 'b1', 3, (3)), (1.39, 'b1', 3, (3)), (1.51, 'b1', 3, (3)), (1.63, 'b1', 3, (3)), (1.75, 'b1', 3, (3)), (1.87, 'b1', 3, (3)), (1.99, 'b1', 3, (3)), (2.11, 'b1', 3, (3)), (2.23, 'b1', 3, (3)), (2.35, 'b1', 3, (3)), (2.47, 'b1', 3, (3)), (2.59, 'b1', 3, (3)), (2.71, 'b1', 3, (3)), (2.83, 'b1', 3, (3)), (2.95, 'b1', 3, (3))),)

self.setPositionDescriptions.extend(((0.07, 'b2', 2, (2)), (0.31, 'b2', 2, (2)), (0.55, 'b2', 2, (2)), (0.79, 'b2', 2, (2)), (1.03, 'b2', 2, (2)), (1.27, 'b2', 2, (2)), (1.51, 'b2', 2, (2)), (1.75, 'b2', 2, (2)), (1.99, 'b2', 2, (2)), (2.23, 'b2', 2, (2)), (2.47, 'b2', 2, (2)), (2.71, 'b2', 2, (2)), (2.95, 'b2', 2, (2)))))

self.setPositionDescriptions.extend(((0.07, 't1', 3, (3)), (0.19, 't1', 3, (3)), (0.31, 't1', 3, (3)), (0.43, 't1', 3, (3)), (0.55, 't1', 3, (3)), (0.67, 't1', 3, (3)), (0.79, 't1', 3, (3)), (0.91, 't1', 3, (3)), (1.03, 't1', 3, (3)), (1.15, 't1', 3, (3)), (1.27, 't1', 3, (3)), (1.39, 't1', 3, (3)), (1.51, 't1', 3, (3)), (1.63, 't1', 3, (3)), (1.75, 't1', 3, (3)), (1.87, 't1', 3, (3)), (1.99, 't1', 3, (3)), (2.11, 't1', 3, (3)), (2.23, 't1', 3, (3)), (2.35, 't1', 3, (3)), (2.47, 't1', 3, (3)), (2.59, 't1', 3, (3)), (2.71, 't1', 3, (3)), (2.83, 't1', 3, (3)), (2.95, 't1', 3, (3))),)

print self.Case_Name
if self.Case_Name == 'Long_JZ3.0-N' or self.beamCaseNo==209:

self.L = 2720.00/1000.0
self.beamh = 300.00/1000.0
```

518
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.bamb = 300.00/1000.0
self.bont = 6.00/1000.0
self.top = 6.00/1000.0

self.plateFyBottom = 350.00
self.plateFyTop = 350.00
self.plateFuBottom = 440.00
self.plateFuTop = 440.00
self.concretefc = 25.60

self.stud1h = 80.00/1000.0
self.stud1Dia = 8.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 16.04*1000.0
self.stud1fy = 550.00
self.stud1fu = 600.00

self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00*1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.stud1h = 80.00/1000.0
self.stud1Dia = 8.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 16.04*1000.0
self.stud1fy = 550.00
self.stud1fu = 600.00

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 210.00/1000.0
self.support1a = 2510.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1110.00/1000.0
self.load1a = 1610.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1110/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.03,'b1',5,(5)),(0.1,'b1',5,(5)),(0.17,'b1',5,(5)),(0.24,'b1',5,(5)),(0.31,'b1',5,(5)),(0.38,'b1',5,(5)),(0.45,'b1',5,(5)),(0.52,'b1',5,(5)),(0.59,'b1',5,(5)),(0.66,'b1',5,(5)),(0.73,'b1',5,(5)),(0.8,'b1',5,(5)),(0.87,'b1',5,(5)),(0.94,'b1',5,(5)),(1.01,'b1',5,(5)),(1.08,'b1',5,(5)),(1.15,'b1',5,(5)),(1.22,'b1',5,(5)),(1.29,'b1',5,(5)),(1.36,'b1',5,(5)),(1.43,'b1',5,(5)),(1.5,'b1',5,(5)),(1.57,'b1',5,(5)),(1.64,'b1',5,(5)),(1.71,'b1',5,(5)),(1.78,'b1',5,(5)),(1.85,'b1',5,(5)),(1.92,'b1',5,(5)),(1.99,'b1',5,(5)),(2.06,'b1',5,(5)),(2.13,'b1',5,(5)),(2.2,'b1',5,(5)),(2.27,'b1',5,(5)),(2.34,'b1',5,(5)),(2.41,'b1',5,(5)),(2.48,'b1',5,(5)),(2.55,'b1',5,(5)),(2.62,'b1',5,(5)),(2.69,'b1',5,(5)),(2.76,'b1',5,(5)),(2.83,'b1',5,(5)),(2.9,'b1',5,(5))
```

519
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```
(.1.57, 'b1', 5, 5), (1.64, 'b1', 5, 5), (1.71, 'b1', 5, 5), (1.78, 'b1', 5, 5), (1.85, 'b1', 5, 5), (1.92, 'b1', 5, 5),
(1.99, 'b1', 5, 5), (2.06, 'b1', 5, 5), (2.13, 'b1', 5, 5), (2.2, 'b1', 5, 5), (2.27, 'b1', 5, 5), (2.34, 'b1', 5, 5),
(2.41, 'b1', 5, 5), (2.48, 'b1', 5, 5), (2.55, 'b1', 5, 5), (2.62, 'b1', 5, 5), (2.69, 'b1', 5, 5),
self.studPositionDescriptions.extend(((0.03, 't1', 5, 5), (0.1, 't1', 5, 5), (0.17, 't1', 5, 5), (0.24, 't1', 5, 5),
(0.31, 't1', 5, 5), (0.38, 't1', 5, 5), (0.45, 't1', 5, 5), (0.52, 't1', 5, 5), (0.59, 't1', 5, 5), (0.66, 't1', 5, 5),
(0.73, 't1', 5, 5), (0.8, 't1', 5, 5), (0.87, 't1', 5, 5), (0.94, 't1', 5, 5), (1.01, 't1', 5, 5), (1.08, 't1', 5, 5),
(1.15, 't1', 5, 5), (1.22, 't1', 5, 5), (1.29, 't1', 5, 5), (1.36, 't1', 5, 5), (1.43, 't1', 5, 5), (1.5, 't1', 5, 5),
(1.57, 't1', 5, 5), (1.64, 't1', 5, 5), (1.71, 't1', 5, 5), (1.78, 't1', 5, 5), (1.85, 't1', 5, 5), (1.92, 't1', 5, 5),
(1.99, 't1', 5, 5), (2.06, 't1', 5, 5), (2.13, 't1', 5, 5), (2.2, 't1', 5, 5), (2.27, 't1', 5, 5), (2.34, 't1', 5, 5),
(2.41, 't1', 5, 5), (2.48, 't1', 5, 5), (2.55, 't1', 5, 5), (2.62, 't1', 5, 5), (2.69, 't1', 5, 5)),

print self.Case_Name
if self.Case_Name == 'Leng_JZ2.5-2' or self.beamCaseNo==210:

self.L = 2800.00/1000.0
self.beamh = 3800.00/1000.0
self.beamb = 3000.00/1000.0
self.bottomt = 8.60/1000.0
self.top t = 8.60/1000.0
self.plateFyBottom = 400.00
self.plateFyTop = 400.00
self.plateFuBottom = 500.00
self.plateFuTop = 500.00
self.concretefc = 30.60

self.studb1h = 80.00/1000.0
self.studb1Dia = 8.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 18.01*1000.0
self.studb1fy = 550.00
self.studb1fu = 600.00

self.studb2h = 362.80/1000.0
self.studb2Dia = 6.80/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 13.01*1000.0
self.studb2fy = 405.00
self.studb2fu = 600.00

self.studt1h = 80.00/1000.0
self.studt1Dia = 8.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 18.01*1000.0
self.studt1fy = 550.00
self.studt1fu = 600.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 210.00/1000.0
self.support1a = 2590.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1160.00/1000.0
self.load1a = 1440.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
```
if Case_Name == 'Leng_JZ3.0':
    print(self.

    self.loadDefPoints = []
    self.loadDefPoints.extend(((1160/1000.0,0.0),))

    self.studyPositionDescriptions.extend(((0.04, 'b1', 5,(5)), (0.11, 'b1', 5,(5)), (0.18, 'b1', 5,(5)), (0.25, 'b1', 5,(5)),
                                            (0.32, 'b1', 5,(5)), (0.39, 'b1', 5,(5)), (0.46, 'b1', 5,(5)), (0.53, 'b1', 5,(5)), (0.6, 'b1', 5,(5)), (0.67, 'b1', 5,(5)),
                                            (0.74, 'b1', 5,(5)), (0.81, 'b1', 5,(5)), (0.88, 'b1', 5,(5)), (0.95, 'b1', 5,(5)), (1.02, 'b1', 5,(5)), (1.09, 'b1', 5,(5)),
                                            (1.16, 'b1', 5,(5)), (1.23, 'b1', 5,(5)), (1.3, 'b1', 5,(5)), (1.37, 'b1', 5,(5)), (1.44, 'b1', 5,(5)), (1.51, 'b1', 5,(5)),
                                            (1.58, 'b1', 5,(5)), (1.65, 'b1', 5,(5)), (1.72, 'b1', 5,(5)), (1.79, 'b1', 5,(5)), (1.86, 'b1', 5,(5)), (1.93, 'b1', 5,(5)),
                                            (2.0, 'b1', 5,(5)), (2.07, 'b1', 5,(5)), (2.14, 'b1', 5,(5)), (2.21, 'b1', 5,(5)), (2.28, 'b1', 5,(5)), (2.35, 'b1', 5,(5)),
                                            (2.42, 'b1', 5,(5)), (2.49, 'b1', 5,(5)), (2.56, 'b1', 5,(5)), (2.63, 'b1', 5,(5)), (2.7, 'b1', 5,(5)), (2.77, 'b1', 5,(5)),))

    self.studyPositionDescriptions.extend(((0.09, 'b2', 2,(2)), (0.16, 'b2', 2,(2)), (0.23, 'b2', 2,(2)), (0.3, 'b2', 2,(2)), (0.37, 'b2', 2,(2)), (0.44, 'b2', 2,(2)),
                                            (0.51, 'b2', 2,(2)), (0.58, 'b2', 2,(2)), (0.65, 'b2', 2,(2)), (0.72, 'b2', 2,(2)), (0.79, 'b2', 2,(2)), (0.86, 'b2', 2,(2)),
                                            (0.93, 'b2', 2,(2)), (1, 'b2', 2,(2)), (1.07, 'b2', 2,(2)), (1.14, 'b2', 2,(2)), (1.21, 'b2', 2,(2)), (1.28, 'b2', 2,(2)),
                                            (1.35, 'b2', 2,(2)), (1.42, 'b2', 2,(2)), (1.49, 'b2', 2,(2)), (1.56, 'b2', 2,(2)), (1.63, 'b2', 2,(2)), (1.7, 'b2', 2,(2)),
                                            (1.77, 'b2', 2,(2)), (1.84, 'b2', 2,(2)), (1.91, 'b2', 2,(2)), (1.98, 'b2', 2,(2)),))

    self.studyPositionDescriptions.extend(((0.04, 't1', 5,(5)), (0.11, 't1', 5,(5)), (0.18, 't1', 5,(5)), (0.25, 't1', 5,(5)),
                                            (0.32, 't1', 5,(5)), (0.39, 't1', 5,(5)), (0.46, 't1', 5,(5)), (0.53, 't1', 5,(5)), (0.6, 't1', 5,(5)), (0.67, 't1', 5,(5)),
                                            (0.74, 't1', 5,(5)), (0.81, 't1', 5,(5)), (0.88, 't1', 5,(5)), (0.95, 't1', 5,(5)), (1.02, 't1', 5,(5)), (1.09, 't1', 5,(5)),
                                            (1.16, 't1', 5,(5)), (1.23, 't1', 5,(5)), (1.3, 't1', 5,(5)), (1.37, 't1', 5,(5)), (1.44, 't1', 5,(5)), (1.51, 't1', 5,(5)),
                                            (1.58, 't1', 5,(5)), (1.65, 't1', 5,(5)), (1.72, 't1', 5,(5)), (1.79, 't1', 5,(5)), (1.86, 't1', 5,(5)), (1.93, 't1', 5,(5)),
                                            (2, 't1', 5,(5)), (2.07, 't1', 5,(5)), (2.14, 't1', 5,(5)), (2.21, 't1', 5,(5)), (2.28, 't1', 5,(5)), (2.35, 't1', 5,(5)),
                                            (2.42, 't1', 5,(5)), (2.49, 't1', 5,(5)), (2.56, 't1', 5,(5)), (2.63, 't1', 5,(5)), (2.7, 't1', 5,(5)), (2.77, 't1', 5,(5)),))
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2Resistance = 0.00*1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 210.00/1000.0
self.supportla = 2890.00/1000.0
self.supportrb = -100.00/1000.0
self.supportrc = -100.00/1000.0

self.load1 = 1350.00/1000.0
self.load1a = 1750.00/1000.0
self.loaddb = -100.00/1000.0
self.loaddc = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.load1h = -100.00/1000.0
self.load1i = -100.00/1000.0
self.load1j = -100.00/1000.0
self.loadFactor1a = 1.00
self.loadFactor1ba = 1.00
self.loadFactor1bb = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints.extend(((1350/1000.0, 0.0),))
self.loadFactor1 = 1.00

self.studPositionDescriptions.extend(((0.08, 'b1', 5,(5)),(0.15, 'b1', 5,(5)),(0.22, 'b1', 5,(5)),(0.29, 'b1', 5,(5))
(0.36, 'b1', 5,(5)),(0.43, 'b1', 5,(5)),(0.50, 'b1', 5,(5)),(0.57, 'b1', 5,(5)),(0.64, 'b1', 5,(5)),(0.71, 'b1', 5,(5))
(0.78, 'b1', 5,(5)),(0.85, 'b1', 5,(5)),(0.92, 'b1', 5,(5)),(0.99, 'b1', 5,(5)),(1.06, 'b1', 5,(5)),(1.13, 'b1', 5,(5))
(1.2, 'b1', 5,(5)),(1.27, 'b1', 5,(5)),(1.34, 'b1', 5,(5)),(1.41, 'b1', 5,(5)),(1.48, 'b1', 5,(5)),(1.55, 'b1', 5,(5))
(1.62, 'b1', 5,(5)),(1.69, 'b1', 5,(5)),(1.76, 'b1', 5,(5)),(1.83, 'b1', 5,(5)),(1.9, 'b1', 5,(5)),(1.97, 'b1', 5,(5))
(2.04, 'b1', 5,(5)),(2.11, 'b1', 5,(5)),(2.18, 'b1', 5,(5)),(2.25, 'b1', 5,(5)),(2.32, 'b1', 5,(5)),(2.39, 'b1', 5,(5))
(2.46, 'b1', 5,(5)),(2.53, 'b1', 5,(5)),(2.6, 'b1', 5,(5)),(2.67, 'b1', 5,(5)),(2.74, 'b1', 5,(5)),(2.81, 'b1', 5,(5))
(2.88, 'b1', 5,(5)),(2.95, 'b1', 5,(5)),(3.02, 'b1', 5,(5)).))

self.studPositionDescriptions.extend(((0.185, 'b2', 2,(2)),(0.395, 'b2', 2,(2)),(0.605, 'b2', 2,(2)),(0.815, 'b2'

print self.Case_Name

if self.Case_Name == 'Long_J23.0-3' or self.beamCaseNo==212:

self.L = 3100.00/1000.0
self.beamb = 380.00/1000.0
self.beamb = 300.00/1000.0
self.bottom = 8.66/1000.0
self.top = 8.60/1000.0
self.beamb = 400.00
self.beamb = 400.00
self.beamb = 500.00
self.beamb = 500.00
self.concretetc = 30.60
self.studbh = 80.00/1000.0
self.studDi = 8.00/1000.0

522
```
```python
self superintendentPosDescs.extend([(0.08, 't1', 5, (5)), (0.15, 't1', 5, (5)), (0.22, 't1', 5, (5)), (0.29, 't1', 5, (5))])
self superintendentPosDescs.extend([(0.08, 'b1', 5, (5))])
self loadDefPoints.extend([(1350/1000, 0.0)])
self load1g = self.load1f = self.load1e = self.load1d = self.load1c = self.load1b = self.load1a = 1350/1000
self support1c = self support1b = self support1a = 2100/1000
self appliedDeflection = 0.05
self CyclicLoad = 50.00
self studt2fue = 1.50
self studt2fy = 1.00
self studt2Resistance = 0.00
self studt2FullHeight = False
self studt2Dia = 0.00/1000
self studt2h = 0.00/1000
self studt1fue = 600.00
self studt1Resistance = 18.01
self studt1FullHeight = False
self studt1Dia = 8.00/1000
self studt1h = 80.00/1000
self stud2h = 362.80/1000
self stud2Dia = 9.50/1000
self stud2FullHeight = True
self stud2Resistance = 25.40/1000
self stud2fy = 295.00
self stud2fu = 455.00
self stud2D = 8.00/1000
self stud1Dia = 8.00/1000
self stud1FullHeight = False
self stud1Resistance = 18.01/1000
self stud1fy = 550.00
self stud1fu = 600.00
self self支撑点描述s.extend([(1.2, 'b1', 5, (5)), (1.27, 'b1', 5, (5)), (1.34, 'b1', 5, (5)), (1.41, 'b1', 5, (5)), (1.48, 'b1', 5, (5)), (1.55, 'b1', 5, (5))])
self self支撑点描述s.extend([(1.2, 'b1', 5, (5)), (1.27, 'b1', 5, (5)), (1.34, 'b1', 5, (5)), (1.41, 'b1', 5, (5)), (1.48, 'b1', 5, (5)), (1.55, 'b1', 5, (5))])
self self支撑点描述s.extend([(1.2, 'b1', 5, (5)), (1.27, 'b1', 5, (5)), (1.34, 'b1', 5, (5)), (1.41, 'b1', 5, (5)), (1.48, 'b1', 5, (5)), (1.55, 'b1', 5, (5))])
self self支撑点描述s.extend([(1.2, 'b1', 5, (5)), (1.27, 'b1', 5, (5)), (1.34, 'b1', 5, (5)), (1.41, 'b1', 5, (5)), (1.48, 'b1', 5, (5)), (1.55, 'b1', 5, (5))])
self self支撑点描述s.extend([(1.2, 'b1', 5, (5)), (1.27, 'b1', 5, (5)), (1.34, 'b1', 5, (5)), (1.41, 'b1', 5, (5)), (1.48, 'b1', 5, (5)), (1.55, 'b1', 5, (5))])
self self支撑点描述s.extend([(1.2, 'b1', 5, (5)), (1.27, 'b1', 5, (5)), (1.34, 'b1', 5, (5)), (1.41, 'b1', 5, (5)), (1.48, 'b1', 5, (5)), (1.55, 'b1', 5, (5))])
self self支撑点描述s.extend([(1.2, 'b1', 5, (5)), (1.27, 'b1', 5, (5)), (1.34, 'b1', 5, (5)), (1.41, 'b1', 5, (5)), (1.48, 'b1', 5, (5)), (1.55, 'b1', 5, (5))])
self self支撑点描述s.extend([(1.2, 'b1', 5, (5)), (1.27, 'b1', 5, (5)), (1.34, 'b1', 5, (5)), (1.41, 'b1', 5, (5)), (1.48, 'b1', 5, (5)), (1.55, 'b1', 5, (5))])
```

APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```
print self.Case_Name
if self.Case_Name == 'Leng_JZ3.0-4' or self.beamCaseNo==213:

    self.L = 3100.00/1000.0
    self.beamb = 380.00/1000.0
    self.beambm = 300.00/1000.0
    self.bottomt = 8.60/1000.0
    self.top = 8.60/1000.0

    self.plateFyBottom = 400.00
    self.plateFyTop = 400.00
    self.plateFyBottom = 500.00
    self.concretefc = 30.60

    self.studb1h = 80.00/1000.0
    self.studb1Dia = 8.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 18.01+1000.0
    self.studb1fy = 550.00
    self.studb1fu = 600.00

    self.studb2h = 362.80/1000.0
    self.studb2Dia = 6.80/1000.0
    self.studb2FullHeight = True
    self.studb2Resistance = 13.01+1000.0
    self.studb2fy = 405.00
    self.studb2fu = 600.00

    self.studt1h = 80.00/1000.0
    self.studt1Dia = 8.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 18.01+1000.0
    self.studt1fy = 550.00
    self.studt1fu = 600.00

    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00+1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50

    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05

    self.support1 = 210.00/1000.0
    self.support1a = 2890.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0

    self.load1 = 1350.00/1000.0
    self.load1a = 1750.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0

    self.loadFactor1 = 1.00
    self.loadFactor1a = 1.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
    self.loadFactor1d = -100.00
    self.loadFactor1e = -100.00
```
self.loadFactor1 = -100.0
self.loadFactor2 = -100.0
self.loadDefPoints = []
self.loadDefPoints.extend(((1350/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.08,'b1',5,(5))),(0.36,'b1',5,(5))
,(0.08,'b1',5,(5))),(0.15,'b1',5,(5))),(0.22,'b1',5,(5))
,(0.29,'b1',5,(5))),(0.36,'b1',5,(5))),(0.43,'b1',5,(5))
,(0.5,'b1',5,(5))),(0.57,'b1',5,(5))),(0.64,'b1',5,(5))
,(0.71,'b1',5,(5))),(0.78,'b1',5,(5))),(0.85,'b1',5,(5))
,(0.92,'b1',5,(5))),(0.99,'b1',5,(5))),(1.06,'b1',5,(5))
,(1.13,'b1',5,(5))),(1.2,'b1',5,(5))),(1.27,'b1',5,(5))
,(1.34,'b1',5,(5))),(1.41,'b1',5,(5))),(1.48,'b1',5,(5))
,(1.55,'b1',5,(5))),(1.62,'b1',5,(5))),(1.69,'b1',5,(5))
,(1.76,'b1',5,(5))),(1.83,'b1',5,(5))),(1.9,'b1',5,(5))
,(1.97,'b1',5,(5))),(2.04,'b1',5,(5))),(2.11,'b1',5,(5))
,(2.18,'b1',5,(5))),(2.25,'b1',5,(5))),(2.32,'b1',5,(5))
,(2.39,'b1',5,(5))),(2.46,'b1',5,(5))),(2.53,'b1',5,(5))
,(2.6,'b1',5,(5))),(2.67,'b1',5,(5))),(2.74,'b1',5,(5))
,(2.81,'b1',5,(5))),(2.88,'b1',5,(5))),(3.02,'b1',5,(5))
)
self.studPositionDescriptions.extend(((0.08,'b2',2,(2))),(0.22,'b2',2,(2))
,(0.36,'b2',2,(2))),(0.64,'b2',2,(2))
,(0.85,'b2',2,(2))),(1.06,'b2',2,(2))
,(1.27,'b2',2,(2))),(1.48,'b2',2,(2))
,(1.69,'b2',2,(2))),(1.9,'b2',2,(2))
,(2.11,'b2',2,(2))),(2.32,'b2',2,(2))
,(2.53,'b2',2,(2))),(2.74,'b2',2,(2))
,(2.95,'b2',2,(2))
)

self.loadDefPoints.extend(((1350/1000.0,0.0),))

if self.loadFactor1 == 0.0 or self.loadFactor2 == 0.0:
    print 'Error in loadFactor.'
if self.loadDefPoints == []: print 'Error in loadDefPoints.'
if not self.studPositionDescriptions: print 'Error in studPositionDescriptions.'
if not self.loadDefPoints: print 'Error in loadDefPoints.'
if self.beamCaseNo==214:
    self.beamL = 3400.00/1000.0
    self.beamb = 380.00/1000.0
    self.beambottom = 8.60/1000.0
    self.beamtop = 8.60/1000.0
    self.beamh = 380.00/1000.0
    self.beamf = 3400.00/1000.0
    self.beamCaseNo = 214

if self.Case_Name == 'Leng_1JZ3.5-2' or self.beamCaseNo==214:
    self.beamL = 3400.00/1000.0
    self.beamb = 380.00/1000.0
    self.beambottom = 8.60/1000.0
    self.beamtop = 8.60/1000.0
    self.beamh = 380.00/1000.0
    self.beamf = 3400.00/1000.0
    self.beamCaseNo = 214

if self.loadFactor1 == 0.0 or self.loadFactor2 == 0.0:
    print 'Error in loadFactor.'
if self.loadDefPoints == []: print 'Error in loadDefPoints.'
if not self.studPositionDescriptions: print 'Error in studPositionDescriptions.'
if not self.loadDefPoints: print 'Error in loadDefPoints.'
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1 = 1.0
self.loadFactor2 = 0.0
self.loadFactor3 = 0.0
self.loadFactor4 = 0.0

self.loadDefPoints = []
self.loadDefPoints.extend(((1540/1000.0, 0.0),))

self.loadFactor1a = 1.00
self.loadFactor1b = 2.00
self.loadFactor1c = 3.00
self.loadFactor1d = 4.00
self.loadFactor1e = 5.00
self.loadFactor1f = 6.00

self.loadFactor2a = 7.00
self.loadFactor2b = 8.00
self.loadFactor2c = 9.00
self.loadFactor2d = 10.00

self.loadFactor3a = 11.00
self.loadFactor3b = 12.00
self.loadFactor3c = 13.00
self.loadFactor3d = 14.00

self.loadFactor4a = 15.00
self.loadFactor4b = 16.00
self.loadFactor4c = 17.00
self.loadFactor4d = 18.00

self.loadFactor1 = 1.00

# End of code
```

print self.Case_Name
if self.Case_Name == 'McKinley thesis City1' or self.beamCaseNo==243:

```python
self.L = 4000.0/1000.0
self.beam = 3200.0/1000.0
self.bottom = 10.00/1000.0
self.top = 10.00/1000.0

self.plateFyBottom = 452.20
self.plateFyTop = 452.20
self.plateFullBottom = 540.00
self.plateFullTop = 540.00
self.concrete = 55.00

self.studb1 = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
```

526
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb1Resistance = 0.00 * 1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50
self.studb2h = 200.00 / 1000.0
self.studb2Dia = 25.00 / 1000.0
self.studb2FullHeight = True
self.studb2Resistance = 196.35 * 1000.0
self.studb2fy = 450.00
self.studb2fu = 500.00

self.stud1h = 0.00 / 1000.0
self.stud1Dia = 0.00 / 1000.0
self.stud1FullHeight = False
self.stud1Resistance = 0.00 * 1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50

self.stud2h = 0.00 / 1000.0
self.stud2Dia = 0.00 / 1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00 * 1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00 * 1000.0
self.appliedDeflection = 0.05

self.support1 = 300.00 / 1000.0
self.support1a = 3700.00 / 1000.0
self.support1b = -100.00 / 1000.0
self.support1c = -100.00 / 1000.0

self.load1 = 2000.00 / 1000.0
self.load1a = -100.00 / 1000.0
self.load1b = -100.00 / 1000.0
self.load1c = -100.00 / 1000.0
self.load1d = -100.00 / 1000.0
self.load1e = -100.00 / 1000.0
self.load1f = -100.00 / 1000.0
self.load1g = -100.00 / 1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = None
self.loadDefPoints.extend(((2000 / 1000.0, 0.0),))

self.studyPositionDescriptions.extend(((0.1, 'b2', 5, (5)), (0.3, 'b2', 5, (5)), (0.5, 'b2', 5, (5)), (0.7, 'b2', 5, (5)), (0.9, 'b2', 5, (5)), (1.1, 'b2', 5, (5)), (1.3, 'b2', 5, (5)), (1.5, 'b2', 5, (5)), (1.7, 'b2', 5, (5)), (1.9, 'b2', 5, (5)), (2.1, 'b2', 5, (5)), (2.3, 'b2', 5, (5)), (2.5, 'b2', 5, (5)), (2.7, 'b2', 5, (5)), (2.9, 'b2', 5, (5)), (3.1, 'b2', 5, (5)), (3.3, 'b2', 5, (5)), (3.5, 'b2', 5, (5)), (3.7, 'b2', 5, (5)), (3.9, 'b2', 5, (5)),))

print self.Case_Name

if self.Case_Name == 'McKinley thesis City2' or self.beamCaseNo==244:
    self.L = 4000.00 / 1000.0
self.beamb = 220.00 / 1000.0
self.beamb = 1000.00 / 1000.0
self.bottomt = 10.00 / 1000.0
self.toprt = 10.00 / 1000.0

self.plateFyBottom = 452.20
self.plateFyTop = 452.20
```

527
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.plateFuBottom = 540.00
self.plateFuTop = 540.00
self.concretefc = 56.00
self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50
self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 196.35+1000.0
self.studb2fy = 450.00
self.studb2fu = 500.00
self.studt1h = 0.00/1000.0
self.studt1Dia = 0.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 0.00+1000.0
self.studt1fy = 1.00
self.studt1fu = 1.50
self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00+1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50
self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05
self.support1 = 300.00/1000.0
self.support1a = 3700.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 2000.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((200/1000.0, 0.0), ))
self.studPositionDescriptions.extend(((0.1, 'b2', 5,(5)), (0.3, 'b2', 5,(5)), (0.5, 'b2', 5,(5)), (0.7, 'b2', 5,(5)), (0.9, 'b2', 5,(5)), (1.1, 'b2', 5,(5)), (1.3, 'b2', 5,(5)), (1.5, 'b2', 5,(5)), (1.7, 'b2', 5,(5)), (1.9, 'b2', 5,(5)), (2.1, 'b2', 5,(5)), (2.3, 'b2', 5,(5)), (2.5, 'b2', 5,(5)), (2.7, 'b2', 5,(5)), (2.9, 'b2', 5,(5)), (3.1, 'b2', 5,(5)), (3.3, 'b2', 5,(5)), (3.5, 'b2', 5,(5)), (3.7, 'b2', 5,(5)), (3.9, 'b2', 5,(5)), )

print self.Case_Name
if self.Case_Name == 'McKinley thesis Stud2' or self.beamCaseNo==245:
    self.L = 4000.00/1000.0
```

528
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.beamh = 220.00/1000.0
self.beamb = 1000.00/1000.0
self.bottomt = 10.00/1000.0
self.top t = 10.00/1000.0
self.plateFyBottom = 452.20
self.plateFyTop = 452.20
self.plateFuBottom = 452.20
self.plateFuTop = 452.20
self.concretefc = 53.30

self.studb1h = 150.00/1000.0
self.studb1Dia = 19.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 113.41*1000.0
self.studb1fy = 450.00
self.studb1fu = 500.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 150.00/1000.0
self.studt1Dia = 19.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 113.41*1000.0
self.studt1fy = 450.00
self.studt1fu = 500.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 300.00/1000.0
self.support1a = 3700.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 2000.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((2000/1000,0,0),))

self.studPositionDescriptions.extend([(0.1715, 'b1',7,(7) ),(0.3305, 'b1',7,(7) ),(0.4985, 'b1',7,(7) ),(0.6485, 'b1',7,(7) ),(0.8075, 'b1',7,(7) ),(0.9665, 'b1',7,(7) ),(1.1255, 'b1',7,(7) ),(1.2845, 'b1',7,(7) ),(1.4435, 'b1',7,(7) ),(1.6025, 'b1',7,(7) ),(1.7615, 'b1',7,(7) ),(1.9205, 'b1',7,(7) ),(2.0795, 'b1',7,(7) ),(2.2385, 'b1',7,(7) ),(2.3975, 'b1',7,(7) )])
```

529
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
b1', 't1', '7', '7')}, (2.5565, 'b1', '7', '7'), (2.7155, 'b1', '7', '7'), (2.8745, 'b1', '7', '7'), (3.0335, 'b1', '7', '7'), (3.1925, 'b1', '7', '7'), (3.3515, 'b1', '7', '7'), (3.5105, 'b1', '7', '7'), (3.6695, 'b1', '7', '7'), (3.8285, 'b1', '7', '7'), (3.9875, 'b1', '7', '7'),
}
self.studPositionDescriptions.extend(((0.1215, 't1', '7', '7'), (0.2805, 't1', '7', '7'), (0.4395, 't1', '7', '7'), (0.5985, 't1', '7', '7'), (0.7575, 't1', '7', '7'), (0.9165, 't1', '7', '7'), (1.0755, 't1', '7', '7'), (1.2345, 't1', '7', '7'), (1.3935, 't1', '7', '7'), (1.5525, 't1', '7', '7'), (1.7115, 't1', '7', '7'), (1.8705, 't1', '7', '7'), (2.0295, 't1', '7', '7'), (2.1885, 't1', '7', '7'), (2.3475, 't1', '7', '7'), (2.5065, 't1', '7', '7'), (2.6655, 't1', '7', '7'), (2.8245, 't1', '7', '7'), (2.9835, 't1', '7', '7'), (3.1425, 't1', '7', '7'), (3.3015, 't1', '7', '7'), (3.4605, 't1', '7', '7'), (3.6195, 't1', '7', '7'), (3.7785, 't1', '7', '7'), (3.9375, 't1', '7', '7'),)

print self.Case_Name
if self.Case_Name == 'McKinley thesis Stud2b' or self.beamCaseNo==246:
    self.L = 4000.00/1000.0
    self.beamh = 220.00/1000.0
    self.beamb = 100.00/1000.0
    self.bottomt = 10.00/1000.0
    self.top_t = 10.00/1000.0
    self.plateFyBottom = 452.20
    self.plateFyTop = 452.20
    self.plateFuBottom = 540.00
    self.plateFuTop = 540.00
    self.concretefc = 58.00
    self.studb1h = 150.00/1000.0
    self.studb1Dia = 19.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 113.41*1000.0
    self.studb1fy = 450.00
    self.studb1fu = 500.00
    self.studb2h = 0.00/1000.0
    self.studb2Dia = 0.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 0.00*1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50
    self.studt1h = 150.00/1000.0
    self.studt1Dia = 19.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 113.41*1000.0
    self.studt1fy = 450.00
    self.studt1fu = 500.00
    self.studt2h = 0.00/1000.0
    self.studt2Dia = 0.00/1000.0
    self.studt2FullHeight = False
    self.studt2Resistance = 0.00*1000.0
    self.studt2fy = 1.00
    self.studt2fu = 1.50
    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05
    self.support1 = 300.00/1000.0
    self.support1a = 3700.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 2000.00/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = -100.00
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1b = -100.0
self.loadFactor1c = -100.0
self.loadFactor1d = -100.0
self.loadFactor1e = -100.0
self.loadFactor1f = -100.0
self.loadFactor1g = -100.0
self.loadDefPoints = []
self.loadDefPoints.extend(((200/1000.0, 0.0),))

self.studPositionDescriptions.extend(((0.1715, 'b1', 7, (7)), (0.3305, 'b1', 7, (7)), (0.4895, 'b1', 7, (7)), (0.6485, 'b1', 7, (7)), (0.8075, 'b1', 7, (7)), (0.9665, 'b1', 7, (7)), (1.1255, 'b1', 7, (7)), (1.2845, 'b1', 7, (7)), (1.4435, 'b1', 7, (7)), (1.6025, 'b1', 7, (7)), (1.7615, 'b1', 7, (7)), (1.9205, 'b1', 7, (7)), (2.0795, 'b1', 7, (7)), (2.2385, 'b1', 7, (7)), (2.3975, 'b1', 7, (7)), (2.5565, 'b1', 7, (7)), (2.7155, 'b1', 7, (7)), (2.8745, 'b1', 7, (7)), (3.0335, 'b1', 7, (7)), (3.1925, 'b1', 7, (7)), (3.3515, 'b1', 7, (7)), (3.5105, 'b1', 7, (7)), (3.6695, 'b1', 7, (7)), (3.8285, 'b1', 7, (7)), (3.9875, 'b1', 7, (7)), (4.1465, 'b1', 7, (7)), (4.3055, 'b1', 7, (7)), (4.4645, 'b1', 7, (7)), (4.6235, 'b1', 7, (7)), (4.7825, 'b1', 7, (7)), (4.9415, 'b1', 7, (7)), (5.0875, 'b1', 7, (7)), (5.2465, 'b1', 7, (7)), (5.3955, 'b1', 7, (7)), (5.5555, 'b1', 7, (7)));

self.studPositionDescriptions.extend(((0.1215, 't1', 7, (7)), (0.2805, 't1', 7, (7)), (0.4395, 't1', 7, (7)), (0.5985, 't1', 7, (7)), (0.7575, 't1', 7, (7)), (0.9165, 't1', 7, (7)), (1.0755, 't1', 7, (7)), (1.2345, 't1', 7, (7)), (1.3935, 't1', 7, (7)), (1.5525, 't1', 7, (7)), (1.7115, 't1', 7, (7)), (1.8705, 't1', 7, (7)), (2.0295, 't1', 7, (7)), (2.1885, 't1', 7, (7)), (2.3475, 't1', 7, (7)), (2.5065, 't1', 7, (7)), (2.6655, 't1', 7, (7)), (2.8245, 't1', 7, (7)), (2.9835, 't1', 7, (7)), (3.1425, 't1', 7, (7)), (3.3015, 't1', 7, (7)), (3.4605, 't1', 7, (7)), (3.6195, 't1', 7, (7)), (3.7785, 't1', 7, (7)), (3.9375, 't1', 7, (7)));

print self.Case_Name
if self.Case_Name == 'McKinley thesis City3' or self.beamCaseNo==247:

self.L = 3000.00/1000.0
self.beamh = 224.00/1000.0
self.beamb = 600.00/1000.0
self.bottomt = 12.00/1000.0
self.tpt = 12.00/1000.0

self.plateFyBottom = 431.20
self.plateFyTop = 431.20
self.plateFuBottom = 546.00
self.plateFuTop = 546.00
self.concretefc = 65.20

self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 259.18+1000.0
self.studb2fy = 610.00
self.studb2fu = 660.00

self.studt1h = 170.00/1000.0
self.studt1Dia = 19.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 113.41+1000.0
self.studt1fy = 450.00
self.studt1fu = 500.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00+1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2800.00/1000.0
self.support1b = -100.00/1000.0
```

531
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.support1c = -100.00/1000.0
self.load1 = 1500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((1500/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.2, 'b2', 3,(3)),(0.4, 'b2', 3,(3)),(0.6, 'b2', 3,(3)),(0.8, 'b2', 3,(3)),(1.0, 'b2', 3,(3)),(1.2, 'b2', 3,(3)),(1.4, 'b2', 3,(3)),(1.6, 'b2', 3,(3)),(1.8, 'b2', 3,(3)),(2.0, 'b2', 3,(3)),(2.2, 'b2', 3,(3)),(2.4, 'b2', 3,(3)),(2.6, 'b2', 3,(3)),(2.8, 'b2', 3,(3)),))
self.studPositionDescriptions.extend(((0.3, 't1', 3,(3)),(0.5, 't1', 3,(3)),(0.7, 't1', 3,(3)),(1.7, 't1', 3,(3)),(2.5, 't1', 3,(3)),(2.7, 't1', 3,(3)),))

print self.Case_Name
if self.Case_Name == 'McKinley thesis City4a' or self.beamCaseNo==248:

self.L = 3000.00/1000.0
self.beamb = 220.00/1000.0
self.bottomt = 10.00/1000.0
self.top = 10.00/1000.0
self.plateFyBottom = 381.20
self.plateFyTop = 381.20
self.plateFuBottom = 546.00
self.plateFuTop = 546.00
self.concretefc = 50.00
self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50
self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 242.00+1000.0
self.studb2fy = 610.00
self.studb2fu = 660.00
self.studt1h = 170.00/1000.0
self.studt1Dia = 19.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 113.41+1000.0
self.studt1fy = 450.00
self.studt1fu = 500.00
self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00+1000.0
self.studt2fy = 1.00
```

```python
self.stud2fup = 1.50

self.CyclicLoad = 50.00/1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1500/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.2, 'b2', '3', (3)), (0.4, 'b2', '3', (3)), (0.6, 'b2', '3', (3)), (0.8, 'b2', '3', (3)), (1.0, 'b2', '3', (3)), (1.2, 'b2', '3', (3)), (1.4, 'b2', '3', (3)), (1.6, 'b2', '3', (3)), (1.8, 'b2', '3', (3)), (2.0, 'b2', '3', (3)), (2.2, 'b2', '3', (3)), (2.4, 'b2', '3', (3)), (2.6, 'b2', '3', (3)), (2.8, 'b2', '3', (3)), (3.0, 'b2', '3', (3)))
self.studPositionDescriptions.extend(((0.3, 't1', '3', (3)), (0.5, 't1', '3', (3)), (0.7, 't1', '3', (3)), (0.9, 't1', '3', (3)), (1.1, 't1', '3', (3)), (1.3, 't1', '3', (3)), (1.5, 't1', '3', (3)), (1.7, 't1', '3', (3)))

print self.Case_Name
if self.Case_Name == 'McKinley thesis City4b' or self.beamCaseNo==249:

self.L = 3000.00/1000.0
self.beamh = 220.00/1000.0
self.beamb = 600.00/1000.0
self.bottomt = 10.00/1000.0
self.topt = 10.00/1000.0

self.plateFyBottom = 430.60
self.plateFyTop = 430.60
self.plateFuBottom = 546.00
self.plateFuTop = 546.00
self.concretes = 55.60

self.studh1 = 0.00/1000.0
self.studh1a = 0.00/1000.0
self.studh1FullHeight = False
self.studh1Resistance = 0.00+1000.0
self.studh1f = 1.00
self.studh1fu = 1.50

self.studh2 = 200.00/1000.0
self.studh2a = 25.00/1000.0
self.studh2FullHeight = True
self.studh2Resistance = 259.18+1000.0
self.studh2f = 610.00
self.studh2fu = 660.00

self.stud1h = 170.00/1000.0
self.stud1Da = 19.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 113.41+1000.0
```

533
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud1fy = 450.00
self.stud1fu = 500.00

self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1500/1000.0, 0.0),))
self.studPositionDescriptions.extend(((0.2, 'b2', 3, (3)), (0.4, 'b2', 3, (3)), (0.6, 'b2', 3, (3)), (0.8, 'b2', 3, (3)), (1.0, 'b2', 3, (3)), (1.2, 'b2', 3, (3)), (1.4, 'b2', 3, (3)), (1.6, 'b2', 3, (3)), (1.8, 'b2', 3, (3)), (2.0, 'b2', 3, (3)), (2.2, 'b2', 3, (3)), (2.4, 'b2', 3, (3)), (2.6, 'b2', 3, (3)), (2.8, 'b2', 3, (3)),))
self.studPositionDescriptions.extend(((0.3, 't1', 3, (3)), (0.5, 't1', 3, (3)), (0.7, 't1', 3, (3)), (0.9, 't1', 3, (3)), (1.1, 't1', 3, (3)), (1.3, 't1', 3, (3)), (1.5, 't1', 3, (3)), (1.7, 't1', 3, (3)), (1.9, 't1', 3, (3)), (2.1, 't1', 3, (3)), (2.3, 't1', 3, (3)), (2.5, 't1', 3, (3)), (2.7, 't1', 3, (3)),))

print self.Case_Name
if self.Case_Name == 'McKinley thesis City4c' or self.beamCaseNo==250:

self.L = 3000.00/1000.0
self.beamh = 220.00/1000.0
self.beamb = 500.00/1000.0
self.bottomt = 10.00/1000.0
self.topt = 10.00/1000.0

self.plateFyBottom = 430.60
self.plateFyTop = 430.60
self.plateFuBottom = 546.00
self.plateFuTop = 546.00
self.concretefc = 70.40

self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2Resistance = 259.18*1000.0
self.studb2fy = 610.00
self.studb2fu = 660.00

self.studt1h = 170.00/1000.0
self.studt1Dia = 19.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 113.41*1000.0
self.studt1fy = 450.00
self.studt1fu = 500.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1500/1000,0,0),))

self.studPositionDescriptions.extend(((0.2, 'b2', 3,(3)), (0.4, 'b2', 3,(3)), (0.6, 'b2', 3,(3)), (0.8, 'b2', 3,(3)), (1.0, 'b2', 3,(3)), (1.2, 'b2', 3,(3)), (1.4, 'b2', 3,(3)), (1.6, 'b2', 3,(3)), (1.8, 'b2', 3,(3)), (2.0, 'b2', 3,(3)), (2.2, 'b2', 3,(3)), (2.4, 'b2', 3,(3)), (2.6, 'b2', 3,(3)), (2.8, 'b2', 3,(3)), (3.0, 'b2', 3,(3)), (3.2, 'b2', 3,(3)), (3.4, 'b2', 3,(3)), (3.6, 'b2', 3,(3)), (3.8, 'b2', 3,(3)), (4.0, 'b2', 3,(3)), (4.2, 'b2', 3,(3)), (4.4, 'b2', 3,(3)), (4.6, 'b2', 3,(3)), (4.8, 'b2', 3,(3)), (5.0, 'b2', 3,(3)), (5.2, 'b2', 3,(3)), (5.4, 'b2', 3,(3)), (5.6, 'b2', 3,(3)), (5.8, 'b2', 3,(3)), (6.0, 'b2', 3,(3)), (6.2, 'b2', 3,(3)), (6.4, 'b2', 3,(3)), (6.6, 'b2', 3,(3)), (6.8, 'b2', 3,(3)), (7.0, 'b2', 3,(3)), (7.2, 'b2', 3,(3)), (7.4, 'b2', 3,(3)), (7.6, 'b2', 3,(3)), (7.8, 'b2', 3,(3)), (8.0, 'b2', 3,(3)), (8.2, 'b2', 3,(3)), (8.4, 'b2', 3,(3)), (8.6, 'b2', 3,(3)), (8.8, 'b2', 3,(3)), (9.0, 'b2', 3,(3)), (9.2, 'b2', 3,(3)), (9.4, 'b2', 3,(3)), (9.6, 'b2', 3,(3)), (9.8, 'b2', 3,(3)), (10.0, 'b2', 3,(3)))))

self.studPositionDescriptions.extend(((0.3, 't1', 3,(3)), (0.5, 't1', 3,(3)), (0.7, 't1', 3,(3)), (0.9, 't1', 3,(3)), (1.1, 't1', 3,(3)), (1.3, 't1', 3,(3)), (1.5, 't1', 3,(3)), (1.7, 't1', 3,(3)), (1.9, 't1', 3,(3)), (2.1, 't1', 3,(3)), (2.3, 't1', 3,(3)), (2.5, 't1', 3,(3)), (2.7, 't1', 3,(3)), (2.9, 't1', 3,(3)),))

self.studPositionDescriptions.extend(((0.3, 't1', 3,(3)), (0.5, 't1', 3,(3)), (0.7, 't1', 3,(3)), (0.9, 't1', 3,(3)), (1.1, 't1', 3,(3)), (1.3, 't1', 3,(3)), (1.5, 't1', 3,(3)), (1.7, 't1', 3,(3)), (1.9, 't1', 3,(3)), (2.1, 't1', 3,(3)), (2.3, 't1', 3,(3)), (2.5, 't1', 3,(3)), (2.7, 't1', 3,(3)), (2.9, 't1', 3,(3)),))

print self.Case_Name
if self.Case_Name == 'McKinley thesis City4d' or self.beamCaseNo==251:

self.L = 3000.00/1000.0
self.beamh = 220.00/1000.0
self.beamb = 600.00/1000.0
self.bottomt = 10.00/1000.0
self.top t = 10.00/1000.0
self.plateFyBottom = 397.30
self.plateFyTop = 397.30
self.plateFuBottom = 547.00
self.plateFuTop = 547.00
self.concretef = 39.00

self.studbh1h = 0.00/1000.0
self.studbh1Dia = 0.00/1000.0
```

self.studb1FullHeight = False
self.studb1Resistance = 0.00 * 1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50
self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 205.91 * 1000.0
self.studb2fy = 610.00
self.studb2fu = 660.00
self.studt1h = 170.00/1000.0
self.studt1Dia = 19.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 113.41*1000.0
self.studt1fy = 450.00
self.studt1fu = 500.00
self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00 * 1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50
self.CyclicLoad = 50.00 * 1000.0
self.appliedDeflection = 0.05
self.support1 = 200.00/1000.0
self.support1a = 2800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 1500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((1500/1000.0,0.0),))
self.studPositionDescriptions.extend(((0.2,'b2',3,(3)),(0.4,'b2',3,(3)),(0.6,'b2',3,(3)),(0.8,'b2',3,(3)),(1.0,'b2',3,(3)),(1.2,'b2',3,(3)),(1.4,'b2',3,(3)),(1.6,'b2',3,(3)),(1.8,'b2',3,(3)),(2.0,'b2',3,(3)),(2.2,'b2',3,(3)),(2.4,'b2',3,(3)),(2.6,'b2',3,(3)),(2.8,'b2',3,(3)),))
self.studPositionDescriptions.extend(((0.3,'t1',3,(3)),(0.5,'t1',3,(3)),(0.7,'t1',3,(3)),(1.7,'t1',3,(3)),(2.5,'t1',3,(3)),(2.7,'t1',3,(3)),))

print self.Case_Name
if self.Case_Name == 'McKinley thesis City5' or self.beamCaseNo==252:

self.L = 3000.00/1000.0
self.beamb = 216.00/1000.0
self.beamb = 600.00/1000.0
self.bottomt = 8.00/1000.0
self.top t = 8.00/1000.0

536
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.plateFyBottom = 432.00
self.plateFyTop = 432.00
self.plateFuBottom = 547.00
self.plateFuTop = 547.00
self.concretefc = 57.80

self.studb1h = 0.00 / 1000.0
self.studb1Dia = 0.00 / 1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00 * 1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 200.00 / 1000.0
self.studb2Dia = 25.00 / 1000.0
self.studb2FullHeight = True
self.studb2Resistance = 259.18 * 1000.0
self.studb2fy = 610.00
self.studb2fu = 660.00

self.stud1h = 170.00 / 1000.0
self.stud1Dia = 19.00 / 1000.0
self.stud1FullHeight = False
self.stud1Resistance = 113.41 * 1000.0
self.stud1fy = 450.00
self.stud1fu = 500.00

self.stud2h = 0.00 / 1000.0
self.stud2Dia = 0.00 / 1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00 * 1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00 * 1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00 / 1000.0
self.supportRa = 2800.00 / 1000.0
self.supportRb = -100.00 / 1000.0
self.supportRc = -100.00 / 1000.0

self.load1 = 1500.00 / 1000.0
self.load1a = -100.00 / 1000.0
self.load1b = -100.00 / 1000.0
self.load1c = -100.00 / 1000.0
self.load1d = -100.00 / 1000.0
self.load1e = -100.00 / 1000.0
self.load1f = -100.00 / 1000.0
self.load1g = -100.00 / 1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1500 / 1000, 0.0),))

self.studPositionDescriptions.extend(((0.2, 'b2', 3, (3)), (0.4, 'b2', 3, (3)), (0.6, 'b2', 3, (3)), (0.8, 'b2', 3, (3)), (1.0, 'b2', 3, (3)), (1.2, 'b2', 3, (3)), (1.4, 'b2', 3, (3)), (1.6, 'b2', 3, (3)), (1.8, 'b2', 3, (3)), (2.0, 'b2', 3, (3)), (2.2, 'b2', 3, (3)), (2.4, 'b2', 3, (3)), (2.6, 'b2', 3, (3)), (2.8, 'b2', 3, (3)), (3.0, 'b2', 3, (3))))
self.studPositionDescriptions.extend(((0.3, 't1', 3, (3)), (1.7, 't1', 3, (3)),))

print self.Case_Name
if self.Case_Name == 'McKinley thesis City6' or self.beamCaseNo==253:
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.L = 3000.00/1000.0
self.beamb = 216.00/1000.0
self.bottomt = 8.00/1000.0
self.topt = 8.00/1000.0

self.plateFyBottom = 339.70
self.plateFyTop = 339.70
self.plateFuBottom = 547.00
self.plateFuTop = 547.00
self.concretefc = 61.20

self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00*1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 259.18*1000.0
self.studb2fy = 610.00
self.studb2fu = 660.00

self.studt1h = 170.00/1000.0
self.studt1Dia = 19.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 113.41*1000.0
self.studt1fy = 450.00
self.studt1fu = 500.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1500/1000.0,0.0),))
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studPositionDescriptions.extend(((0.2, 'b2', 3, (3)), (0.44, 'b2', 3, (3)), (0.68, 'b2', 3, (3)), (0.92, 'b2', 3, (3)),
                                      (1.16, 'b2', 3, (3)), (1.4, 'b2', 3, (3)), (1.6, 'b2', 3, (3)), (1.8, 'b2', 3, (3)), (2.2, 'b2', 3, (3)), (2.4, 'b2', 3, (3))
                                      )
self.studPositionDescriptions.extend(((0.32, 't1', 3, (3)), (1.7, 't1', 3, (3)), ))
print self.Case_Name
if self.Case_Name == 'McKinley thesis City6b' or self.beamCaseNo==254:

    self.L = 3000.00/1000.0
    self.beamH = 216.00/1000.0
    self.beamB = 600.00/1000.0
    self.bottomT = 8.08/1000.0
    self.topT = 8.00/1000.0

    self.plateFyBottom = 404.40
    self.plateFyTop = 404.40
    self.plateFuTop = 555.00
    self.plateFuBottom = 555.00
    self.concreteFc = 39.10

    self.studB1h = 0.08/1000.0
    self.studB1Dia = 0.00/1000.0
    self.studB1FullHeight = False
    self.studB1Resistance = 0.00+1000.0
    self.studB1fY = 1.00
    self.studB1fU = 1.50

    self.studB2h = 200.00/1000.0
    self.studB2Dia = 25.00/1000.0
    self.studB2FullHeight = True
    self.studB2Resistance = 206.25+1000.0
    self.studB2fY = 610.00
    self.studB2fU = 660.00

    self.studT1h = 170.00/1000.0
    self.studT1Dia = 19.00/1000.0
    self.studT1FullHeight = False
    self.studT1Resistance = 113.41+1000.0
    self.studT1fY = 450.00
    self.studT1fU = 500.00

    self.studT2h = 0.08/1000.0
    self.studT2Dia = 0.00/1000.0
    self.studT2FullHeight = False
    self.studT2Resistance = 0.00+1000.0
    self.studT2fY = 1.00
    self.studT2fU = 1.50

    self.CyclicLoad = 50.00+1000.0
    self.appliedDeflection = 0.05

    self.support1 = 200.00/1000.0
    self.support1a = 2800.00/1000.0
    self.support1b = -100.00/1000.0
    self.support1c = -100.00/1000.0

    self.load1 = 1500.00/1000.0
    self.load1a = -100.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0

    self.loadFactor1 = 1.00
    self.loadFactor1a = -100.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
    self.loadFactor1d = -100.00
```

539
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend(((1500/1000.0, 0.0),))

self.studPositionDescriptions.extend(((0.2, 'b2', 3, (3)), (0.44, 'b2', 3, (3)), (0.68, 'b2', 3, (3)), (0.92, 'b2', 3, (3)), (1.16, 'b2', 3, (3)), (1.44, 'b2', 3, (3)), (1.68, 'b2', 3, (3)), (1.92, 'b2', 3, (3)), (2.2, 'b2', 3, (3)), (2.4, 'b2', 3, (3)), (2.6, 'b2', 3, (3)), (2.8, 'b2', 3, (3)),))
self.studPositionDescriptions.extend(((0.32, 't1', 3, (3)), (1.7, 't1', 3, (3)),))

print self.Case_Name
if self.Case_Name == 'McKinley thesis City7' or self.beamCaseNo==255:

self.L = 3000.00/1000.0
self.beamb = 216.00/1000.0
self.bottomt = 8.00/1000.0
self.topt = 8.00/1000.0

self.plateFyBottom = 432.00
self.plateFyTop = 432.00
self.plateFuBottom = 555.00
self.plateFuTop = 555.00
self.concretefc = 63.00

self.studb1h = 0.08/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00*1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 259.18*1000.0
self.studb2fy = 610.00
self.studb2fu = 660.00

self.studt1h = 170.00/1000.0
self.studt1Dia = 19.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 113.41*1000.0
self.studt1fy = 450.00
self.studt1fu = 500.00

self.studt2h = 0.08/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
```

540
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((1500/1000.0,0.0),))

self.studPositionDescriptions.extend(((0.2,'b2',3,(3))),(0.43,'b2',3,(3)),(0.66,'b2',3,(3)),(0.89,'b2',3,(3))
 ,(1.12,'b2',3,(3)),(1.4,'b2',3,(3)),(1.6,'b2',3,(3)),(1.8,'b2',3,(3)),(2,'b2',3,(3)),(2.2,'b2',3,(3)),(2.4,'b2
 ','3,(3)),(2.6,'b2',3,(3)),(2.8,'b2',3,(3)),))
self.studPositionDescriptions.extend(((0.315,'t1',3,(3)),(0.545,'t1',3,(3)),(1.7,'t1',3,(3)),))

print self.Case_Name
if self.Case_Name == 'McKinley thesis City8' or self.beamCaseNo==256:

self.L = 3000.00/1000.0
self.beamh = 216.00/1000.0
self.beamb = 600.00/1000.0
self.bottomt = 8.00/1000.0
self.topt = 8.00/1000.0

self.plateFyBottom = 432.00
self.plateFyTop = 432.00
self.plateFuBottom = 432.00
self.plateFuTop = 432.00
self.concretefc = 64.50

self.studb1h = 0.00/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00*1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 259.18*1000.0
self.studb2fy = 610.00
self.studb2fu = 660.00

self.studt1h = 170.00/1000.0
self.studt1Dia = 19.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 113.41*1000.0
self.studt1fy = 450.00
self.studt1fu = 500.00

self.studt2h = 0.00/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00*1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
```

541
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.load1 = 1500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.loadDefPoints.extend(((1500.00/1000.0, 0.0),))

self.studPositionDescriptions.extend(((0.2, 'b2', 3, 3), (0.4, 'b2', 3, 3), (0.6, 'b2', 3, 3), (0.8, 'b2', 3, 3)), (0.98, 'b2', 3, 3), (1.08, 'b2', 3, 3), (1.4, 'b2', 3, 3), (1.6, 'b2', 3, 3), (1.8, 'b2', 3, 3), (2.0, 'b2', 3, 3), (2.2, 'b2', 3, 3), (2.4, 'b2', 3, 3), (2.6, 'b2', 3, 3), (2.8, 'b2', 3, 3), (3.0, 'b2', 3, 3), (3.2, 'b2', 3, 3)))
self.studPositionDescriptions.extend(((0.31, 't1', 3, 3), (0.53, 't1', 3, 3), (1.7, 't1', 3, 3),))

print self.Case_Name
if self.Case_Name == 'McKinley thesis City9' or self.beamCaseNo==257:

    self.L = 3000.00/1000.0
self.beamb = 216.00/1000.0
self.beamb = 600.00/1000.0
self.bottomt = 8.00/1000.0
self.topt = 8.00/1000.0
self.plateFyBottom = 432.00
self.plateFyTop = 432.00
self.plateFuBottom = 555.00
self.plateFuTop = 555.00
self.concretefc = 4.690
self.studb1h = 0.08/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 100.00*1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50
self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 259.18+1000.0
self.studb2fy = 610.00
self.studb2fu = 660.00
self.studt1h = 170.00/1000.0
self.studt1Dia = 19.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 113.41+1000.0
self.studt1fy = 450.00
self.studt1fu = 500.00
self.studt2h = 0.08/1000.0
self.studt2Dia = 0.00/1000.0
self.studt2FullHeight = False
self.studt2Resistance = 0.00+1000.0
self.studt2fy = 1.00
self.studt2fu = 1.50
self.CyclicLoad = 50.00+1000.0
```

542
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints.extend([(1500/1000.0, 0.0),])

self.studPositionDescriptions.extend([(0.2, 'b2', 3, (3)),
                                     (0.3, 'b2', 3, (3)), (0.62, 'b2', 3, (3)), (0.83, 'b2', 3, (3)),
                                     (1.04, 'b2', 3, (3)), (1.4, 'b2', 3, (3)), (1.6, 'b2', 3, (3)),
                                     (1.8, 'b2', 3, (3)), (2.0, 'b2', 3, (3)), (2.2, 'b2', 3, (3)),
                                     (2.4, 'b2', 3, (3)), (2.6, 'b2', 3, (3)), (2.8, 'b2', 3, (3)),])

self.studPositionDescriptions.extend([(0.305, 't1', 3, (3)), (0.515, 't1', 3, (3)), (1.7, 't1', 3, (3)),])

print self.Case_Name
if self.Case_Name == 'McKinley thesis City10' or self.beamCaseNo==258:

self.L = 3000.00/1000.0
self.beamb = 216.00/1000.0
self.beamb = 600.00/1000.0
self.bottomt = 8.00/1000.0
self.topt = 8.00/1000.0

self.plateFyBottom = 418.00
self.plateFyTop = 418.00
self.plateFuBottom = 547.00
self.plateFuTop = 547.00
self.concretefc = 39.30

self.studb1h = 0.08/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 200.00/1000.0
self.studb2Dia = 25.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 286.94+1000.0
self.studb2fy = 610.00
self.studb2fu = 660.00

self.studt1h = 170.00/1000.0
self.studt1Dia = 19.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 115.41+1000.0
self.studt1fy = 450.00
self.studt1fu = 500.00
self.studt2h = 0.08/1000.0
```

543
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00+1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 200.00/1000.0
self.support1a = 2800.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 1500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.loadDefPoints . extend(((1500/1000.0,0.0),))

self.studPositionDescriptions . extend(((0.2 , 'b2' ,3, (3) ),(0.4 , 'b2' ,3, (3) ),(0.6 , 'b2' ,3, (3) ),(0.8 , 'b2' ,3, (3) ),(1.0 , 'b2' ,3, (3) ),(1.1 , 'b2' ,3, (3) ),(1.2 , 'b2' ,3, (3) ),(1.3 , 'b2' ,3, (3) ),(1.4 , 'b2' ,3, (3) ),(1.5 , 'b2' ,3, (3) ),(1.6 , 'b2' ,3, (3) ),(1.7 , 'b2' ,3, (3) ),(1.8 , 'b2' ,3, (3) ),(1.9 , 'b2' ,3, (3) ),(2.0 , 'b2' ,3, (3) ),(2.1 , 'b2' ,3, (3) ),(2.2 , 'b2' ,3, (3) ),(2.3 , 'b2' ,3, (3) ),(2.4 , 'b2' ,3, (3) ),(2.5 , 'b2' ,3, (3) ),(2.6 , 'b2' ,3, (3) ),(2.7 , 'b2' ,3, (3) ),(2.8 , 'b2' ,3, (3) ),(2.9 , 'b2' ,3, (3) ),(3.0 , 'b2' ,3, (3) ),(3.1 , 'b2' ,3, (3) ),(3.2 , 'b2' ,3, (3) ),(3.3 , 'b2' ,3, (3) )))
self.studPositionDescriptions . extend(((0.3 , 't1' ,3, (3) ),(0.5 , 't1' ,3, (3) ),(0.7 , 't1' ,3, (3) ),(0.9 , 't1' ,3, (3) ),(1.1 , 't1' ,3, (3) ),(1.3 , 't1' ,3, (3) ),(1.5 , 't1' ,3, (3) ),(1.7 , 't1' ,3, (3) ),(1.9 , 't1' ,3, (3) )))

print self.Case_Name
if self.Case_Name == 'Nojiri_2' or self.beamCaseNo==260:

    self.L = 1220.00/1080.0
self.beamh = 312.00/1000.0
self.beamb = 300.00/1000.0
self.bottomt = 6.00/1000.0
self.topt = 6.00/1000.0

self.plateFyBottom = 310.00
self.plateFyTop = 310.00
self.plateFuBottom = 450.00
self.plateFuTop = 450.00
self.concretef = 37.00

self.studb1h = 80.00/1000.0
self.studb1Dia = 16.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 65.95*1000.0
self.studb1fy = 250.00
self.studb1fu = 410.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studth = 80.00/1000.0
```

544
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

self.studt1Dia = 16.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 65.95*1000.0
self.studt1fy = 250.00
self.studt1fu = 410.00
self.studt2h = 156.00/1000.0
self.studt2Dia = 100.00/1000.0
self.studt2FullHeight = True
self.studt2Resistance = 3183.76*1000.0
self.studt2fy = 1000.00
self.studt2fu = 1100.00
self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05
self.support1 = 10.00/1000.0
self.support1a = 610.00/1000.0
self.support1b = 1210.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 310.00/1000.0
self.load1a = 910.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.studPositionDescriptions.extend(((0.085, 'b1', 3, (3)), (0.235, 'b1', 3, (3)), (0.385, 'b1', 3, (3)), (0.535, 'b1', 3, (3)), (0.685, 'b1', 3, (3)), (0.835, 'b1', 3, (3)), (0.985, 'b1', 3, (3)), (1.135, 'b1', 3, (3)), ))
self.studPositionDescriptions.extend(((0.085, 't1', 3, (3)), (0.235, 't1', 3, (3)), (0.385, 't1', 3, (3)), (0.535, 't1', 3, (3)), (0.685, 't1', 3, (3)), (0.835, 't1', 3, (3)), (0.985, 't1', 3, (3)), (1.135, 't1', 3, (3)), ))
self.studPositionDescriptions.extend(((0.011, 't2', 1, (1)), (1.21, 't2', 1, (1)), ))

print self.Case_Name
if self.Case_Name == 'Nojiri_4' or self.beamCaseNo==262:
    self.L = 1520.00/1000.0
    self.beamh = 312.00/1000.0
    self.beamb = 300.00/1000.0
    self.bottomt = 6.00/1000.0
    self.top = 6.00/1000.0
    self.plateFyBottom = 310.00
    self.plateFyTop = 310.00
    self.plateFuBottom = 450.00
    self.plateFuTop = 450.00
    self.concretefc = 37.00
    self.studb1h = 80.00/1000.0
    self.studb1Dia = 16.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 65.95*1000.0
    self.studb1fy = 250.00
    self.studb1fu = 410.00
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

call stud1h = 80.00/1000.0
self.stud1Dia = 16.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 65.95*1000.0
self.stud1fy = 250.00
self.stud1fu = 410.00

self.stud2h = 156.00/1000.0
self.stud2Dia = 100.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 3183.76*1000.0
self.stud2fy = 1000.00
self.stud2fu = 1100.00

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 10.00/1000.0
self.supportla = 760.00/1000.0
self.supportlb = 1510.00/1000.0
self.supportlc = -100.00/1000.0

self.load1 = 385.00/1000.0
self.loadla = 1135.00/1000.0
self.loadlb = -100.00/1000.0
self.loadlc = -100.00/1000.0
self.loadld = -100.00/1000.0
self.loadle = -100.00/1000.0
self.loadlf = -100.00/1000.0
self.loadlg = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactorla = 1.00
self.loadFactorlb = -100.00
self.loadFactorlc = -100.00
self.loadFactorld = -100.00
self.loadFactorle = -100.00
self.loadFactorlf = -100.00
self.loadFactorlg = -100.00

self.loadDefPoints = []

self.studPositionDescriptions.extend(((0.085,‘b1’,3,(3))),(0.235,’b1’,3,(3)),(0.385,’b1’,3,(3)),(0.535,’b1’,3,(3)),(0.685,’b1’,3,(3)),(0.835,’b1’,3,(3)),(0.985,’b1’,3,(3)),(1.135,’b1’,3,(3)),(1.285,’b1’,3,(3)),(1.435,’b1’,3,(3))))

self.studPositionDescriptions.extend(((0.085,’t1’,3,(3)),(0.235,’t1’,3,(3)),(0.385,’t1’,3,(3)),(0.535,’t1’,3,(3)),(0.685,’t1’,3,(3)),(0.835,’t1’,3,(3)),(0.985,’t1’,3,(3)),(1.135,’t1’,3,(3)),(1.285,’t1’,3,(3)),(1.435,’t1’,3,(3))))

self.studPositionDescriptions.extend(((0.011,’t2’,1,(1)),(1.51,’t2’,1,(1))))

print self.Case_Name
if self.Case_Name == ’Nojiri_5’ or self.beamCaseNo==263:
    self.L = 1820.00/1000.0
self.beamb = 306.40/1000.0
self.beamb = 300.00/1000.0
self.bottomt = 3.20/1000.0
self.top = 3.20/1000.0
self.palateFyBottom = 300.00
self.palateFyTop = 300.00
self.palateFUb = 460.00
```

546
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

self.plateFuTop = 460.00
self.concretefc = 37.00

self.studb1h = 80.00/1000.0
self.studb1Dia = 16.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 65.95*1000.0
self.studb1fy = 250.00
self.studb1fu = 410.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 80.00/1000.0
self.studt1Dia = 16.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 65.95*1000.0
self.studt1fy = 250.00
self.studt1fu = 410.00

self.studt2h = 153.20/1000.0
self.studt2Dia = 100.00/1000.0
self.studt2FullHeight = True
self.studt2Resistance = 3183.76*1000.0
self.studt2fy = 1000.00
self.studt2fu = 1100.00

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 10.00/1000.0
self.support1a = 910.00/1000.0
self.support1b = 1810.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 460.00/1000.0
self.load1a = 1360.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []

self.studPositionDescriptions.extend(((0.085, 'b1', 3,(3)),(0.235, 'b1', 3,(3)),(0.385, 'b1', 3,(3)),(0.535, 'b1', 3,(3)),(0.685, 'b1', 3,(3)),(0.835, 'b1', 3,(3)),(0.985, 'b1', 3,(3)),(1.135, 'b1', 3,(3)),(1.285, 'b1', 3,(3)),(1.435, 'b1', 3,(3)),(1.585, 'b1', 3,(3)),(1.735, 'b1', 3,(3)),))
self.studPositionDescriptions.extend(((0.085, 't1', 3,(3)),(0.235, 't1', 3,(3)),(0.385, 't1', 3,(3)),(0.535, 't1', 3,(3)),(0.685, 't1', 3,(3)),(0.835, 't1', 3,(3)),(0.985, 't1', 3,(3)),(1.135, 't1', 3,(3)),(1.285, 't1', 3,(3)),(1.435, 't1', 3,(3)),(1.585, 't1', 3,(3)),(1.735, 't1', 3,(3)),))
self.studPositionDescriptions.extend(((0.011, 't2', 1,(1)),(1.81, 't2', 1,(1)),))

print self.Case_Name
if self.Case_Name == 'Nojiri_b' or self.beamCaseNo==264:
Appendix B: Test Database in Python Code Format

```python
self.L = 1820.00/1000.0
self.beamh = 312.00/1000.0
self.beamb = 300.00/1000.0
self.bottomt = 6.00/1000.0
self.top = 6.00/1000.0

self.plateFyBottom = 310.00
self.plateFyTop = 310.00
self.plateFuBottom = 450.00
self.plateFuTop = 450.00
self.concretefc = 37.00

self.stud1h = 80.00/1000.0
self.stud1Dia = 16.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 65.95*1000.0
self.stud1fy = 250.00
self.stud1fu = 410.00

self.stud2h = 0.00/1000.0
self.stud2Dia = 0.00/1000.0
self.stud2FullHeight = False
self.stud2Resistance = 0.00*1000.0
self.stud2fy = 1.00
self.stud2fu = 1.50

self.stud3h = 80.00/1000.0
self.stud3Dia = 16.00/1000.0
self.stud3FullHeight = False
self.stud3Resistance = 65.95*1000.0
self.stud3fy = 250.00
self.stud3fu = 410.00

self.stud4h = 156.00/1000.0
self.stud4Dia = 100.00/1000.0
self.stud4FullHeight = True
self.stud4Resistance = 3183.76*1000.0
self.stud4fy = 1000.00
self.stud4fu = 1100.00

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.50

self.support1 = 10.00/1000.0
self.support1a = 910.00/1000.0
self.support1b = 1810.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 460.00/1000.0
self.load1a = 1360.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
```

548
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studyPositionDescriptions.extend(((0.11, 'b1', 3, (3)), (0.26, 'b1', 3, (3)), (0.685, 'b1', 3, (3)), (0.835, 'b1', 3, (3)),
(0.985, 'b1', 3, (3)), (1.135, 'b1', 3, (3)), (1.56, 'b1', 3, (3)), (1.71, 'b1', 3, (3)),))

self.studyPositionDescriptions.extend(((0.011, 't1', 3, (3)), (0.026, 't1', 3, (3)), (0.835, 't1', 3, (3)), (0.985, 't1', 3, (3)),
(1.135, 't1', 3, (3)), (1.56, 't1', 3, (3)), (1.71, 't1', 3, (3)),))

self.studyPositionDescriptions.extend(((0.011, 't2', 1, (1)), (1.81, 't2', 1, (1)),))

print self Case_Name
if self.Case_Name == 'Nojiri_7' or self.beamCaseNo==265:

    self.L = 1820.00/1000.0
    self.beamb = 324.00/1000.0
    self.beamb = 300.00/1000.0
    self.bottomt = 12.00/1000.0
    self.topt = 12.00/1000.0
    self.plateFyBottom = 300.00
    self.plateFyTop = 300.00
    self.concretefc = 37.00
    self.studb1h = 80.00/1000.0
    self.studb1Dia = 16.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 65.95*1000.0
    self.studb1fy = 250.00
    self.studb1fu = 410.00
    self.studb2h = 0.00/1000.0
    self.studb2Dia = 0.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 0.00*1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50
    self.studt1h = 80.00/1000.0
    self.studt1Dia = 16.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 65.95*1000.0
    self.studt1fy = 250.00
    self.studt1fu = 410.00
    self.studt2h = 162.00/1000.0
    self.studt2Dia = 100.00/1000.0
    self.studt2FullHeight = True
    self.studt2Resistance = 3183.76*1000.0
    self.studt2fy = 1000.00
    self.studt2fu = 1100.00
    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05
    self.support1 = 10.00/1000.0
    self.support1a = 910.00/1000.0
    self.support1b = 1810.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 460.00/1000.0
    self.load1a = 1360.00/1000.0
    self.load1b = -100.00/1000.0
    self.load1c = -100.00/1000.0
    self.load1d = -100.00/1000.0
    self.load1e = -100.00/1000.0
    self.load1f = -100.00/1000.0
    self.load1g = -100.00/1000.0
    self.loadFactor1 = 1.00
    self.loadFactor1a = 1.00
    self.loadFactor1b = -100.00
    self.loadFactor1c = -100.00
```

549
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```
self.loadFactorId = -100.00
self.loadFactorle = -100.00
self.loadFactorlf = -100.00
self.loadFactorlg = -100.00

self.loadDefPoints = []

self.studPositionDescriptions.extend(((0.085, 'b1', 3, (3)), (0.235, 'b1', 3, (3)), (0.385, 'b1', 3, (3)), (0.535, 'b1', 3, (3)), (0.685, 'b1', 3, (3)), (0.835, 'b1', 3, (3)), (0.985, 'b1', 3, (3)), (1.135, 'b1', 3, (3)), (1.285, 'b1', 3, (3)), (1.435, 'b1', 3, (3)), (1.585, 'b1', 3, (3)), (1.735, 'b1', 3, (3)), ))
self.studPositionDescriptions.extend(((0.085, 't1', 3, (3)), (0.235, 't1', 3, (3)), (0.385, 't1', 3, (3)), (0.535, 't1', 3, (3)), (0.685, 't1', 3, (3)), (0.835, 't1', 3, (3)), (0.985, 't1', 3, (3)), (1.135, 't1', 3, (3)), (1.285, 't1', 3, (3)), (1.435, 't1', 3, (3)), (1.585, 't1', 3, (3)), (1.735, 't1', 3, (3)), ))
self.studPositionDescriptions.extend(((0.011, 't2', 1, (1)), (1.81, 't2', 1, (1)), ))

print self.Case_Name
if self.Case_Name == 'Nojiri_8' or self.beamCaseNo==266:

    self.L = 1820.00/1000.0
    self.beamb = 312.00/1000.0
    self.beamb = 300.00/1000.0
    self.bottomt = 6.00/1000.0
    self.topf = 6.00/1000.0
    self.plateFyBottom = 310.00
    self.plateFyTop = 310.00
    self.plateFuBottom = 450.00
    self.plateFuTop = 450.00
    self.concretec = 37.00
    self.studb1h = 240.00/1000.0
    self.studb1Dia = 16.00/1000.0
    self.studb1FullHeight = False
    self.studb1Resistance = 65.95*1000.0
    self.studb1fy = 250.00
    self.studb1fu = 410.00
    self.studb2h = 0.00/1000.0
    self.studb2Dia = 0.00/1000.0
    self.studb2FullHeight = False
    self.studb2Resistance = 0.00*1000.0
    self.studb2fy = 1.00
    self.studb2fu = 1.50
    self.studt1h = 240.00/1000.0
    self.studt1Dia = 16.00/1000.0
    self.studt1FullHeight = False
    self.studt1Resistance = 65.95*1000.0
    self.studt1fy = 350.00
    self.studt1fu = 410.00
    self.studt2h = 156.00/1000.0
    self.studt2Dia = 100.00/1000.0
    self.studt2FullHeight = True
    self.studt2Resistance = 3183.76*1000.0
    self.studt2fy = 1000.00
    self.studt2fu = 1100.00
    self.CyclicLoad = 50.00*1000.0
    self.appliedDeflection = 0.05
    self.support1 = 10.00/1000.0
    self.support1a = 910.00/1000.0
    self.support1b = 1810.00/1000.0
    self.support1c = -100.00/1000.0
    self.load1 = 460.00/1000.0
    self.load1a = 1360.00/1000.0
    self.load1b = -100.00/1000.0

```
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = 1.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []

self.setPositionDescriptions.extend(((0.16, 'b1', 3,(3)) , (0.31, 'b1', 3,(3)) , (0.46, 'b1', 3,(3)) , (0.61, 'b1', 3,(3)) , (0.76, 'b1', 3,(3)) , (0.91, 'b1', 3,(3)) , (1.06, 'b1', 3,(3)) , (1.21, 'b1', 3,(3)) , (1.36, 'b1', 3,(3)) , (1.51, 'b1', 3,(3)) , (1.66, 'b1', 3,(3)) ,))
self.setPositionDescriptions.extend(((0.085, 't1', 3,(3)) , (0.235, 't1', 3,(3)) , (0.385, 't1', 3,(3)) , (0.535, 't1', 3,(3)) , (0.685, 't1', 3,(3)) , (0.835, 't1', 3,(3)) , (0.985, 't1', 3,(3)) , (1.135, 't1', 3,(3)) , (1.285, 't1', 3,(3)) , (1.435, 't1', 3,(3)) , (1.585, 't1', 3,(3)) , (1.735, 't1', 3,(3)) ,))
self.setPositionDescriptions.extend(((0.011, 't2', 1,(1)) , (1.81, 't2', 1,(1)) ,))

print self.Case_Name
if self.Case_Name == 'Njr1_14' or self.beamCaseNo==272:

self.L = 1000.00/1000.0
self.beamh = 384.00/1000.0
self.beamb = 300.00/1000.0
self.bottomt = 12.00/1000.0
self.top = 12.00/1000.0
self.plateFyBottom = 300.00
self.plateFyTop = 300.00
self.plateFuBottom = 450.00
self.plateFuTop = 450.00
self.concretefc = 39.00

self.studb1h = 80.00/1000.0
self.studb1Dia = 16.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 65.95*1000.0
self.studb1fy = 250.00
self.studb1fu = 410.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00*1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 80.00/1000.0
self.studt1Dia = 16.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 65.95*1000.0
self.studt1fy = 250.00
self.studt1fu = 410.00

self.studt2h = 192.00/1000.0
self.studt2Dia = 100.00/1000.0
self.studt2FullHeight = True
self.studt2Resistance = 3294.59*1000.0
self.studt2fy = 1000.00
self.studt2fu = 1100.00
self.CyclicLoad = 50.00*1000.0
```

551
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.appliedDeflection = 0.05
self.support1 = 50.00/1000.0
self.supportla = 950.00/1000.0
self.supportlb = -100.00/1000.0
self.supportlc = -100.00/1000.0
self.load1 = 500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []

self.studPositionDescriptions.extend(((0.15, 'b1', 3, (3)), (0.3, 'b1', 3, (3)), (0.725, 'b1', 3, (3)), (0.875, 'b1', 3, (3)),))
self.studPositionDescriptions.extend(((0.15, 't1', 3, (3)), (0.3, 't1', 3, (3)), (0.725, 't1', 3, (3)), (0.875, 't1', 3, (3))))
self.studPositionDescriptions.extend(((0.051, 't2', 1, (1)), (0.95, 't2', 1, (1)),))

print self.Case_Name
if self.Case_Name == 'Nojiiri_15' or self.beamCaseNo==273:

self.L = 1000.00/1000.0
self.bamb = 384.00/1000.0
self.beamb = 900.00/1000.0
self.bottomt = 12.00/1000.0
self.topt = 12.00/1000.0
self.plateFyBottom = 300.00
self.plateFyTop = 300.00
self.plateFuBottom = 450.00
self.plateFuTop = 450.00
self.concretetc = 39.00
self.studb1h = 80.00/1000.0
self.studb1Dia = 16.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 65.95+1000.0
self.studb1fy = 250.00
self.studb1fu = 410.00
self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00+1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50
self.studt1h = 80.00/1000.0
self.studt1Dia = 16.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 65.95+1000.0
self.studt1fy = 250.00
self.studt1fu = 410.00
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud2h = 192.00/1000.0
self.stud2Dia = 100.00/1000.0
self.stud2FullHeight = True
self.stud2Resistance = 3294.59*1000.0
self.stud2fy = 1000.00
self.stud2fu = 1100.00
self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05
self.support1 = 50.00/1000.0
self.support1a = 950.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []
self.studPositionDescriptions.extend(((0.1,'b1',9,(9)),(0.25,'b1',9,(9)),(0.4,'b1',9,(9)),(0.55,'b1',9,(9)),(0.7,'b1',9,(9)),(0.85,'b1',9,(9)),))
self.studPositionDescriptions.extend(((0.1,'t1',9,(9)),(0.25,'t1',9,(9)),(0.4,'t1',9,(9)),(0.55,'t1',9,(9)),(0.7,'t1',9,(9)),(0.85,'t1',9,(9)),))
self.studPositionDescriptions.extend(((0.051,'t2',1,(1)),(0.95,'t2',1,(1)),))
print self.Case_Name
if self.Case_Name == 'Nojiri_17' or self.beamCaseNo==275:
    self.L = 1000.00/1000.0
    self.beamh = 378.00/1000.0
    self.beamb = 900.00/1000.0
    self.bottomt = 9.00/1000.0
    self.top = 9.00/1000.0
    self.plateFyBottom = 420.00
    self.plateFyTop = 420.00
    self.plateFuBottom = 510.00
    self.plateFuTop = 510.00
    self.concretefc = 39.00
    self.stud1h = 0.00/1000.0
    self.stud1Dia = 0.00/1000.0
    self.stud1FullHeight = False
    self.stud1Resistance = 0.00+1000.0
    self.stud1fy = 1.00
    self.stud1fu = 1.50
    self.stud2h = 0.00/1000.0
    self.stud2Dia = 0.00/1000.0
    self.stud2FullHeight = False
    self.stud2Resistance = 0.00+1000.0
    self.stud2fy = 1.00
    self.stud2fu = 1.50
```

553
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.stud1h = 0.00/1000.0
self.stud1Dia = 0.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 0.00+1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50

self.stud2h = 189.00/1000.0
self.stud2Dia = 100.00/1000.0
self.stud2FullHeight = True
self.stud2Resistance = 3294.59+1000.0
self.stud2fy = 100.00
self.stud2fu = 1100.00

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 50.00/1000.0
self.support1a = 950.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 500.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []

self.studPositionDescriptions.extend(((0.051, 't2', 1,(1)) , (0.95, 't2', 1,(1)) ,))

self.studpositiondescriptions, extend(((0.051, 't2', 1,(1)) , (0.95, 't2', 1,(1)) ,))

print self.Case_Name
if self.Case_Name == 'Nojiri_18' or self.beamCaseNo==276:
  self.L = 1000.00/1000.0
  self.beamh = 378.00/1000.0
  self.beamb = 900.00/1000.0
  self.bottomt = 9.00/1000.0
  self.topt = 9.00/1000.0
  self.plateFyBottom = 420.00
  self.plateFyTop = 420.00
  self.plateFuBottom = 510.00
  self.plateFuTop = 510.00
  self.concretesfc = 39.00
  self.studb1h = 0.00/1000.0
  self.studb1Dia = 0.00/1000.0
  self.studb1FullHeight = False
  self.studb1Resistance = 0.00+1000.0
  self.studb1fy = 1.00
  self.studb1fu = 1.50
  self.studb2h = 0.00/1000.0
  self.studb2Dia = 0.00/1000.0

554
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb2FullHeight = False
self.studb2Resistance = 0.00 * 1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 0.00 / 1000.0
self.studt1Dia = 0.00 / 1000.0
self.studt1FullHeight = False
self.studt1Resistance = 0.00 * 1000.0
self.studt1fy = 1.00
self.studt1fu = 1.50

self.studt2h = 189.00 / 1000.0
self.studt2Dia = 100.00 / 1000.0
self.studt2FullHeight = True
self.studt2Resistance = 3294.59 * 1000.0
self.studt2fy = 1000.00
self.studt2fu = 1100.00

self.CyclicLoad = 50.00 * 1000.0
self.appliedDeflection = 0.05

self.support1 = 50.00 / 1000.0
self.support1a = 950.00 / 1000.0
self.support1b = -100.00 / 1000.0
self.support1c = -100.00 / 1000.0

self.load1 = 500.00 / 1000.0
self.load1a = -100.00 / 1000.0
self.load1b = -100.00 / 1000.0
self.load1c = -100.00 / 1000.0
self.load1d = -100.00 / 1000.0
self.load1e = -100.00 / 1000.0
self.load1f = -100.00 / 1000.0
self.load1g = -100.00 / 1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []

self.studPositionDescriptions.extend(((0.051, 't2', 1, (1)), (0.95, 't2', 1, (1))))

print self.Case_Name
if self.Case_Name == 'Noriiri_20' or self.beamCaseNo==278:

self.L = 1600.00 / 1000.0
self.beamb = 378.00 / 1000.0
self.beamb = 900.00 / 1000.0
self.bottomt = 9.00 / 1000.0
self.topt = 9.00 / 1000.0

self.plateFyBottom = 420.00
self.plateFyTop = 420.00
self.plateFullBottom = 510.00
self.plateFuTop = 510.00
self.concretef = 39.00

self.studblh = 0.00 / 1000.0
self.studblDia = 0.00 / 1000.0
self.studbFullHeight = False
self.studbResistance = 0.00 * 1000.0
self.studbfy = 1.00

```

555
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studb1fu = 1.50
self.studb2h = 180.00/1000.0
self.studb2Dia = 7.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 16.14*1000.0
self.studb2fy = 480.0
self.studb2fu = 530.00
self.stud1h = 0.08/1000.0
self.stud1Dia = 0.00/1000.0
self.stud1FullHeight = False
self.stud1Resistance = 0.00+1000.0
self.stud1fy = 1.00
self.stud1fu = 1.50

self.stud2h = 189.00/1000.0
self.stud2Dia = 100.00/1000.0
self.stud2FullHeight = True
self.stud2Resistance = 3294.59*1000.0
self.stud2fy = 1000.00
self.stud2fu = 1100.00
self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 50.00/1000.0
self.support1a = 1550.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0
self.load1 = 800.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0
self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00
self.loadDefPoints = []

self.studPositionDescriptions.extend(((0.125, 'b2', 9, (9)), (0.2, 'b2', 9, (9)), (0.275, 'b2', 9, (9)), (0.35, 'b2', 9, (9)), (0.425, 'b2', 9, (9)), (0.5, 'b2', 9, (9)), (0.575, 'b2', 9, (9)), (0.65, 'b2', 9, (9)), (0.725, 'b2', 9, (9)), (0.875, 'b2', 9, (9)), (0.95, 'b2', 9, (9)), (1.025, 'b2', 9, (9)), (1.1, 'b2', 9, (9)), (1.175, 'b2', 9, (9)), (1.25, 'b2', 9, (9)), (1.325, 'b2', 9, (9)), (1.4, 'b2', 9, (9)), (1.475, 'b2', 9, (9)), (1.55, 'b2', 9, (9)), (1.625, 'b2', 9, (9))))
self.studPositionDescriptions.extend(((0.5, 't2', 1, (1)), (1.55, 't2', 1, (1)),))

print self.Case_Name
if self.Case_Name == 'Nojiri_21' or self.beamCaseNo==279:

self.L = 1600.00/1000.0
self.beamb = 378.00/1000.0
self.beamb = 900.00/1000.0
self.bottomt = 9.00/1000.0
self.topT = 9.00/1000.0
self.plateFyBottom = 420.00
self.plateFyTop = 420.00
self.plateFyBottom = 510.00
```

556
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.plateFuTop = 510.00
self.concretefc = 39.00

self.studb1h = 0.08/1000.0
self.studb1Dia = 0.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 0.00+1000.0
self.studb1fy = 1.00
self.studb1fu = 1.50

self.studb2h = 180.00/1000.0
self.studb2Dia = 10.00/1000.0
self.studb2FullHeight = True
self.studb2Resistance = 32.95+1000.0
self.studb2fy = 480.00
self.studb2fu = 530.00

self.studt1h = 0.00/1000.0
self.studt1Dia = 0.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 0.00
self.studt1fy = 1.00
self.studt1fu = 1.50

self.studt2h = 189.00/1000.0
self.studt2Dia = 100.00/1000.0
self.studt2FullHeight = True
self.studt2Resistance = 3294.59+1000.0
self.studt2fy = 1000.00
self.studt2fu = 1100.00

self.CyclicLoad = 50.00+1000.0
self.appliedDeflection = 0.05

self.support1 = 50.00/1000.0
self.support1a = 1550.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 800.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []
self.studPositionDescriptions.extend(((0.125, 'b2', 9, (9)), (0.2, 'b2', 9, (9)), (0.275, 'b2', 9, (9)), (0.35, 'b2', 9, (9)), (0.425, 'b2', 9, (9)), (0.5, 'b2', 9, (9)), (0.575, 'b2', 9, (9)), (0.65, 'b2', 9, (9)), (0.725, 'b2', 9, (9)), (0.875, 'b2', 9, (9)), (0.95, 'b2', 9, (9)), (1.025, 'b2', 9, (9)), (1.1, 'b2', 9, (9)), (1.175, 'b2', 9, (9)), (1.25, 'b2', 9, (9)), (1.325, 'b2', 9, (9)), (1.4, 'b2', 9, (9)), (1.475, 'b2', 9, (9)))))
self.studPositionDescriptions.extend(((0.051, 't2', 1, (1)), (1.55, 't2', 1, (1)))))

print self.Case_Name
if self.Case_Name == 'Nojiri_24' or self.beamCaseNo==282:

self.L = 1600.00/1000.0
```
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.beamh = 258.00/1000.0
self.beamb = 600.00/1000.0
self.bottomt = 9.00/1000.0
self.topt = 9.00/1000.0

self.plateFyBottom = 250.00
self.plateFyTop = 250.00
self.plateFuBottom = 250.00
self.plateFuTop = 250.00
self.concretefc = 38.00

self.studb1h = 288.00/1000.0
self.studb1Dia = 10.00/1000.0
self.studb1FullHeight = False
self.studb1Resistance = 32.39*1000.0
self.studb1fy = 370.00
self.studb1fu = 520.00

self.studb2h = 0.00/1000.0
self.studb2Dia = 0.00/1000.0
self.studb2FullHeight = False
self.studb2Resistance = 0.00+1000.0
self.studb2fy = 1.00
self.studb2fu = 1.50

self.studt1h = 288.00/1000.0
self.studt1Dia = 10.00/1000.0
self.studt1FullHeight = False
self.studt1Resistance = 32.39+1000.0
self.studt1fy = 370.00
self.studt1fu = 520.00

self.studt2h = 189.00/1000.0
self.studt2Dia = 100.00/1000.0
self.studt2FullHeight = True
self.studt2Resistance = 3239.43*1000.0
self.studt2fy = 1000.00
self.studt2fu = 1100.00

self.CyclicLoad = 50.00*1000.0
self.appliedDeflection = 0.05

self.support1 = 50.00/1000.0
self.support1a = 1550.00/1000.0
self.support1b = -100.00/1000.0
self.support1c = -100.00/1000.0

self.load1 = 800.00/1000.0
self.load1a = -100.00/1000.0
self.load1b = -100.00/1000.0
self.load1c = -100.00/1000.0
self.load1d = -100.00/1000.0
self.load1e = -100.00/1000.0
self.load1f = -100.00/1000.0
self.load1g = -100.00/1000.0

self.loadFactor1 = 1.00
self.loadFactor1a = -100.00
self.loadFactor1b = -100.00
self.loadFactor1c = -100.00
self.loadFactor1d = -100.00
self.loadFactor1e = -100.00
self.loadFactor1f = -100.00
self.loadFactor1g = -100.00

self.loadDefPoints = []

self.studPositionDescriptions.extend(((0.125, 'b1', 6, (6)), (0.2, 'b1', 6, (6)), (0.275, 'b1', 6, (6)), (0.35, 'b1', 6, (6)), (0.425, 'b1', 6, (6)), (0.5, 'b1', 6, (6)), (0.575, 'b1', 6, (6)), (0.65, 'b1', 6, (6)), (0.725, 'b1', 6, (6)), (0.757, 'b1', 6, (6)), (0.875, 'b1', 6, (6)), (0.95, 'b1', 6, (6)), (1.025, 'b1', 6, (6)), (1.1, 'b1', 6, (6)), (1.175, 'b1', 6, (6)), (1.25, 'b1', 6, (6)), (1.325, 'b1', 6, (6)), (1.375, 'b1', 6, (6))
```

558
APPENDIX B: TEST DATABASE IN PYTHON CODE FORMAT

```python
self.studPositionDescriptions.extend([(0.125, 't1', 6, (6)), (0.2, 't1', 6, (6)), (0.275, 't1', 6, (6)), (0.35, 't1', 6, (6)),
(0.425, 't1', 6, (6)), (0.5, 't1', 6, (6)), (0.575, 't1', 6, (6)), (0.65, 't1', 6, (6)), (0.725, 't1', 6, (6)), (0.875, 't1', 6, (6)),
(0.95, 't1', 6, (6)), (1.025, 't1', 6, (6)), (1.1, 't1', 6, (6)), (1.175, 't1', 6, (6)), (1.25, 't1', 6, (6)), (1.325, 't1', 6, (6))]
self.studPositionDescriptions.extend([(0.051, 't2', 1, (1)), (1.55, 't2', 1, (1))])

print self.Case_Name
```

APPENDIX C

Python code for assembling parameters used for modelling an SCS beam using ABAQUS

This appendix presents code to translate the model data presented in Appendix B into a form that can used with the code presented in Appendix A

```python
Areas.append(('401_Outline',((0.0,0.0),(self.L,0.0),(self.L,self.beamh),(0.0,self.beamh))))
Areas.append(('402_Concrete',((0.0,self.bottomt),(self.L,self.bottomt),(self.L,self.beamh-self.top),(0.0,self.beamh-self.top))))

Lines.append(('601_Beam Interactions',((0.0,self.bottomt),(self.L,self.bottomt),)))
Lines.append(('601_Beam Interactions',((0.0,self.beamh-self.top),(self.L,self.beamh-self.top),)))

Lines.append(('551_Supports',((self.support1,0),(self.support1,-0.1),)))
Lines.append(('551_Supports',((self.support1a,0.0),(self.support1a,-0.1),)))
Lines.append(('501_Load 1',((self.load1,self.beamh),(self.load1,self.beamh+0.1),)))

if self.load1a > 0.0:
    Lines.append(('502_Load 2',((self.load1a,self.beamh),(self.load1a,self.beamh+0.1),)))
if self.load1b > 0.0:
    Lines.append(('503_Load 3',((self.load1b,self.beamh),(self.load1b,self.beamh+0.1),)))
if self.load1c > 0.0:
    Lines.append(('504_Load 4',((self.load1c,self.beamh),(self.load1c,self.beamh+0.1),)))
if self.load1d > 0.0:
    Lines.append(('505_Load 5',((self.load1d,self.beamh),(self.load1d,self.beamh+0.1),)))
if self.load1e > 0.0:
    Lines.append(('506_Load 6',((self.load1e,self.beamh),(self.load1e,self.beamh+0.1),)))
if self.load1f > 0.0:
    Lines.append(('507_Load 7',((self.load1f,self.beamh),(self.load1f,self.beamh+0.1),)))
if self.load1g > 0.0:
    Lines.append(('508_Load 8',((self.load1g,self.beamh),(self.load1g,self.beamh+0.1),)))

Lines.append(('602_Load Interactions',((0.0,0.0),(self.L,0.0),)))
Lines.append(('602_Load Interactions',((0.0,self.beamh),(self.L,self.beamh),)))

self.plateFy1 = self.plateFyBottom
self.plateFu1 = self.plateFuBottom

#Top plate is grade 2
self.plateFy2 = self.plateFyTop
self.plateFu2 = self.plateFuTop
self.plateE2 = 210.0e9
```

560
APPENDIX C: PYTHON CODE FOR ASSEMBLING PARAMETERS USED FOR MODELLING AN SCS BEAM USING ABAQUS

```python
#3 is nothing in these cases
self.plateFy3 = 1.0
self.plateFu3 = 1.0
self.plateE3 = 1.0

Areas.append(('351_Steel Grade 2', ((0.0, self.beamh-self.topt), (self.L, self.beamh-self.topt), (self.L, self.beamh), (0.0, self.beamh))))

for ld in self.loadDefPoints:
    Lines.append(('561_LoadDefs', ((ld[0],ld[1]),(ld[0],ld[0]+0.1))))

for stud in self.studPositionDescriptions:
    print stud
    line = ''
    FullHeight = 1
    if len(self.studPositions) < 5000:
        if stud[1] == 'b1':
            if self.studb1FullHeight == False:
                line = ('Layer',((stud[0],self.bottomt+studb1th),(stud[0],self.bottomt),(stud[0],self.bottomt/2.0),))
                Lines.append(('302_Steel Lines',((stud[0]-0.01,0.0),(stud[0]-0.01,self.beamh/2.0),)))
                Lines.append(('302_Steel Lines',((stud[0]+0.01,0.0),(stud[0]+0.01,self.beamh/2.0),)))
            else:
                line = ('Layer',((stud[0],self.bottomt/2.0),(stud[0],self.bottomt),(stud[0],self.beamh-self.topt),
                (stud[0],self.beamh-self.topt/2.0))))
        FullHeight = 2
        elif stud[1] == 'b2':
            if self.studb2FullHeight == False:
                line = ('Layer',((stud[0],self.bottomt+studb2h),(stud[0],self.bottomt),(stud[0],self.bottomt/2.0),))
                Lines.append(('302_Steel Lines',((stud[0]-0.01,0.0),(stud[0]-0.01,self.beamh/2.0),)))
                Lines.append(('302_Steel Lines',((stud[0]+0.01,0.0),(stud[0]+0.01,self.beamh/2.0),)))
            else:
                line = ('Layer',((stud[0],self.bottomt/2.0),(stud[0],self.bottomt),(stud[0],self.beamh-self.topt),
                (stud[0],self.beamh-self.topt/2.0))))
        FullHeight = 2
        elif stud[1] == 't1':
            if self.studt1FullHeight == False:
                line = ('Layer',((stud[0],self.beamh-(self.topt+studt1h)),(stud[0],self.beamh-(self.topt)),
                (stud[0],self.beamh-(self.topt/2.0),)))
                Lines.append(('302_Steel Lines',((stud[0]-0.01, self.beamh/2.0),(stud[0]-0.01,self.beamh),)))
                Lines.append(('302_Steel Lines',((stud[0]+0.01, self.beamh/2.0),(stud[0]+0.01,self.beamh),)))
            else:
                line = ('Layer',((stud[0],self.bottomt/2.0),(stud[0],self.bottomt),(stud[0],self.beamh-self.topt),
                (stud[0],self.beamh-self.topt/2.0))))
        FullHeight = 2
        elif stud[1] == 't2':
            if self.studt2FullHeight == False:
                line = ('Layer',((stud[0],self.beamh-(self.topt+studt2h)),(stud[0],self.beamh-(self.topt)),
                (stud[0],self.beamh-(self.topt/2.0),)))
                Lines.append(('302_Steel Lines',((stud[0]-0.01, self.beamh/2.0),(stud[0]-0.01,self.beamh),)))
                Lines.append(('302_Steel Lines',((stud[0]+0.01, self.beamh/2.0),(stud[0]+0.01,self.beamh),)))
            else:
                line = ('Layer',((stud[0],self.bottomt/2.0),(stud[0],self.bottomt),(stud[0],self.beamh-self.topt),
                (stud[0],self.beamh-self.topt/2.0))))
        FullHeight = 2
    if line != '':
        print line[1]
        #if FullHeight == 2:
        self.studPositions.append((line[1],stud[1],stud[2],FullHeight))
```
APPENDIX C: PYTHON CODE FOR ASSEMBLING PARAMETERS USED FOR MODELLING AN SCS BEAM USING ABAQUS

```python
if FullHeight == 2:
    Lines.append(('302_Steel Lines', ((stud[0]-0.01, 0.0), (stud[0]-0.01, self.beamh),)))
    Lines.append(('302_Steel Lines', ((stud[0]+0.01, 0.0), (stud[0]+0.01, self.beamh),)))
else:
    self.studPositions.append((line[1], stud[1], stud[2], FullHeight))

print self.studPositions
self.Areas = Areas
self.Lines = Lines

if self.load1g > 0.0:
    self.LoadArrangement = '8 Point Loads'
    Lines.append(('561_LoadDefs', ((self.load1c+ self.load1d) / 2.0, 0.0), ((self.load1c+ self.load1d) / 2.0, -0.01),)))
elif self.load1c > 0.0:
    self.LoadArrangement = '4 Point Loads'
elif self.load1b > 0.0:
    self.LoadArrangement = '3 Point Loads'
elif self.load1a > 0.0:
    self.LoadArrangement = '2 Point Loads'
elif self.load1 > 0.0:
    self.LoadArrangement = '1 Point Load'

if self.L < 3.0:
    self.supportWidth = self.supportWidth / 5.0
    #Reduce support width for very small beams
```