The TDHF Code Sky3D Version 1.1

B. Schuetrumpf*

National Superconducting Cyclotron Laboratory
Michigan State University, East Lansing, Michigan 48824, USA

P.-G. Reinhard

Institut für Theoretische Physik II, Universität Erlangen-Nürnberg,
Staudtstrasse 7, 91058 Erlangen, Germany

P. D. Stevenson

Department of Physics, University of Surrey, Guildford, Surrey, GU2 7XH, United Kingdom

A. S. Umar

Department of Physics and Astronomy, Vanderbilt University,
Nashville, Tennessee 37235, USA

J. A. Maruhn**

Institut für Theoretische Physik, Goethe-Universität, Max-von-Laue-Str. 1,
60438 Frankfurt am Main, Germany

Abstract

The nuclear mean-field model based on Skyrme forces or related density functionals has found widespread application to the description of nuclear ground states, collective vibrational excitations, and heavy-ion collisions. The code Sky3D solves the static or dynamic equations on a three-dimensional Cartesian mesh with isolated or periodic boundary conditions and no further symmetry assumptions. Pairing can be included in the BCS approximation for the static case. The code is implemented with a view to allow easy modifications for including additional physics or special analysis of the results.

Keywords: Hartree-Fock; BCS; Density-functional theory; Skyrme energy functional; Giant Resonances; Heavy-Ion collisions.

NEW VERSION PROGRAM SUMMARY PROGRAM SUMMARY

Title: The TDHF Code Sky3D Version 1.1

Authors: B. Schuetrumpf, P.-G. Reinhard, P. D. Stevenson, A. S. Umar, and J. A. Maruhn,

*Present address: Gesellschaft für Schwerionenforschung, 64291 Darmstadt, Germany

**Corresponding author.

Email addresses: b.schuetrumpf@gsi.de (B. Schuetrumpf), Paul-Gerhard.Reinhard@physik.uni-erlangen.de (P.-G. Reinhard), p.stevenson@surrey.ac.uk (P. D. Stevenson), sait.a.umar@Vanderbilt.Edu (A. S. Umar), maruhn@th.physik.uni-frankfurt.de (J. A. Maruhn)

Program Title: Sky3D
Journal Reference:
Catalogue identifier:
Licensing provisions: none
Programming language: Fortran 90. The `OpenMP` version requires a relatively recent compiler; it was found to work using `gfortran 4.6.2` or later and the Intel compiler version 12 or later.
Computer: All computers with a Fortran compiler supporting at least Fortran 90.
Operating system: All operating systems with such a compiler. Some of the Makefiles and scripts depend on a Unix-like system and need modification under Windows.
RAM: 1 GB
Number of processors used: no built-in limit, parallelization using both OpenMP and MPI.
Supplementary material: Extensive documentation and a number of utility programs to analyze the results and prepare them for graphics output using the Silo library (`http://wci.llnl.gov/simulation/computer-codes/silo`) for use in VisIT [1] or Paraview (`https://www.paraview.org`). The code can serve as a template for interfacing to other database or graphics systems.
Keywords: Nuclear theory, Mean-field models, Nuclear reactions
Classification: 17.16 Theoretical Methods - General, 17.22 Hartree-Fock Calculations, 17.23 Fission and Fusion Processes
External routines/libraries: LAPACK, FFTW3.
Catalogue identifier of previous version:
AESW_v1_0.
Journal reference of previous version:
J. A. Maruhn, P.-G. Reinhard, P. D. Stevenson, and A. S. Umar, "The TDHF Code Sky3D", Comp. Phys. Comm. **185**, 2195 (2014).
Does the new version supersede the previous version?:
Yes.

Nature of problem: The time-dependent Hartree-Fock equations can be used to simulate nuclear vibrations and collisions between nuclei for low energies. This code implements the equations based on a Skyrme energy functional and also allows the determination of the ground-state structure of nuclei through the static version of the equations. For the case of vibrations the principal aim is to calculate the excitation spectra by Fourier-analyzing the time dependence of suitable observables. In collisions, the formation of a neck between nuclei, the dissipation of energy from collective motion, processes like charge transfer and the approach to fusion are of principal interest.

Solution method: The nucleonic wave function spinors are represented on a three-dimensional Cartesian mesh with no further symmetry restrictions. The boundary conditions are always periodic for the wave functions, while the Coulomb potential can also be calculated for an isolated charge distribution. All spatial derivatives are evaluated using the finite Fourier transform method. The code solves the static Hartree-Fock equations with a damped gradient iteration method and the time-dependent Hartree-Fock equations with an expansion of the time-development operator. Any number of initial nuclei can be placed into the mesh in with arbitrary positions and initial velocities.

Reasons for the new version:
A few bugs were fixed and a number of enhancements added concerning faster convergence, better stability, and more sophisticated analysis of some results.
Summary of revisions:
The following is a brief summary. A more complete documentation can be found as `update.pdf`

in the `Documentation` subdirectory.

New documentation: It was decided to switch the documentation to using the `Doxygen` system (available from `www.doxygen.org`), which can generate the documentation in a variety of formats. To generate the documentation, go into the `Doc-doxygen` subdirectory and execute `make html`, `make latex`, or `make all` to produce the corresponding version or both of them.

The documentation inserted into the source files accounts for most of the formal changes in them. The general documentation is also updated and present as "Documentation.pdf".

Bug fixes:

1. In the force database `forces.data` two digits were interchanged in the definition of SLy4d, leading to wrong results for that force.
2. If a restart is done for a two-body collision, the code changed the number of fragments to `nof`=1. The restart is then initialized like a single-nucleus case with `nof`=1. But two-body analysis was activated only for `nof`¿1 such that it was absent after restart.
3. In the time-dependent mode, the wave functions were only save at intervals of `mprint` and `mrest`, respectively. If a calculation stops because of reaching the final distance or fulfilling the convergence criterion, this may lead to a loss of information, so that now both are done also in this event before the job finishes.
4. The external field parameters were calculated directly from the input in `getin_external`. Since this is called before the fragment initialization is done, coefficients depending on proton or neutron number will not be calculated correctly. For this reason, the calculation of these coefficients is separated into a new routine `init_external`, which is called directly before the dynamic calculation starts.

Array allocation: It turned out that having the working arrays as automatic variables could cause problems, as they are allocated on the stack and the proper stack size must be calculated. Therefore in all cases where a larger array is concerned, it is now changed to `ALLOCATABLE` and allocated and deallocated as necessary.

Elimination of "guru" mode of FFTW3 While the guru mode as defined in the FFTW3 package (see `fftw.org`) offers an elegant formulation of complicated multidimensional transforms, it is not implemented in some support libraries like the Intel® MKL. There is not much loss in speed when this is replaced by standard transforms with some explicit loops added where necessary. This affects the wave function transforms in the y and z direction.

Enhancement of the makefile In the previous version there were several versions of the makefile, which had to be edited by hand to use different compilers. This was reformulated using a more flexible file with various targets predefined. Thus to generate the executable code, it is sufficient to execute "`make` *target* " in the `Code` subdirectory, where *target* is one of the following:

- `seq` : simple sequential version with the gfortran compiler.
- `ifort`, `ifort_seq` : sequential version using the Intel compiler.
- `omp` and `ifort_omp` produce the OpenMP version for the gfortran and Intel compiler, respectively.
- `mpi` : MPI version, which uses the compiler mpif90.
- `mpi-omp` : MPI version also using OpenMP on each node.
- `debug`, `seq_debug`, `omp_debug`, `mpi_debug` : enable debugging mode for these cases. The first three use the gfortran compiler.
- `clean` : removes the generated object and module files.
- `clean-exec` : same as `clean` but removes the executable files as well.

The generated executable files are called `sky3d.seq`, `sky3d.ifort.seq`, `sky3d.mpi`, `sky3d.omp`, `sky3d.ifort.omp`, and `sky3d.mpi-omp`, which should be self-explanatory. Thus several versions may be kept in the code directory, but a `make clean` should be done before producing a new version to make sure the object and module files are correct.

Skyrme-force compatibility for static restart: the code normally checks that the Skyrme forces for all the input wave functions agree. It may be useful, however, to initialize a static calculation from results for a different Skyrme force. Therefore the consistency check was eliminated for the static case.

Acceleration of the static calculations: The basic parameters for the static iterations are (see Eq. 12 of the original paper) x_0 (variable `x0dmp`), which determines the size of the gradient step, and E_0 (variable `e0dmp`) for the energy damping. These were read in and never changed throughout the calculation, except possibly through a restart. This can cause slow convergence, so that a method was developed to change `x0dmp` during the iterations. The value from the input is now regarded as the minimum allowed one and saved in `x0dmpmin`. At the start of the iterations, however, `x0dmp` is multiplied by 3 to attempt a faster convergence.

The change in `x0dmp` is then implemented by comparing the HF energy `ehf` and the fluctuations `efluct1` and `efluct2` to the previous values saved in the variables `ehfprev`, `efluct1prev`, and `efluct2prev`. If the energy decreases or one of the fluctuations decreases by a factor of less than $1 - 10^{-5}$, `x0dmp` is increased by a factor 1.005 to further speed up convergence. If none of these conditions holds, it is assumed that the step was too large and `x0dmp` is reduced by a factor 0.8, but is never allowed to fall below `x0dmpmin`.

This whole process is turned on only if the input variable `tvaryx_0` in the namelist "static" is `.TRUE.` The default value is `.FALSE.`

A speedup up to a factor of 3 has been observed.

External field expectation value This value, which is printed in the file `external.res`, was calculated from the spatial field including the (time-independent) amplitude `amplq0`. The temporal Fourier transform then becomes quadratic in the amplitude, as the fluctuations in the density also grow linearly in `amplq0` (provided the perturbation is not strong enough to take it into the nonlinear regime). This may be confusing and we therefore divided the expectation value by this factor. Note that if the external field is composed of a mixture of different multipoles (not coded presently), an overall scale factor should instead be used.

Enhanced two-body analysis: the analysis of the final two-body quantities after breakup included directly in the code was very simplified and actually it was superfluous to do this so frequently. This is replaced by a much more thorough analyis, including determination of the internal angular momenta of the fragments and of a quite accurate Coulomb energy. It is done only when the final separation is reached, while a simple determination of whether the fragments have separated and, if so, what their distance is, is performed every time step.

Diagonalization In the original program the diagonalization of the Hamiltonian in the subroutine `diagstep` was carried out employing an eigenvalue decomposition using the LAPACK routine `ZHBEVD` which is optimized for banded matrices. This routine is replaced in the update by the routine `ZHEEVD` which is optimized for general hermitian matrices. This change should result in a moderate speed up for very large calculations. Furthermore the array `unitary`, previously a `nstmax`×`nstmax` array has been reduced to a `nlin`×`nlin` array, where `nlin` is the number of wave functions of either neutrons or protons. This array is now used as input and output for `ZHEEVD`.

New formulation of the spin-orbit term: The action of the spin-orbit term has been corrected to comply with a strictly variational form. Starting from the spin-orbit energy

$$E_{ls} = t_{ls} \int d^3r \, \vec{J} \cdot \nabla \rho \quad , \tag{1}$$

we obtain by variation with respect to the s.p. wavefunction ψ^* the spin-orbit term in the mean

field in the symmetrized form

$$\hat{h}_{ls}\psi = \frac{\mathrm{i}}{2}\left(\vec{W}\cdot(\vec{\sigma}\times\nabla)\psi + \vec{\sigma}\cdot(\nabla\times(\vec{W}\psi)\right) \tag{2}$$

where $\vec{W} = t_{ls}\nabla\rho$. In the previous version of the code, this term was simplified by applying the product rule for the ∇ operator yielding

$$\frac{\mathrm{i}}{2}\left(\vec{W}\cdot(\vec{\sigma}\times\nabla)\psi + \vec{\sigma}\cdot(\nabla\times(\vec{W}\psi)\right) \quad = \quad \mathrm{i}\vec{W}\cdot(\vec{\sigma}\times\nabla)\psi \quad . \tag{3}$$

Closer inspection reveals that the product rule is not perfectly fulfilled if the ∇ operator is evaluated with finite Fourier transformation as inevitably done in the grid representation of the code. It turned out that this slight mismatch can accumulate to instabilities in TDHF runs over long times. Thus the variationally correct form (2) has been implemented now, although it leads to slightly longer running times.

Restrictions: The reliability of the mean-field approximation is limited by the absence of hard nucleon-nucleon collisions. This limits the scope of applications to collision energies about a few MeV per nucleon above the Coulomb barrier and to relatively short interaction times. Similarly, some of the missing time-odd terms in the implementation of the Skyrme interaction may restrict the applications to even-even nuclei.

Unusual features:
The possibility of periodic boundary conditions and the highly flexible initialization make the code also suitable for astrophysical nuclear-matter applications.

Running time: The running time depends strongly on the size of the grid, the number of nucleons, and the duration of the collision. For a single-processor PC-type computer it can vary betweeneen a few minutes and weeks.

References

[1] H. Childs, E. Brugger, B. Whitlock, J. Meredith, S. Ahern, D. Pugmire, K. Biagas, M. Miller, C. Harrison, G. H. Weber, H. Krishnan, T. Fogal, A. Sanderson, C. Garth, E. W. Bethel, D. Camp, O. Rübel, M. Durant, J. M. Favre, P. Navrátil, VisIt: An End-User Tool For Visualizing and Analyzing Very Large Data, in: High Performance Visualization–Enabling Extreme-Scale Scientific Insight, 2012, pp. 357–372.