The MADNESS Skyrme Hartree-Fock code MSHF

Authors

Affiliations

Abstract

Manual abstract

1. Introduction

The code MSHF code performs Hartree-Fock simulations of nuclear matter with a focus on ground states of nuclei and so-called nuclear pasta phases. The code is written using the Multi-resolution Adaptive Numerical Environment for Scientific Simulations (MADNESS).

Nuclear pasta phases are exotic shapes of nuclear matter that can form in the crust of neutron stars and during core-collapse supernovae. The presence of nuclear pasta in neutron star crusts can impact the stars' properties and observables such as thermal evolution, magnetic field, oscillations and deformations. In MSHF, the ground state nuclear configuration is determined by finding the minimum binding energy of a system of proton and neutron single particle states in a given simulation volume. The nucleons are hereby subject to self-consistently determined nuclear and Coulomb mean-field potentials. MSHF iteratively solves the Hartree-Fock equations of the nucleonic system using a Skyrme density functional.

In MADNESS, functions and operators are represented via multi-wavelets. MADNESS has built-in solvers for differential equations in many dimensions and for different boundary conditions. For references regarding MADNESS, see [**?** 1, 2, 3, 4]. For applications of MADNESS in density functional theory, see e.g. [5, 6]. For an example of other Skyrme Hartree-Fock simulation codes of nuclei and nuclear pasta phases see e.g. [**?**].

Need an overview of MADNESS here and a bit more description about nuclear pasta in astrophysics. Also an overview of different methods to calculate nuclear pasta. Also a note on the justification of the code since there is an open source version of Sky3D

2. Installing and Running

The MSHF code consists of a main code file `mshf.cc` and two input files, `mshf_input` and `mshf_skyrme`. The first contains parameters for the simulation, e.g. simulation box size and boundary conditions. The second specifies the parameters of the Skyrme nuclear potential. MSHF runs in the MADNESS numerical environment using its functions, operators and parallel runtime. The code files should be saved in the MADNESS example folder and the makefile modified by adding the `mshf` executable:

```
bin_PROGRAMS = hello ...mshf
mshf_SOURCES = mshf.cc
```

Once MADNESS is built, the user can compile MSHF by typing `make mshf`. When running the code, the input files `mshf_input` and `mshf_skyrme` should be in the same folder as the executable. Otherwise, MSHF will run a simulation with default input and nuclear force parameters.

3. The Skyrme Hartree-Fock Equations

MSHF iteratively solves the Hartree-Fock equations for a system consisting of A nucleons:

$$H_q \, \psi_{i,q,s}(\vec{r}) = E_{i,q} \, \psi_{i,q,s}(\vec{r}), \tag{1}$$

with the Hamiltonian:

$$H_q = -\frac{\hbar^2}{2m_q}\,\Delta + U_{\mathrm{HF},q}(\vec{r}) \tag{2}$$

where $\psi_{i,q,s}$ is a single particle state i with mass m_q and energy $E_{i,q}$. Each state has a spin s (up u or down d) and isospin q (proton p or neutron n). The nucleons are subjects to a mean-field potential $U_{\mathrm{HF},q}$ which depends on the states. To solve the above equations in MADNESS, we rewrite them into the Lippmann-Schwinger form:

$$(k_q\,\Delta + E_{i,q})\,\psi_{i,q,s}(\vec{r}) = U_{\mathrm{HF},q}(\vec{r})\,\psi_{i,q,s}(\vec{r}), \tag{3}$$

$$k_q = \frac{\hbar^2}{2m_q} \tag{4}$$

$$\psi_{i,q,s}(\vec{r}) = -\frac{1}{k_q}\,G_{\mathrm{BSH},i,q} \star [U_{\mathrm{HF},q}(\vec{r})\,\psi_{i,q,s}(\vec{r})] \tag{5}$$

$$= -\frac{1}{k_q}\int_{-\infty}^{\infty} G_{\mathrm{BSH},i,q}(\vec{r},\vec{s})\;[U_{\mathrm{HF},q}(\vec{s})\,\psi_{i,q,s}(\vec{s})]\;d\vec{s} \tag{6}$$

using the Green's function for the bound-state Helmholtz equation $G_{\mathrm{BSH},i,q}$:

$$G_{\mathrm{BSH},i,q}(\vec{r},\vec{s}) = \frac{1}{4\pi|\vec{r}-\vec{s}|}\;\exp\left(-\sqrt{-\frac{E_{i,q}}{k_q}}\,|\vec{r}-\vec{s}|\right) \tag{7}$$

The potential $U_{\mathrm{HF},q}$ is given by:

$$\begin{aligned} U_{\mathrm{HF},p}(\vec{r}) = {} & U_{p,\mathrm{sky}}(\vec{r}) + U_{p,\mathrm{meff}}(\vec{r}) + U_{p,so}(\vec{r}) \\ & + U_{p,\mathrm{curr.}}(\vec{r}) + U_{p,\mathrm{spin}}(\vec{r}) \\ & + U_C(\vec{r}) + U_{C,ex}(\vec{r}) \end{aligned} \tag{8}$$

for protons and

$$\begin{aligned} U_{\mathrm{HF},n}(\vec{r}) = {} & U_{n,\mathrm{sky}}(\vec{r}) + U_{n,\mathrm{meff}}(\vec{r}) + U_{n,so}(\vec{r}) \\ & + U_{n,\mathrm{curr.}}(\vec{r}) + U_{n,\mathrm{spin}}(\vec{r}) \end{aligned} \tag{9}$$

for neutrons, respectively. The first component is a local nuclear potential whereas we use the Skyrme density functional:

$$\begin{aligned} U_{q,\mathrm{sky}}(\vec{r}) = {} & b_0\rho(\vec{r}) - b_0'\rho_q(\vec{r}) + b_1\tau(\vec{r}) - b_1'\tau_q(\vec{r}) \\ & - b_2\Delta\rho(\vec{r}) + b_2'\Delta\rho_q(\vec{r}) + b_3\frac{\alpha+2}{3}\rho^{\alpha+1}(\vec{r}) \\ & - b_3'\frac{2}{3}\rho^{\alpha}(\vec{r})\rho_q(\vec{r}) - b_3'\frac{\alpha}{3}\rho^{\alpha-1}(\vec{r})\left(\rho_n^2(\vec{r}) + \rho_p^2(\vec{r})\right) \\ & - b_4\nabla\cdot\vec{J}(\vec{r}) - b_4'\nabla\cdot\vec{J}_q(\vec{r}), \end{aligned} \tag{10}$$

which is a sum of contributions from neutron and proton number densities ρ_q, kinetic densities τ_q, and spin-orbit densities $\vec{J}_q$. The constants b_j and b_j' ($j = 0\ldots4$) are parameters of the Skyrme potential and fitted to reproduce specific nuclear matter properties. The number densities ρ_q and kinetic densities τ_q are calculated form the single particle states via:

$$\rho_q(\vec{r}) = \sum\nolimits_i^{N_q} |\psi_{i,q,u}(\vec{r})|^2 + |\psi_{i,q,d}(\vec{r})|^2\,, \tag{11}$$

$$\rho(\vec{r}) = \rho_p(\vec{r}) + \rho_n(\vec{r}), \tag{12}$$

$$\tau_q(\vec{r}) = \sum\nolimits_i^{N_q} |\nabla\psi_{q,i,u}(\vec{r})|^2 + |\nabla\psi_{q,i,d}(\vec{r})|^2\,, \tag{13}$$

$$\tau(\vec{r}) = \tau_p(\vec{r}) + \tau_n(\vec{r}), \tag{14}$$

with $N_q = Z$ for protons and $N_q = A - Z$ for neutrons where where A is the mass number and Z the charge number of the nuclear configuration. The divergence of the spin-orbit density $\vec{J}(\vec{r})$ is determined by:

$$\begin{aligned}\nabla \cdot \vec{J}_q(\vec{r}) = -\mathrm{i} \sum\nolimits_i^{N_q} \sum\nolimits_{ss'} \nabla \psi^\star_{i,q,s'}(\vec{r}) \\ \times \nabla \psi_{i,q,s}(\vec{r}) \cdot \langle s' \,|\, \vec{\sigma} \,|\, s \rangle \,,\end{aligned} \tag{15}$$

where $\vec{\sigma} = (\sigma_x, \sigma_y, \sigma_z)^T$ is a vector of Pauli matrices.
For proton single-particle states, the Coulomb exchange potential $U_{C,\mathrm{ex}}$ in eq.(8) is calculated via the so-called Slater approximation [7]:

$$U_{\mathrm{C,ex}}(\vec{r}) = -\mathrm{e}^2 \left(\frac{3\,\rho_p(\vec{r})}{\pi} \right)^{1/3} . \tag{16}$$

The proton Coulomb potential is given by:

$$U_{Cp}(\vec{r}) = \mathrm{e}^2 \int \frac{\rho_p(\vec{s})}{|\vec{r} - \vec{s}|}\, d\vec{s}. \tag{17}$$

For periodic boundary conditions, we impose electric charge neutrality in the simulation volume by adding electrons. They are included via the so-called Jellium approximation, i.e. a homogeneous charge density $\rho_J = -Z/A$. As a consequence, in addition to the proton Coulomb potential we have to include the Coulomb potential of the Jellium:

$$U_{CJ}(\vec{r}) = \mathrm{e}^2 \int \frac{\rho_J}{|\vec{r} - \vec{s}|}\, d\vec{s}. \tag{18}$$

The sum of both contributions results in the total Coulomb potential:

$$U_C(\vec{r}) = U_{Cp}(\vec{r}) + U_{CJ}(\vec{r}) = \mathrm{e}^2 \int \frac{\rho_C(\vec{s})}{|\vec{r} - \vec{s}|}\, d\vec{s}, \tag{19}$$

where $\rho_C(\vec{r}) = \rho_p(\vec{r}) + \rho_J$. Electron screening can be implemented via the following Green's function:

$$G_{S,C}(\vec{r}) = \frac{1}{4\pi r} \exp\left(-\frac{r}{\lambda}\right), \tag{20}$$

with a screening length λ. The resulting screened proton Coulomb potential is then:

$$U_{S,Cp}(\vec{r}) = \frac{\mathrm{e}^2}{4\pi} \int \frac{\rho_p(\vec{s})}{|\vec{r} - \vec{s}|}\, \exp\left(-\frac{|\vec{r} - \vec{s}|}{\lambda}\right) d\vec{s}. \tag{21}$$

The exchange potential is modified according to [8, 9]:

$$U_{S,ex}(\vec{r}) = U_{ex}(\vec{r})\, F(\alpha), \tag{22}$$

$$F(\alpha) = 1 - \frac{4}{3}\alpha \tan^{-1}\left(\frac{2}{\alpha}\right) + \frac{1}{2}\alpha^2 \ln\left(1 + 4\alpha^{-2}\right) \tag{23}$$

$$+ \frac{1}{6}\alpha^2 \left[1 - \frac{1}{4}\alpha^2 \ln(1 + 4\alpha^{-2})\right], \tag{24}$$

$$\alpha(\vec{r}) = \left(\lambda\, k_F(\vec{r})\right)^{-1}, \;\; k_F(\vec{r}) = \left(3\pi^2 \rho_p(\vec{r})\right)^{1/3}. \tag{25}$$

The remaining components of the nucleon potential in eq.(8) and eq.(9) account for the density-dependent effective nucleon mass:

$$U_{q,\mathrm{meff}}(\vec{r}) = -\nabla \cdot \left(b_1 \rho(\vec{r}) - b'_1 \rho_q(\vec{r})\right) \nabla \tag{26}$$

and the spin-orbit potential:

$$U_{q,so}(\vec{r}) = \mathrm{i}\, \nabla W_q \cdot (\vec{\sigma} \times \nabla), \;\; W_q = b_4 \rho(\vec{r}) + b'_4 \rho_q(\vec{r}). \tag{27}$$

For time-independent HF calculations of even-A and even-even nuclei and pasta phases we do not include the current and spin potentials $U_{q,\mathrm{curr.}}$ and $U_{q,\mathrm{spin}}$. The final potentials that are applied in the HF equations for protons and neutrons are:

$$\begin{aligned} U_{\mathrm{HF},p}(\vec{r}) &= U_{p,\mathrm{sky}}(\vec{r}) - \nabla \cdot \left(b_1 \rho(\vec{r}) - b_1' \rho_p(\vec{r})\right) \nabla \\ &\quad + \mathrm{i}\, \vec{W}_p(\vec{r}) \cdot (\vec{\sigma} \times \nabla) + U_C(\vec{r}) + U_{C,ex}(\vec{r}), \qquad (28) \\ U_{\mathrm{HF},n}(\vec{r}) &= U_{n,\mathrm{sky}}(\vec{r}) - \nabla \cdot \left(b_1 \rho(\vec{r}) - b_1' \rho_n(\vec{r})\right) \nabla \\ &\quad + \mathrm{i}\, \vec{W}_n(\vec{r}) \cdot (\vec{\sigma} \times \nabla). \qquad (29) \end{aligned}$$

The total binding energy of the system is calculated from the energy components of the Skyrme density functional:

$$E_0 = \frac{1}{2} \int b_0\, \rho^2(\vec{r}) - b_0' \left[\rho_p^2(\vec{r}) + \rho_n^2(\vec{r})\right]\, d\vec{r}, \qquad (30)$$

$$E_1 = \int b_1 \rho(\vec{r}) \tau(\vec{r}) - b_1' \left[\rho_p(\vec{r}) \tau_p(\vec{r}) + \rho_n(\vec{r}) \tau_n\right]\, d\vec{r}, \qquad (31)$$

$$\begin{aligned} E_2 &= -\frac{1}{2} \int b_2\, \rho(\vec{r}) \Delta \rho(\vec{r}) \\ &\quad - b_2' \left[\rho_p(\vec{r}) \Delta \rho_p + \rho_n(\vec{r}) \Delta \rho_n(\vec{r})\right]\, d\vec{r}, \qquad (32) \end{aligned}$$

$$E_3 = \frac{1}{3} \int b_3\, \rho^{\alpha+2}(\vec{r}) - b_3'\, \rho^{\alpha}(\vec{r}) \left[\rho_p^2(\vec{r}) + \rho_n^2(\vec{r})\right]\, d\vec{r}, \qquad (33)$$

$$\begin{aligned} E_4 &= -\int b_4\, \rho(\vec{r})\, \nabla \vec{J}(\vec{r}) \\ &\quad + b_4' \left[\rho_q(\vec{r})\, \nabla \vec{J}_p(\vec{r}) + \rho_n(\vec{r})\, \nabla \vec{J}_n(\vec{r})\right]\, d\vec{r}, \qquad (34) \end{aligned}$$

as well as the kinetic, Coulomb and Coulomb exchange energies:

$$E_{\mathrm{kin}} = \sum\nolimits_q k_q \int \tau_q(\vec{r})\, d\vec{r}, \qquad (35)$$

$$E_C = \frac{1}{2} \int U_C(\vec{r})\, \rho_p(\vec{r})\, d\vec{r}, \qquad (36)$$

$$E_{C,ex} = -\frac{3}{4} \mathrm{e}^2 \left(\frac{3}{\pi}\right)^{\frac{1}{3}} \int \rho_p(\vec{r})\, d\vec{r}. \qquad (37)$$

With that, the total energy and binding energy per nucleon are then given by:

$$\begin{aligned} E_{\mathrm{total}} &= E_{\mathrm{kin}} + E_C + E_{C,ex} + E_0 + E_1 + E_2 \\ &\quad + E_3 + E_4, \quad E_{\mathrm{bind}} = E_{\mathrm{total}}/\mathrm{A}, \qquad (38) \end{aligned}$$

4. Iterative solution of the Skyrme Hartree-Fock equations

We start out with a set of single particle states $\psi_{i,q,s}^n$ at iteration step $n = 0$. The states are usually initialized either by harmonic oscillator states, plane waves, or Gaussians. After setting up the Hamiltonian H_q, the single particle states are orthogonalized and normalized by solving the generalized eigenvalue

problem:

$$\begin{aligned}
\tilde{H}_q\, C_q &= S_q\, C_q\, E_q, \\
\tilde{H}_{q,ij} &= \int \psi^n_{i,q,u}(\vec{r})^\star\, H_q\, \psi^n_{j,q,u}(\vec{r}) \\
&\quad + \psi^n_{i,q,d}(\vec{r})^\star\, H_q\, \psi^n_{j,q,d}(\vec{r})\, d\vec{r}, \qquad (39) \\
\tilde{S}_{q,ij} &= \int \psi^n_{i,q,u}(\vec{r})^\star\, \psi^n_{j,q,u}(\vec{r}) \\
&\quad + \psi^n_{i,q,d}(\vec{r})^\star\, \psi^n_{j,q,d}(\vec{r})\, d\vec{r} \qquad (40)
\end{aligned}$$

for C_q and E_q. In MADNESS this is done by using the LAPACK hermitian eigensolver. The new orthonormal single particle states $\tilde{\psi}^n_{i,q,s}$ with energies $E_{i,q}$ are obtained via

$$\tilde{\psi}^n_{i,q,s}(\vec{r}) = \sum\nolimits_j^{N_q} \psi^n_{j,q,s}(\vec{r})\, C_{q,ij}. \qquad (41)$$

In the same iteration step, the states are updated by applying $U_{\mathrm{HF,q}}$ and $G_{\mathrm{BSH,i,q}}$:

$$\phi^{n+1}_{i,q,s}(\vec{r}) = -k_q^{-1}\, G_{\mathrm{BSH,i,q}} \star \left(U_{\mathrm{HF,q}}(\vec{r})\, \tilde{\psi}^n_{i,q,s}(\vec{r}) \right). \qquad (42)$$

We then determine the maximum change among all single-particle states:

$$\delta\psi = \max\{\delta\psi_{0,p}, ..., \delta\psi_{N_p,p}, \delta\psi_{0,n}, ..., \delta\psi_{N_n,n}\}, \qquad (43)$$

$$\delta\psi_{i,q}(\vec{r}) = \sqrt{\int |\delta\psi_{i,q,u}|^2 + |\delta\psi_{i,q,d}|^2} \qquad (44)$$

$$\delta\psi_{i,q,s}(\vec{r}) = \phi^{n+1}_{i,q,s}(\vec{r}) - \tilde{\psi}^n_{i,q,s}(\vec{r}) \qquad (45)$$

If $\delta\psi$ is smaller than or equal to a desired precision ϵ, the calculation is considered converged. Otherwise, the new single-particle states are calculated by mixing the old and new states:

$$\psi^{n+1}_{i,q,s}(\vec{r}) = \chi\, \phi^{n+1}_{i,q,s}(\vec{r}) + (1-\chi)\, \tilde{\psi}^n_{i,q,s}(\vec{r}), \qquad (46)$$

with $0 < \chi < 1$, typically $\chi = 0.4$. The new states are used to calculate an updated Hamiltonian and Green's functions and the above steps are repeated until $\delta\psi \leq \epsilon$.
To reduce numerical noise we apply Gaussian smoothing on $\Delta\rho_q$ by convolutions with:

$$G_{\mathrm{smooth}}(\vec{r}) = \left(\frac{1}{\sqrt{2\pi}\, b_r} \right)^3 \exp\left(-\frac{r^2}{2\, b_r^2} \right), \qquad (47)$$

where typically $b_r \sim 0.25$ fm is chosen. Gaussian smoothing is used for $\Delta\rho_q$ and τ_q during the initial iterations. Once the calculations starts to converge smoothing is switched off for τ_q. For $\Delta\rho_q$ it is replaced by mixing, similar to the mixing of single-particle states in eq.(46).

5. Code Files and Routines

5.1. Input files

The `mshf_input` file contains parameters to initialize and run the simulation. They are categorized into general and nuclear parameters, mixing parameters, output and additional parameters and are listed in tables 1 - 5.

Table 1: General parameters in file `mshf_input`

Variable	Description	Values
`A`	Mass number	
`Z`	Charge number	
`Box`	Box length in fm	
`initial`	Single particle initialization	1 = Gaussians 2 = Plane Waves 3 = Harm. Osc.
`boundary`	Boundary conditions	0 = free space, 1 = periodic
`knumber`	Wavelet number	
`thresh`	Truncation threshold	
`IO_nodes`	# nodes for IO	
`project`	Projection switch	1 = yes, else no

Table 2: Nuclear parameters in file `mshf_input`

Variable	Description	Values
`jellium`	Jellium switch	1 = yes, else no
`spinorbit`	Spin-orbit switch	1 = yes, else no
`meff`	Eff. mass switch	1 = yes, else no
`screening`	Electron screening	1 = yes, else no
`screenl`	Screen. length [fm]	
`lap_comp`	Method for $\Delta\rho$	0 - 2

General parameters:. These consist of the mass and charge number of the nuclear configuration `A` and `Z`, respectively. The code can handle up to several thousand nucleons. The problem size is usually dependent on the available memory and number of nodes. The `Box` parameter gives the length of one box side (we assume a cubic simulations space) in fm. The boundary conditions are specified via `periodic`. While MADNESS provides a range of different boundary conditions, our code distinguishes between periodic and free space. The first are used for nuclear pasta calculations while the second are recommended for ground state calculations of isolated nuclei in a large simulation volume. The initialization of the single particle states is specified by the `initial` parameter with values $1-3$. The wavelet number and truncation threshold are given by `knumber` and `thresh`, respectively. However, note that is the automatic projection switch `project` is set to one, these are adjusted as the simulations converges by increasing `knumber` and decreasing `thresh` for better accuracy. For that, we follow the recommendation of MADNESS via ... . Finally, MADNESS allows parallel IO with a given number of nodes which we specify via the parameter `IO_nodes`.

Nuclear parameters:. The nuclear parameter sets contains information on whether the spin-orbit potential and the effective mass potential should be included. For ground state calculations of nuclei they should be included. For pasta simulations it sometimes makes sense to run tests without including both potentials since they significantly increase the simulation time. If `spinorbit` and `meff` are set to one, the spin-orbit potential and effective mass potential, respectively, are included. This parameter set also specifies how the Coulomb potential is calculated. If the simulation is done with periodic boundary conditions, then the jellium approximation must be applied. It includes a homogeneous background of electrons for overall charge neutrality. The jellium is included via the `jellium` switch. Furthermore, we can include electron screening via the `screening` switch. If the latter is switched on, the screening length must be given in femtometer via `screenl`. Finally, we specify how the laplacian of the neutron and proton densities is calculated. We have three different methods which are described in section ... with their advantages and drawbacks. The

Table 3: Mixing parameters in file `mshf_input`

Variable	Description	Values
`avg_pot`	U_{sky} mixing switch	1 = yes, else no
`avg_lap`	$\Delta\rho_q$ mixing switch	1 = yes, else no
`avg_wav`	ψ_i mixing switch	1 = yes, else no
`chi`	Mixing parameter χ	≥ 0.5 for stability

Table 4: Output parameters in file `mshf_input`

Variable	Description	Values
`vtk_output`	vtk output switch	1 = yes, else no
`txt_output`	txt output switch	1 = yes, else no
`timing`	Timing information	1 = yes, else no

method is specified by the parameter `lap_comp`.

Mixing parameters: . These parameters set which quantities should be mixed between the old and the new iteration. We typically mix the single particle states throughout the entire simulation and the laplacian and kinetic densities while the simulation is converging and the nuclear configuration changing. However, we also provide the option to mix the Skyrme nuclear potential. The mixing parameter `chi` gives the amount by which the old and new variables are mixed.

Output parameters:. There are two options for output. One gives .txt files for profiles along the x, y, and z axis for the total number density, the total kinetic density, the laplacian of the density and the potential. The created files are named accordingly. Alternatively or in addition to that, MSHF also gives output files in the vtk format that can be visualized by e.g. Paraview. For the latter there is one single file printed every defined number of iteration steps that contains the neutron, proton and total number densities, the kinetic density, the laplacian of the density and the Skyrme potential. To switch on the .txt or .vtk output the corresponding flags, `vtk_output` and `txt_output`, should be set to one. We also provide the option to print out timing information for different steps in the code. For that the `timing` switch should be set to one.

Additional parameters. Note that if the `project` parameter is set to 1, the wavelet number `knumber` and truncation threshold `thresh` are automatically adjusted for a higher accuracy as the simulation converges. The parameters are checkpointed and used at the restart of the simulation.

Table 5: Additional parameters

Variable	Description	Values
`prec`	Add. precision factor for truncation	`thresh*prec`
`tol`	for `BSH` and `Coulomb`	
`brad`	Smoothing radius	~ 0.25 fm; if < 0 smoothing is replaced by mixing

Skyrme parameters:. The `mshf_skyrme` file contains the parameters $t_0 - t_4$, $x_0 - x_4$, α, k_n and k_p which are used to calculate the b-parameters of the Skyrme potential via:

$$b_0 = t_0\,(1 + 0.5\,x_0)\,,\;\; b'_0 = t_0\,(0.5 + x_0) \tag{48}$$
$$b_1 = 0.25\,[t_1\,(1 + 0.5\,x_1) + t_2\,(1 + 0.5\,x_2)]\,, \tag{49}$$
$$b'_1 = 0.25\,[t_1\,(0.5 + x_1) - t_2\,(0.5 + x_2)] \tag{50}$$
$$b_2 = 0.125\,[3\,t_1\,(1 + 0.5\,x_1) - t_2\,(1 + 0.5\,x_2)]\,, \tag{51}$$
$$b'_2 = 0.125\,[3\,t_1\,(0.5 + x_1) + t_2\,(0.5 + x_2)] \tag{52}$$
$$b_3 = 0.25\,t_3\,(1 + 0.5\,x_3)\,,\;\; b'_3 = 0.25\,t_3\,(0.5 + x_3) \tag{53}$$
$$b_4 = 0.5\,t_4,\;\; b'_4 = 0.5\,t_4. \tag{54}$$

The coefficient `alpha` is α in eq.(10), and `k_fn` and `k_fp` are

$$k_{fn} = \hbar^2/(2\,m_n),\;\; k_{fp} = \hbar^2/(2\,m_p). \tag{55}$$

5.2. Code structure

The simulations are started out with the setup of the single particle states. These are either initialized from scratch or read-in from a checkpoint file. We then calculate the density terms, the local nuclear potential, the Coulomb and Coulomb exchange potentials. Together with the effective mass potential and the spin-orbit potential, the Hamiltonian is built. It is used to ortho-normalize the single particle states and update them via convolution with $G_{\mathrm{BSH},i,q}$. The old and the new states are compared to determine the maximum change in the wavefunctions `delta_psi`. If the change is smaller than a defined mass number dependent threshold value but larger than the required precision for convergence we increase the wavelet number `knumber` and decrease the truncation threshold `thresh`. The the above iteration steps are then repeated. In computationally intensive parts of the iteration step, we perform MPI load balancing and for specified iterations write output files for plotting and checkpointing.
The main routine in the code is the `ground_state` routine. It performs the following loop (the corresponding code routines are indicated by an arrow):

1. Initialization of single particle states. If checkpointing files exist, read states from the file and go to step 2, otherwise initialize states as Gaussians around point coordinates from an input file (`make_MD`, `MD`), Harmonic Oscillator states (`make_HO`, `HO`, `HOm`) or Plane waves (`make_Fermi`, `Fermi`), all with spin up and spin down
2. Print out iteration information (e.g. iteration number, wavelet number aso.) and check whether convergence criteria is fulfilled. If not proceed.
3. Build local potential (`Potential`)
   - Calculate densities (via `ndensity`, `kdensity`, `so_density`) and laplacian of the density (via `laplacian`, `laplacian1` or `laplacian2`)
   - Determine kinetic energies of single particle states (`Kmatrix`)
   - Perform Gaussian smoothing for $\Delta\rho_q$ and τ_q terms, or mixing of $\Delta\rho_q$
   - Calculate density terms with fractional power (`rho_power`)
   - Determine Coulomb potential (`CoulombOperator`, `BSHOperator3D`) and the Coulomb exchange potential (`U_ex`)
   - Assemble the local potential
   - Calculate energies (`Energies`) and total binding energy (`Ebinding`)
   - Write output file for plotting (`output`)
4. Ortho-normalize and update single particle states (`iterate`):
   - Apply spin-orbit potential (`Uso`) and effective mass potential (`Umeff`)

- Ortho-normalize states (`orthonorm`)
- Build Green's functions $G_{\mathrm{BSH},i,q}$ (`BSHOperators`) and update ortho-normal states by convolution with $G_{\mathrm{BSH},i,q}$
- Calculate `delta_psi` between old and new states
- Mix single particle states

5. If necessary, project to higher wavelet number and decrease truncation threshold
6. Create checkpoint files and go back to step 2

5.3. State initialization

There are currently three different ways to initialize the nucleon single-particle states:

1. `make_MD`, `MD`
   This routine sets up Gaussians

$$\psi_{i,q,s}(\vec{r}) = \exp\left(-\frac{1}{2d^2}\left(\Delta n_x^2 + \Delta n_y^2 + \Delta n_z^2\right)\right), \tag{56}$$

$$\Delta n_x = x - N_x,\ \ \Delta n_y = y - N_y,$$
$$\Delta n_z = z - N_z, \tag{57}$$

   of widths $d = 3$ fm around given nucleon coordinates $(N_x, N_y, N_z)^T$ that should be provided in a separate input file as well as their 27 mirror images. This initialization can only be used with periodic boundary conditions.
2. `make_HO`, `HO`, `HOm`
   The routine `HOm` sets up modified Harmonic Oscillator states

$$\psi_{i,q,s}(\vec{r}) = x^{N_x}\, y^{N_y}\, z^{N_z} \times \exp\left[-\frac{1}{2}\left(\frac{x^2}{d_x^2} + \frac{y^2}{d_y^2} + \frac{z^2}{d_z^2}\right)\right], \tag{58}$$

   given a sequence of integers N_x, N_y and N_z and constant d_x, d_y, and d_z. In the routine `HO`, we set $d_x = d_y = d_z = d$. These initializations can be used with periodic and free-space boundary conditions.
3. `make_Fermi`, `Fermi`
   We initialize single-particle states as Plane Waves

$$\psi_{i,q,s}(\vec{r}) = \exp\left(\mathrm{i}\,\vec{k}\cdot\vec{r}\right), \tag{59}$$

$$k_j = \pm\frac{2\pi}{\lambda_j},\quad \lambda_i = \frac{2L}{N_j},\quad j = x, y, z \tag{60}$$

   with a sequence of integers N_x, N_y, and N_z. This initialization can be used with periodic and free-space boundary conditions, but we recommend to use periodic boundary conditions only.

The neutron and proton states are initialized as vectors of functions and arranged into spin up and down states in the following way (the corresponding notation in the code for each vectors is given at the beginning of each line):

$$\texttt{psi\_nu}: \quad \vec{\psi}_{n,u}(\vec{r}) = \left(\psi_{1,n,u}(\vec{r}),\ \ldots,\ \psi_{A-Z,n,u}(\vec{r})\right)^T \tag{61}$$

$$\texttt{psi\_nd}: \quad \vec{\psi}_{n,d}(\vec{r}) = \left(\psi_{1,n,d}(\vec{r}),\ \ldots,\ \psi_{A-Z,n,d}(\vec{r})\right)^T \tag{62}$$

$$\texttt{psi\_pu}: \quad \vec{\psi}_{p,u}(\vec{r}) = \left(\psi_{1,p,u}(\vec{r}),\ \ldots,\ \psi_{Z,p,u}(\vec{r})\right)^T \tag{63}$$

$$\texttt{psi\_pd}: \quad \vec{\psi}_{p,d}(\vec{r}) = \left(\psi_{1,p,d}(\vec{r}),\ \ldots,\ \psi_{Z,p,d}(\vec{r})\right)^T \tag{64}$$

For states that can be either protons or neutrons we usually use the notations `psi_qu` and `psi_ud`.

5.4. Simple manipulation of single particle states

These routines manipulate existing single-particle states, e.g. normalization or truncation:

- `normalize_2v`: Normalizes any two vectors of functions $\vec{\psi}_n$ (`psi_n`) and $\vec{\psi}_p$ (`psi_p`) by scaling each state with its L_2 norm:

$$\vec{\psi}_q/|\vec{\psi}_q|, \quad |\vec{\psi}_q| = \sqrt{\int |\vec{\psi}_q|^2 \, d\vec{r}}, \quad q = n \text{ or } p \tag{65}$$

- `normalize_ud`: Normalizes spin-up (`psi_qu`) and down (`psi_qd`) states via:

$$\psi_{i,q,s}/\sqrt{|\psi_{i,q,u}|^2 + |\psi_{i,q,d}|^2}, \quad s = u \text{ or } d \tag{66}$$

- `spin_split`: Splits a vector of single particle states $\vec{\psi}_q$ (`psi_q`) into two vectors of functions of spin-up $\vec{\psi}_{q,u}$ (`psi_qu`) and down $\vec{\psi}_{q,d}$ (`psi_qd`) states

- `trunate1`, `truncate2`: Truncates one and two vectors of single particle states at the level `thresh*prec`. With free boundary conditions, we use truncation mode 1 while for periodic boundary conditions we use mode 0.

5.5. Densities

These routines calculate various particle densities for the the Skyrme, Coulomb and Coulomb exchange potentials. Most of these routines are called in the `Potential` routine (see section 5.6).

- `ndensity`: Calculates the number density of protons or neutrons using the single particle states $\psi_{q,u}$ (`psi_qu`) and $\psi_{q,d}$ (`psi_qd`):

$$\rho_q = \sum\nolimits_i \left(|\psi_{i,q,u}|^2 + |\psi_{i,q,d}|^2\right) \tag{67}$$

- `kdensity`: **intput**: Determines kinetic densities τ_q from partial derivatives of single particle states

$$\tau_q = \sum\nolimits_i \left(|\nabla\psi_{i,q,u}|^2 + |\nabla\psi_{i,q,d}|^2\right) \tag{68}$$

- `laplacian`: Used when `lap_comp = 0`. Calculates the laplacian of the number density in the usual way via:

$$\Delta\rho_q = \frac{\partial^2}{\partial x^2}\rho_q + \frac{\partial^2}{\partial y^2}\rho_q + \frac{\partial^2}{\partial z^2}\rho_q. \tag{69}$$

- `laplacian1`: Used when `lap_comp = 1`. Calculates the laplacian of the number density by the sum:

$$\begin{aligned}\Delta\rho_q &= \Delta \sum\nolimits_i \left(\psi^\star_{i,q,u}\,\psi_{i,q,u} + \psi^\star_{i,q,d}\,\psi_{i,q,d}\right) \\ &= \nabla\cdot\sum\nolimits_i \left(\nabla\psi^\star_{i,q,u}\right)\psi_{i,q,u} + \left(\nabla\psi^\star_{i,q,d}\right)\psi_{i,q,d} \\ &+ \psi^\star_{i,q,u}\nabla\psi_{i,q,u} + \psi^\star_{i,q,d}\nabla\psi_{i,q,d}\end{aligned} \tag{70}$$

  This takes longer than `laplacian`. However, the resulting function for $\Delta\rho_q$ seems to be less prone to numerical noise.

- `laplacian2`: Used when `lap_comp = 2`. Calculates the laplacian of the number density by the sum:

$$\begin{aligned}\Delta\rho_q &= \Delta\sum\nolimits_i \left(\psi^\star_{i,q,u}\,\psi_{i,q,u} + \psi^\star_{i,q,d}\,\psi_{i,q,d}\right) &(71)\\ &= \sum\nolimits_i \psi_{i,q,u}\Delta\psi^\star_{i,q,u} + 2\nabla\psi^\star_{i,q,u}\cdot\nabla\psi_{i,q,u}\\ &+ \psi^\star_{i,q,u}\Delta\psi_{i,q,u} + \psi_{i,q,d}\Delta\psi^\star_{i,q,d}\\ &+ 2\nabla\psi^\star_{i,q,d}\cdot\nabla\psi_{i,q,d} + \psi^\star_{i,q,d}\Delta\psi_{i,q,d} &(72)\end{aligned}$$

  This takes significantly longer than e.g. `laplacian`. However, the resulting $\Delta\rho_q$ seems to be less prone to numerical noise

- `rho_power`: Calculates the functions ρ^α for a given α

- `so_density`: Calculates $\nabla\vec{J}_q$ according to:

$$\nabla\vec{J}_q = -\mathrm{i}\sum\nolimits_i\sum\nolimits_{ss'} \nabla\psi^\star_{i(q,s')}\times\nabla\psi_{i(q,s)}\cdot\langle s'\,|\,\vec{\sigma}\,|\,s\rangle\,. \tag{73}$$

  The outer product for one state is:

$$\begin{aligned}&\nabla\psi^\star_{i,q,s'}\times\nabla\psi_{i,q,s}\\ &= \begin{pmatrix}\partial_y\psi^\star_{i,q,s'}\,\partial_z\psi_{i,q,s} - \partial_y\psi_{i,q,s}\,\partial_z\psi^\star_{i,q,s'}\\ \partial_z\psi^\star_{i,q,s'}\,\partial_x\psi_{i,q,s} - \partial_z\psi_{i,q,s}\,\partial_x\psi^\star_{i,q,s'}\\ \partial_x\psi^\star_{i,q,s'}\,\partial_y\psi_{i,q,s} - \partial_z\psi_{i,q,s}\,\partial_y\psi^\star_{i,q,s'}\end{pmatrix}.\end{aligned} \tag{74}$$

  The spin eigenstates can either be spin-up $u = (1,0)^T$ or spin-down $d = (0,1)^T$. With the Pauli matrices

$$\sigma_x = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix},\ \sigma_y = \begin{pmatrix}0 & -\mathrm{i}\\ \mathrm{i} & 0\end{pmatrix},$$
$$\sigma_z = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}, \vec{\sigma} = (\sigma_x,\sigma_y,\sigma_z)^T$$

  the different contributions to eq.(73) can be calculated as:

$$\begin{aligned}\nabla\vec{J}_{i,q,uu} &= \nabla\psi^\star_{i,q,u}\times\nabla\psi_{i,q,d}\cdot\langle u\,|\,\vec{\sigma}\,|\,u\rangle &(75)\\ &= \partial_x\psi^\star_{i,q,u}\,\partial_y\psi_{i,q,u} - \partial_x\psi_{i,q,u}\,\partial_y\psi^\star_{i,q,u}\\ \nabla\vec{J}_{i,q,dd} &= \nabla\psi^\star_{i,q,d}\times\nabla\psi_{i,q,d}\cdot\langle d\,|\,\vec{\sigma}\,|\,d\rangle &(76)\\ &= -\partial_x\psi^\star_{i,q,d}\,\partial_y\psi_{i,q,d} + \partial_x\psi_{i,q,d}\partial_y\psi^\star_{i,q,d}\\ \nabla\vec{J}_{i,q,ud} &= \nabla\psi^\star_{i,q,u}\times\nabla\psi_{i,q,d}\cdot\langle u\,|\,\vec{\sigma}\,|\,d\rangle &(77)\\ &= \partial_y\psi^\star_{i,q,u}\,\partial_z\psi_{i,q,d} - \partial_y\psi_{i,q,d}\,\partial_z\psi^\star_{i,q,u}\\ &- \mathrm{i}\partial_z\psi^\star_{i,q,u}\partial_x\psi_{i,q,d} + \mathrm{i}\partial_z\psi_{i,q,d}\partial_x\psi^\star_{i,q,u}\\ \nabla\vec{J}_{i,q,du} &= \nabla\psi^\star_{i,q,d}\times\nabla\psi_{i,q,u}\cdot\langle d\,|\,\vec{\sigma}\,|\,u\rangle &(78)\\ &= \partial_y\psi^\star_{i,q,d}\,\partial_z\psi_{i,q,u} - \partial_y\psi_{i,q,u}\,\partial_z\psi^\star_{i,q,d}\\ &+ \mathrm{i}\partial_z\psi^\star_{i,q,d}\partial_x\psi_{i,q,u} - \mathrm{i}\partial_z\psi_{i,q,u}\partial_x\psi^\star_{i,q,d}\end{aligned}$$

  The final expression for the divergence of the spin-density is the sum:

$$\begin{aligned}\nabla\vec{J}_q(\vec{r}) = \sum\nolimits_i &\nabla\vec{J}_{i,q,uu}(\vec{r}) + \nabla\vec{J}_{i,q,du}(\vec{r})\\ &+ \nabla\vec{J}_{i,q,ud}(\vec{r}) + \nabla\vec{J}_{i,q,dd}(\vec{r}).\end{aligned} \tag{79}$$

  It consists of the 12 terms in equations (76) - (79). The subroutine `so_densities` calculates and sums these terms for the proton and neutron states.

5.6. Potentials

With the various calculated density terms we setup the Skyrme nuclear mean-field potential in the routine `Potential` together with the Coulomb and Coulomb exchange potentials. The `Potential` routine also performs Gaussian smoothing of $\Delta\rho_q$ and τ_q or mixing of $\Delta\rho_q$ between the current and previous iteration. Summing up all density terms, $U_{q,\mathrm{sky}}$ is assembled. The binding energies and all energy terms are calculated via the `Ebinding` and `Energies` for output information followed by plotting data in the `output` routine. Some potential routines that are applied in `Potential` are:

- `uex`: Calculates the Coulomb exchange potential `U_ex` from the proton density. We first determine

$$U = -\left(3/\pi\right)^{1/3} \rho_p^{-2/3} \tag{80}$$

  and then multiply by ρ_p and the elementary charge $\mathrm{e}^2 = 1.43989$ MeV fm:

$$U_{C,ex} = -\mathrm{e}^2 \left(3/\pi\right)^{1/3} \rho_p^{1/3}. \tag{81}$$

- `Fgamma`: Returns the modification function $F(\alpha)$ for the screened exchange potential in eq.(22) and eq.(25).

The effective mass and spin-orbit potential are calculated in the `iterate` routine. In fact, instead of just calculating the potentials, we determine the products $U_{\mathrm{meff},q,s}\,\vec{\psi}_{q,s}$ and $U_{so}\,\vec{\psi}_{q,u}$ that are represented by vectors of functions in the the code.

- `Umeff`: Applies the effective mass potential on single particle state:

$$\begin{aligned} U_{\mathrm{meff},q,s}\,\vec{\psi}_{q,s} &= -\nabla\cdot\left(\, B_q\,\nabla\vec{\psi}_{q,s}\,\right) && (82)\\ &= -\partial_x\left(B_q\,\partial_x\vec{\psi}_{q,s}\right) - \partial_y\left(B_q\,\partial_y\vec{\psi}_{q,s}\right) \\ &\quad - \partial_z\left(B_q\,\partial_z\vec{\psi}_{q,s}\right) && (83)\\ B_q &= b_1\rho - b_1'\rho_q && (84) \end{aligned}$$

- `Uso`: Calculates and applies the spin-orbit potential on the proton and neutron single-particle states:

$$\begin{aligned} U_{q,so}\begin{pmatrix}\vec{\psi}_{q,u}\\ \vec{\psi}_{q,d}\end{pmatrix} &= \mathrm{i}\,\left(\nabla\,W_q\right)\left(\vec{\sigma}\times\vec{\nabla}\right)\begin{pmatrix}\vec{\psi}_{q,u}\\ \vec{\psi}_{q,d}\end{pmatrix}, && (85)\\ W_q &= b_4\rho + b_4'\rho_q && (86) \end{aligned}$$

  The outer product between the Pauli matrices and the nabla operator gives

$$\vec{\sigma}\times\vec{\nabla} = \begin{pmatrix}\sigma_x\\ \sigma_y\\ \sigma_z\end{pmatrix}\times\begin{pmatrix}\partial_x\\ \partial_y\\ \partial_z\end{pmatrix} = \begin{pmatrix}\sigma_y\partial_z - \sigma_z\partial_y\\ \sigma_z\partial_x - \sigma_x\partial_z\\ \sigma_x\partial_y - \sigma_y\partial_x\end{pmatrix} \tag{87}$$

  resulting in

$$\left(\nabla\,W_q\right)\left(\vec{\sigma}\times\vec{\nabla}\right) = \begin{pmatrix}\partial_x W_q\\ \partial_y W_q\\ \partial_z W_q\end{pmatrix}\begin{pmatrix}\sigma_y\partial_z - \sigma_z\partial_y\\ \sigma_z\partial_x - \sigma_x\partial_z\\ \sigma_x\partial_y - \sigma_y\partial_x\end{pmatrix} \tag{88}$$

$$\begin{aligned} &= \left(\partial_x W_q\right)\left(\sigma_y\partial_z - \sigma_z\partial_y\right) + \left(\partial_y W_q\right)\left(\sigma_z\partial_x - \sigma_x\partial_z\right) \\ &+ \left(\partial_z W_q\right)\left(\sigma_x\partial_y - \sigma_y\partial_x\right). && (89) \end{aligned}$$

With that:

$$\begin{aligned} U_{q,so}\,\vec{\psi}_{q,u} &= [-\mathrm{i}(\partial_x W_q)\,\partial_y + \mathrm{i}(\partial_y W_q)\,\partial_x]\,\vec{\psi}_{q,u} \\ &+ [(\partial_x W_q)\,\partial_z - \mathrm{i}(\partial_y W_q)\,\partial_z]\,\vec{\psi}_{q,d} \\ &+ [\mathrm{i}(\partial_z W_q)\,\partial_y - (\partial_z W_q)\,\partial_x]\,\vec{\psi}_{q,d} \qquad (90) \\ U_{q,so}\,\vec{\psi}_{q,d} &= [-(\partial_x W_q)\,\partial_z - \mathrm{i}(\partial_y W_q)\,\partial_z]\,\vec{\psi}_{q,u} \\ &+ [\mathrm{i}(\partial_z W_q)\,\partial_y + (\partial_z W_q)\,\partial_x]\,\vec{\psi}_{q,u} \\ &+ [\mathrm{i}(\partial_x W_q)\,\partial_y - \mathrm{i}(\partial_y W_q)\,\partial_x]\,\vec{\psi}_{q,d} \qquad (91) \end{aligned}$$

5.7. Iteration of single particle states

These routines change/update the single particle states. Examples are convolutions with Green's function for the bound-state Helmholtz equation or orthonormalization.

- `iterate`: Ortho-normalizes the single particle states and updates them by applying the convolution with $G_{\mathrm{BSH},i(q)}$. First, we create vectors of functions for $U_{\mathrm{q,sky}}\,\vec{\psi}_{q,s}$ (`Upsi_qu` and `Upsi_qd`) by performing the corresponding of the potential with the single particle states. After that, we update `Upsi_qu` and `Upsi_qd` by adding the contribution from the spin-orbit potential $U_{\mathrm{q,so}}\,\vec{\psi}_{q,s}$ (see `Uso`) and the effective mass potential term $U_{\mathrm{q,meff}}\,\vec{\psi}_{q,s}$ (see `Umeff`).
  The single particle states are ortho-normalized in the `orthonorm` routine which also returns the energies of the single particle states. These are used to set up the Green's functions $\vec{G}_{\mathrm{BSH},i,q}$ in the `BSHoperators` routine, which are used to update `phi_qu` and `phi_qd` via convolutions of the Green's functions with `Upsi_qu` and `Upsi_qd`. The new states are normalized and compared to `psi_qu` and `psi_qd`. The maximum change in the single particle states `delta_psi` is determined. After that, `phi_qu` and `phi_qd` are mixed with `psi_qu` and `psi_qd` using the mixing parameter `chi`. The resulting updated single particle states are again normalized.

- `orthonorm`: This routine ortho-normalizes the single particle states. The Hamiltonian matrix `H_q` from eq.(40) is formed by first calculating the kinetic matrix `K_q` via `Kmatrix` and then adding the inner products of the single particle states with `Upsi_qu` and `Upsi_qd`. The overlap matrix `S_q` is calculated as inner products of the single particle states (see eq.(40)). We then use the built-in lapack routine to solve the eigenvalue problem for the single particle states `psi_qu` and `psi_qd`.
  If neither spin-orbit potentials nor effective mass potentials are included, we only ortho-normalize the single-particle states and then simply multiply them by the local nuclear potentials. The solution of the eigenvalue problem also provides the energies of the ortho-normal single particle states.
  Especially in the beginning of the iterations, it can happen that the energy of a state is positive. This will lead to problems when setting up the BSH Green's functions since these are only defined for negative energies. The problem can be solved via scaling the energies of the single particle states by a constant value `Es`. The reduction of the energy has to be done consistently for the built of the Green's functions and also by scaling `Upsi_qu` and `Upsi_qd`.
  The scaling energy is determined in the `orthonorm` routine in each iteration. We have a minimal default value for `Es` but replace it with twice the maximum energy of the single-particle states in case the latter is larger than the default value.

- `Kmatrix`: Calculates matrix of kinetic energies of the single particle states:

$$K_{ij,q} = -k_q \left(\int \psi_{i,q,u} \nabla^2 \psi_{j,q,u} + \int \psi_{i,q,d} \nabla^2 \psi_{j,q,d} \right). \qquad (92)$$

- `BSHoperators`: Creates a vector of $\vec{G}_{\rm BSH}$ for scaled single particle state energies:

$$G_{{\rm BSH},i,q}(\vec{r},\vec{s}) = \frac{1}{4\pi|\vec{r}-\vec{s}|}\exp\left(-\kappa_{i,q}\,|\vec{r}-\vec{s}|\right), \tag{93}$$

$$\kappa_{i,q} = \sqrt{-\frac{E_{i,q}-E_s}{k_q}}. \tag{94}$$

5.8. Output

There are different output files in the code. These include plotting files in vtk (`paraview_#iter.vts`) and csv (`densities_#iter.txt`, `potential_#iter.txt`) formats for potential and density profiles at iteration `iter`, and a log file (`log.txt`) for `maxerr`, `BE` aso. for each iteration step. The file `En_#iter.txt` contains information about the energy terms of the Skyrme density functional and energies of the single particle states.

The plotting and energy files are created in the `output` and `Ebinding` routines, respectively. The `log.txt` file is created and updated in the `doit` routine.

- `Energies`: Creates an output file with the single particle state energies. First, the matrix of kinetic energies is created in the `Potential` routine. The `Energies` routine uses the kinetic energies and the matrix of total single particle state energies to write an output file with a table that contains:
  1. The number of the single particle state
  2. The kinetic energy of the state
  3. The potential energy of the state (as difference between total and kinetic energies)
  4. The total energy of the state

  In the zeroth iteration, we calculate the potential energy as inner products

$$E_{i,q} = \int \psi_{i,q,u}\, U_{\rm q,sky}\, \psi_{i,q,u} + \int \psi_{i,q,d}\, U_{\rm q,sky}\, \psi_{i,q,d} \tag{95}$$

  and calculate the total energy as a sum of the kinetic and potential contributions. If the `En_#iter.txt` file has already been created in the `Ebinding` routine, the `Energies` routine appends the single particle state energies to this file.

- `Ebinding`: Calculates the five Skyrme Energy terms (see eq.(31) - eq.(34)), the Coulomb and Coulomb exchange energies (see eq.(37)) and the total kinetic energy (see eq.(35)). As in eq.(38), It adds all contributions to calculate the total binding energy per baryon. The routine writes out an energy file `En_#iter.txt` containing the energy contributions.

6. Examples

6.1. Relaxation of puren neutron matter

6.2. Relaxation of isospin symmetric nuclear matter at saturation density

6.3. Nuclear ground states for O^{16}, Pb^{208}, U^{238}, Mo^{110}

References

[1] R. J. Harrison, G. I. Fann, Z. Gan, T. Yanai, S. Sugiki, A. Beste, G. Beylkin, Journal of Physics: Conference Series 16 (2005) 243.

[2] R. J. Harrison, G. I. Fann, T. Yanai, Z. Gan, G. Beylkin, The Journal of Chemical Physics 121 (2004) 11587–11598.

[3] G. I. Fann, R. J. Harrison, J. Jia, J. Hill, D. Galindo, Proceedings from SciDAC (2010).

[4] G. I. Fann, J. Pei, R. J. Harrison, J. Jia, J. Hill, M. Ou, W. Nazarewicz, W. A. Shelton, N. Schunck, Journal of Physics: Conference Series 180 (2009) 012080.

[5] J. C. Pei, G. I. Fann, R. J. Harrison, W. Nazarewicz, J. Hill, D. Galindo, J. Jia, Journal of Physics Conference Series 402 (2012) 012035.

[6] J. C. Pei, G. I. Fann, R. J. Harrison, W. Nazarewicz, Y. Shi, S. Thornton, Phys. Rev. C 90 (2014) 024317.

[7] N. Chamel, P. Haensel, Living Reviews in Relativity 11 (2008). doi:10.12942/lrr-2008-10.

[8] A.-R. E. Mohammed, V. Sahni, Phys. Rev. B 29 (1984) 3687–3690.

[9] J. E. Robinson, F. Bassani, R. S. Knox, J. R. Schrieffer, Phys. Rev. Lett. 9 (1962) 215–217.