^{4}He-DFT BCN-TLS user Manual. Training school version

Ernesto García Alfonso, François Coppens

March 18, 2024

Contents

1 Prerequisites

1.1 General

- An internet connection.
- A recent version of the “Git”-program (`https://git-scm.com/downloads`).
- The Intel MKL/FFT or other Fast Fourier Transform library.
- A recent version of the Intel Fortran compiler (iFort).

1.2 DFT School

1.2.1 UNIX/LINUX

If you have a recent UNIX/Linux distribution installed on your computer, everything you need should already be there.

1.2.2 Windows

In principle, only a good terminal emulator is needed that supports “X11-forwarding”. Something like “PuTTY” will work fine. You can download it here:
`https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html`

2 General notes on obtaining and compiling the code

There are four main programs, each in its own repository on the GitHub website. They are respectively:

1. Static ^{4}He-DFT, supporting classical and quantum impurities.
2. External potential program to build dopant-helium droplet interaction
3. Time-dependent ^{4}He-DFT

To obtain the code, only one command in a terminal window is required. Change to the directory where you want the top level directory of the code to reside and execute

```
$ git clone <GitHub URL>
```

where <GitHub URL > needs to be substituted by on of the following URLs

1. `https://github.com/4He-DFT-BCN-TLS/He_DFT_2024/tree/main/He_statics`
2. `https://github.com/4He-DFT-BCN-TLS/He_DFT_2024/tree/main/External_potential`

3. `https://github.com/4He-DFT-BCN-TLS/He_DFT_2024/tree/main/Dynamics_TDDFT`

This is a shortest version manual for the training school purpose. The full one is available in the following link `https://github.com/4He-DFT-BCN-TLS/HE_DFT_manual`

After git has downloaded the repository from the Internet, change to the repository directory which should be named after the last part of the URL without the .git-part, and create a git "work"-branch

```
$ git checkout -b work
```

This will create a new git branch called "work" in which we can work, whilst not contaminating the "master"-branch and prevents merge conflicts in the "master"- branch during updates.

The next step is to create a makefile and tailor it to your machines architecture. You can use the provided "makefile" called makefile. It's are optimised for the Intel AVX architecture, with some extra machine specific optimisations for EOS's architecture. You can use it as-is or as a template for your own one. Beware that if you use your own FFT-library, make sure that they are locatable by the "make"-program. If you use Intel's MKL library you can use the environment variable $MKLROOT (see include makefile as an example). Start the compilation by executing.

```
$ make && make clean
```

After there are no errors one of the f4hedft, 4hedft-vortex, 4hetddft-isotropic, 4hetddft-anisotropicg executables should be available in the code directory, unless you changed the program name in your makefile.

Make sure that all your changes are staged and committed otherwise your own customisations could be overwritten after a "git pull"!

3 ^{4}HeDFT

3.1 First calculation: A pure $^4\mathrm{He}_N$ droplet

The easiest calculation to do is a pure $^4\mathrm{He}_N$ droplet. A typical droplet size used to study subjects like desorption dynamics and collisions is about $N = 10^3$ atoms and is $\sim$ 44 Å in diameter. It is large enough so that it exhibits the desired physical properties and at the same time small enough to keep it computationally feasible. This is why we shall start with a $^4\mathrm{He}_{1000}$ droplet.

The program will setup a simulation grid in cartesian coordinates. The helium density will be determined at each point of the grid, which should be large enough to contain the droplet. The mesh size should be $\sim$ 0.4 Å. But before we do that we need to pick and compile the needed program. In this case we need the static DFT program (**URL 1** in the list 2).

3.2 Preparing the calculation directory

It is good practice to create a separate directory for each calculation to keep all the input and output files together. First go back to your home directory as explained in the previous section. Once you are there, we are going to create a directory called "static" that will contain all the static calculations. Then inside that one the directory "pure-4he1000" will be created to host your first calculation.

```
$ mkdir static
$ cd static
$ mkdir pure-4he1000
$ cd pure-4he1000
```

Inside the code directory there is a folder named "input_files" which contains the input parameters for the calculation (pure-4he1000.input) and a script to submit the calculation to the computer cluster (pure-4he1000.sbatch). We copy these files into our calculation directory and make a "symbolic link" to the program:

```
$ cp ../../Code_statics/4hedft/input_files/pure-4he1000.input .
$ cp ../../Code_statics/4hedft/input_files/pure-4he1000.sbatch .
$ ln -s ../../Code_statics/4hedft/4hedft .
```

Before we submit our first calculation you have to be familiar with the input variables in the input-file.

3.3 The minimal set of input parameters

Here follows a description of a minimal set of paramaters that need to be setup for a successful calculation. There are more parameters than there are in this file. The full set of parameters that can be controlled will be given and explained later.

Table 1: **General Information**

title	the title of the calculation
nthread	number of OpenMP threads (CPU's) to use
mode	**0-** continue a previous calculation while reading all the input parameters from the density. An input density should be provided. This allows to stop and restart a calculation **1-** build, from scratch, a density for a pure $^4\text{He}_N$ droplet **2-** build, from scratch, a density for a $^4\text{He}_N$ droplet and a quantum impurity **3-** build, from scratch, a density for a $^4\text{He}_N$ droplet and a classical impurity **4-** continue a calculation but ignore the impurity's position in the start-density and use the position given in the input file/impurity-wave-function. Can be used to find e.g. • the ground state of a droplet with an impurity (classical only),starting from a pure droplet, or • the ground state of a displaced impurity (classical/quantum) under a constraint, starting from an unconstrained ground state **5-** build, a density for a $^4\text{He}_N$ droplet with a classical impurity starting from previously converged pure $^4\text{He}_N$ droplet density. **6-** build, a density for a $^4\text{He}_N$ droplet with n-classical impurities (n $\geq$ 2) starting from previously converged pure $^4\text{He}_N$ droplet density. Modes {2-6} are not relevant for a pure droplet but only mentioned here for completeness

Table 2: **Grid**

{x,y,z}max	the grid goes from -{x,y,z } max {x,y,z } max so the size is 2· {x,y,z } max . Typically ≈ 40 Å for 1000 He atoms
n {x,y,z}	number of grid points in the {x,y,z }-direction. Typically ≈ 200. The number needs to be a "magical number" dictated by the Fast Fourier library. Ideally $n = 2^k$ or $n = 2^k + 2^l$ with $l < k$

Table 3: **Droplet properties**

n4	number of ^{4}He atoms. Typically n4= 10^3
{x,y,z }c	droplet Center-of-Mass position. Usually (0,0,0) unless the problem requires something else.
core4	the used ^{4}He-functional . This field is set to “OT” automatically if “lsolid” is set to “.true.” • OT This is the default , no α_s term in the Orsay-Trento functional [1] • OTC Full Orsay-Trento functional [1] • OP Orsay-Paris functional [2]
lsolid	switch to *.true./.false.* to enable/disable the use of the Solid Functional. When it’s enabled , “OT” is used regardless of the value of *core4* [3]
lexternalpotential	*.true./.false.* if we want to read/no read using an external file which contain the total potential Impurities-He atoms
afermi	only important for densities starting from scratch. Its default value is set to 0.5 Å and it corresponds to the surface thickness of the initial density. [4 page 13/59 footnote)] Set to zero for a square density (surface thickness $\emptyset$)
limp	treat the impurity classically (=.false.) or quantum (=.true.)
lexternal	treat the classical impurity as an external field (=.true.)

Table 4: **Output-control**

pener	print an energy overview every *pener* $\in \mathbb{N}$ interactions
pdenpar	print a partially converged density every *pdenpar* $\in \mathbb{N}$ interactions
filedenout	filename of the final converged density

Table 5: **Computational parameters**

niter	total number of $n \in \mathbb{N}$. When n is reached, the program is stopped, even if the density is not converged
precie	precision of the convergence $0 <$*precie* $\in \mathbb{R} \ll 1$ is the size of the error in the total energy of the helium. The calculation stops when the size of the error in the total energy becomes less or equal to this value. Typically $\sim 10^{-7}$ ps
pafl	imaginary timestep size $\tau \in \mathbb{R}$. Usually $\tau \sim 0.05$
lpaflv	switch to allow for the imaginary time-step size to change during the calculation
nstepp	number of imaginary time-step changes
/	
1000 0.0001	use an imaginary time-step of 0.0001 for the first 1000 iterations
2000 0.0005	an imaginary time-step of 0.0005 for the next 2000
3000 0.0010	an imaginary time-step of 0.001 for the rest

3.3.1 Execution and output files

We can start the calculation in two ways. The first way starts the calculation on the current machine.

```
$ ./4hedft < pure-4he1000.input > pure-4he1000.res 2> pure-4he1000.err &
```

The "&" means that the program will be executed in the background so we keep our terminal free to use for other things. The second way is to use a submission script that uses the software installed on the computer cluster to schedule the job and execute it on a free computation node. The script is provided with the code and you already copied it in Section 3.2: *pure-4he1000.sbatch*. To submit the job execute

```
$ sbatch pure-4he1000.sbatch
```

You can view the status of your job by invoking

```
$ squeue -u <cecam user ID>
```

The standard output of the executable itself will be written to the file *pure-4he1000.res* and the errors to *pure-4he1000.err*. The program will now start to write some output files which will be explained in the following table

Table 6: **Output files**

DFT4He3d.namelist.read	a complete list of all the input variables, including the ones that were not explicitly given in the input file
den-{x,y,z}.dat	1-dimensional density profile in the {x,y,z} direction at the time of the last written partial density
den.out (or "filedenout")	the converged output density file
den0-{x,y,z}.dat	1-dimensional density profile in the {x,y,z}-direction at the start of the calculation
density.{iteration#}.out	partially converged density at time of imaginary timestep increment
partial.density{1,2,...}	partially converged density at {1,2,...}×pdenpar
pure-4he1000.error	file containing the messages to *stderr* generated by the program
pure-4he1000.res	file containing the messages to *stdout* generated by the program

To check if the calculation is running okay, check if the error in the total energy of the system is decreasing. This is written in the *pure-4he1000.res* file, as

```
ITERATIVE PROCEDURE
Total Energy (He)..........  −5400.344271 K +/−  −1.7363E−06 K
```

Every *pener* iterations. As long as the error is steadily decreasing, the calculation is running fine. The calculation will terminate whenever this value becomes smaller or equal to *precie* is or if the number of iterations reaches *niter*.

- **Note:** If the step is too large, the calculation might "explode" : either restart it again with a smaller time step or to restart from a previous partial.density# intermediate density file where the error was still decreasing.
- **Note:** If storage space is a problem keep only the last few intermediates density file for a possible restart and remove the first ones.

3.4 A droplet with a classical impurity X-^{4}He$_N$

Building the ground-state density for a droplet with a classical impurity can be done in two ways. Building the whole system from scratch, or using an existing droplet and adding the impurity. The easiest method is the first one, while the fastest methode is the second one; when adding the impurity to a converged pure droplet, it only needs to adapt locally, as long as the initial choice of the impurity location is not chosen too poorly.

For either case, the following parameters need to be changed and added

```
lexternal = .false. =) lexternal = .true.
```

and the following six new parameters in the input file need to be set

Table 7: **Added parameters**

mimpur	impurity mass in unified atomic mass units (u). E.g, potassium is ∼39 u
rimpur	radius of the impurity. Usually the distance (X-He) at the bottom of the X-^{4}He pair potential well is a good choice, about $3 - 4$ Å
{x,y,z}imp	location of the center of mass of the impurity relative to the calculation grid
selec_gs	the ground state pair interaction potential to use. These can be looked-up in the file “V impur.f90” in the code repository
r_cutoff_gs	cut-off radius for the pair interaction potential, usually set to thousands K
umax_gs	cut-off value for the pair interaction potential (see r_cutoff_gs)

As a consequence, there are also six new output files containing information about the interaction between the impurity and the droplet

Table 8: **Output files**

uext-{x,y,z}.dat	1-dimensional cuts of the external potential between the impurity and the helium at the start of the calculation
uext-{xy,yz,xz}.dat	same as *uext-{x,y,z}.dat* , but 2D-dimensional

3.4.1 Building a X-^{4}He$_N$ system from scratch

To build a X-^{4}He$_N$ system we now only need to change one additional input parameter

```
mode=1 -> mode=3,5,6
```

You will find an input file that is already prepared for this in the *input_files* directory called “*ck-4he1000-from_scratch.input*”. Now setup your calculation directory like explained in section 3.1.2 and start the calculation by executing

```
$ cd
$ cd static
$ mkdir ck-4he1000-fs
$ cd ck-4he1000-fs
$ cp ../../Code_statics/4hedft/input_files/ck-4he1000-from_scratch.input .
$ cp ../../Code_statics/4hedft/input_files/ck-4he1000-from_scratch.sbatch .
$ ln -s ../../Code_statics/4hedft/4hedft 4hedft
$ sbatch ck-4he1000-from_scratch.sbatch
```

Check the status of the job as before

```
$ squeue -u <cecam user ID>
```

and monitor the calculation itself by evaluating the output in "*ck-4he1000-from_scratch.inp*"

```
$ tail -f ck-4he1000-from_scratch.input
```

3.4.2 Building a X-^{4}He$_N$ system from a converged pure ^{4}He$_N$ droplet

To build the X-^{4}He$_N$ system we need to change the parameter (see Table [1])

```
mode=1 -> mode=4 or 5
```

and add one extra parameter to our *input-file*

Table 9: **Added parameters**

filedenin	filename of the converged input density of a pure droplet

Once again, prepare your calculation directory like before, with one extra addition. We need provide the file containing a converged density of a pure ^{4}He$_N$ droplet. This can be done in 3 ways:

1. copy the density into the calculation directory
2. make a symbolic link
3. give the full path in the input file

Be careful that the box size and number of points of the input density correspond to the box size and number of points in the input file.

An already prepared input file for this is in the *input_files* directory called '*'ck-4he1000-from_density.input*". Prepare the directory and start the calculation as before:

```
$ cd static
$ mkdir ck-4he1000-fd
$ cd ck-4he1000-fd
$ cp ../../code/4hedft/input_files/ck-4he1000-from_density.input .
$ cp ../../code/4hedft/input_files/ck-4he1000-from_density.sbatch .
$ ln -s ../../code/4hedft/4hedft 4hedft
$ ln -s ../pure-4he1000/den.out den.in
$ sbatch ck-4he1000-from_density.sbatch
```

3.5 He$_N$ droplet with several classical impurities X$_M$

In cases where we deal with multiples impurities, we can use the following program : `https://github.com/4He-DFT-BCN-TLS/He_DFT_2024/tree/main/External_potential`. You can store it in different path with respect to main DFT program.

The folder might look like as:

- My_directory which contains the following folders:

1. External_Potential: (X$_M$-He$_N$) potential interaction
2. ck-4he1000-fd : where I perform my DFT calculations

Table 10: **Input and Output files to build the X$_M$-He$_N$ Potential**

imp.input	It contains: the interaction potential between each impurity with all He atoms of the droplet (X$_M$-He$_N$), Impurities positions, r_cutoff_gs_k and umax_gs_k
potential_multi_impurity.input	Number of impurities, number of points of the grid, {xmax,ymax,zmax} for the He box.
potential_multi_impurity.out	Output file which contains the interaction potential of all X$_M$-He$_N$ for every grid point

You must define the next variable on the static input file (*ck-4he1000-from_scratch.input*) :

```
expotential ='potential_multi_impurity.out'
```

As above, you could copy the full interaction potential in the directory you are working on, make a symbolic link or give the full path in the input file.

3.5.1 Is there anything else to be included in the executable DFT work directory?

It's needed to prepare another input file other than the previous one we have already prepared. We name it as **"imp.input"**. It is going to contain the positions in x,y and z for each impurity.

```
rimp(1,1) = x1,     rimp(1,2) =y1,     rimp(1,3) = z1
rimp(2,1) = x2,     rimp(2,2) =y2,     rimp(2,3) = z2
     .
     .
     .
rimp(n,1) = xn,     rimp(n,2) =yn,     rimp(n,3) = zn
```

4 Dynamics $^4\mathbf{He}_N$

From a pragmatic point of view, the dynamics is much simpler than the statics. It only needs an input density and some initial parameters. It does not care whether the density is converged or not or if it is an eigen state of the Hamiltonian. It will simply take the input data and propagate it in time, even if the inputs are silly or make no physical sense. As long as the errors keep withing bounds, it keeps on going. It assumes the user knows what he/she is doing. In the following two sections we will give two simple dynamical processes as examples. The simplest will be a head-on collision between a heliophilic atomic impurity (Xe) and a $^4\mathrm{He}_{1000}$ droplet. The second more involved example will demonstrate the dynamical process of photo-exciting a heliophobic atomic impurity (K in a K-$^4\mathrm{He}_{1000}$ system) from the K(4s) state to an excited K(4p) state.

4.1 Impurities in a spherically symmetric electronic state

We will now prepare and execute a simulation of a head-on collision between a xenon atom and a $^4\mathrm{He}_{1000}$ droplet at 200 m/s. For this calculation we need two things:

- Time-Dependent ^{4}He-DFT for classical atomic impurities in an isotropic electronic state (Program 3. in the list in Chapter 2), and
- a converged density of a pure droplet so that our Xe has something to collide with.

Downloading and compiling the code is done exactly as before. For this it is assumed you are in your home-directory again. You can always return to your home directory by invoking cd and then press enter. Execute the following series of commands

```
$ module purge
$ module load intel/17.1 (or lastest versions)
$ cd code
$ git clone https://github.com/bcntls2016/4hetddft-isotropic.git
$ cd 4hetddft-isotropic
$ make && make clean
```

After this, you should have an executable called “*4hetddft-isotropic*” .

There is an example input file in */code/4hetddft-isotropic/input_files* called *“xe-he1000-head_on.input”* that we will use to start the calculation. Some of the fields that are in the input file are omitted here because they are already explained in the static chapter, while others are included again because they have changed meaning

Table 11: **New (or changed) input parameters**

mode	the different read modes to start or continue a calculation. **0-** start a dynamic calculation. You have to provide a file containing a density or wave function from a converged static calculation. **2-** continue a dynamic calculation. You have to provide a previously written file containing wave-function of a droplet and an impurity in an isotropic electronic state **3-** same as **mode = 2** but using a file containing a wavefunction of a droplet and an impurity in an anisotropic electronic state. In this case the electronic state of the impurity will be read, but no longer propagated in time. You can use this mode to study the dynamics of an impurity that underwent a transition from an anisotropic electronic state. These states will be discussed in the next section
selec	related to the selec_gs field in the statics. This field selects the He-X interaction potential. Can be looked up in V_impur.f90. E.g. for Xe this is “Xe_He”
r_cutoff	He-X interaction potential cutoff distance
umax	He-X interaction potential cutoff energy
v{x,y,z}imp	initial velocity of the impurity in ps/Å (1 ps/Å=100 m/s)
deltatps	the time step in picoseconds. A typical value is $5 \cdot 10^{-4}$
filerimp	name of the output file containing the impurity location as a function of time. Default name is rimp.out.#, where the # is an ID to distinguish between different runs. But you are free to name it as you want
filevimp	name of the output file containing the impurity velocity as a function of time. Default name is vimp.out.#
fileaimp	name of the output file containing the impurity acceleration as a function of time. Default name is aimp.out.#, where the # is an ID to distinguish between different runs. But you are free to name it as you want
pcurr	this field is related to pdenpar discussed before but now it prints a complex wave function instead of a helium density. The filenames of these files are density.#.dat, where the # is an integer ID.
txmean, tymean, tzmean	sets the location of the mean of the absorbtion potential (see Ref. 4 “Absorbing potential at the box boundaries”). A good rule of thumb is to use : txmean = xmax-2 ,tymean = ymax-2 ,tzmean = zmax-2 Å
txsurf, tysurf, tzsurf	It is defined as the distance from (txmean, tymean, tzmean) to the simulation box boundary. E.g : The absorption potential should act from [-tzsurf,tzsurf] around tzmean.
Λ_0	Damping absorption parameter, typically = 2, see [4]

To prepare our calculation directory and start the simulation execute the following commands

```
$ cd
$ mkdir dynamic
$ cd dynamic
$ mkdir xe-he1000-head_on
$ cd xe-he1000-head_on
$ cp ../../code/4hetddft-isotropic/input_files/xe-he1000-head_on.input .
$ cp ../../code/4hetddft-isotropic/input_files/xe-he1000-head_on.sbatch .
$ ln -s ../../code/4hetddft-isotropic/4hetddft-isotropic 4hetddft-isotropic
$ ln -s ../../static/pure-4he1000/den.out den.in
$ sbatch xe-he1000-head_on.sbatch
```

As usual, you can view the status of your job by invoking

```
$ squeue -u <cecam user ID>
```

Check that is has the status “RUNNING”. When this is the case, the program will generate the following output files.

Table 12: **Output files**

density.#.dat	file containing the full 3-dimensional wave-function of the helium droplet and also the position and velocity of the atomic impurity, if present (otherwise these are set to 0). The file has a header containing information on the number of iterations, the used time-step, the current evolution time, etc, at the time of writing. This information can be extracted when needed. The "#" in the filename is an integer ID that labels the files. The time difference between two files is always deltatps$\times$pcurr.
filerimp	output file containing the impurity position as a function of time. It is structured in 4 columns: time(ps), x(t), y(t), z(t). The coordinates are referred to the origin of the grid.
filevimp	output file containing the impurity velocity as a function of time. It is structured in 4 columns: time(ps), v_x(t),v_y(t), v_z(t). The coordinates are referred to the origin of the grid.
fileaimp	output file containing the impurity accelleration as a function of time. It is structured in 4 columns: time(ps), a_x(t),a_y(t), a_z(t). The coordinates are referred to the origin of the grid
Energies	it is set as 6 columns : time-current, eimpu-impu (impurity-impurity energy), eimpu (impurities total energy), ekinx (impurity total kinetic energy), eHeX (He-impurity/ies interaction energy), etot (total energy impurity/ies-He)
hedenini-{x,y,z}	1-dimensional cuts along the box?s axis of the initial helium density.
*.err	file containing the run-time errors
*.res.#	file containing the results. # is an integer ID labeling the different runs of the program in case a calculation has been interrupted, either intentionally or unintentionally
timec.dat	file containing the time coordinates
uext.dat	Fourier transform of the helium-impurity interaction potential
uimp-{x,y,z}	helium-impurity interaction potential

4.2 Continuing a previous dynamic calculation

Table 13: **New input parameters**

icurr	ID of the current input wave-function. If file containing the droplet's wave function (and λ-vector if applicable) that you are going to use to continue is called "density.123.dat", then icurr=123
iter0	current number of iterations. This number is included in the header of the wave-function file.
time0	current evolution time This number is also included in the wave-function file

5 Internal Units of the Programs

We give here some details on the units used in the calculations. In the helium functional definition the energy unit is Kelvin and the length unit is Å . In the dynamics the time scale is most often ps, and hence velocities appear in Å/ps, (1 Å/ps = 100 m/s). Some ingredients such as pair potentials have to be written in these units before being used. The energy scale of the absorption and emission spectra is cm^{-1}. Let us recall some equivalences and constant values taken from the **NIST** table of constants (`http://physics.nist.gov/constants`):

- Recall that: 1Å= 10^{-10}m ; 1fm= 10^{-15}m; 1ps= 10^{-12}s
- Boltzmann constant: $8.6173303 \cdot 10^{-5}$ eV/K; hence 1 eV = 11604.522 K
- 1 cm^{-1} = 1.43877736 K
- Atomic mass unit: $m_u c^2$ = 931.4940954 MeV = $1.080954 \cdot 10^{13}$ K
- $\hbar = 6.582119514 \cdot 10^{-16}$ eV s =7.638235 K ps
- c=299792458 m/s
- $\hbar$c = 197.3269788 MeV fm = $2.289885 \cdot 10^{7}$ K Å
- He mass : $4.002 \times 931.49 = 3727.8$ MeV

When solving the effective Schrödinger equation for the superfluid, the $\hbar$ constant has been incorporated into the definition of "time". This means that the equation actually solved is

$$i \frac{\partial \Psi}{\partial (t/\hbar)} = \mathrm{H}\Psi \qquad (1)$$

For the sake of consistency, this way of handling time has also been adopted for the dynamic evolution of the impurity. As a consequence, what internally is called "velocity" of the impurity (v_{inter}) for instance, is actually $\frac{d\mathbf{r}}{d(t/\hbar)}$ instead of $\frac{d\mathbf{r}}{dt}$. One may recover the "physical" impurity velocity v_{phys} by dividing the internal velocity v_{int} by $\hbar$, namely $v_{phys} = v_{int}/\hbar$.

Similarly, in order to obtain the kinetic energy of the impurity I in K, as any other energy printed and/or stored during the dynamics, one has to keep in mind that

$$\frac{1}{2} m_I v_{phys}^2 = \frac{1}{2} \frac{m_I}{\hbar^2} v_{inter}^2 \rightarrow \frac{1}{2} \frac{m_I c^2}{(\hbar c)^2} v_{inter}^2 \tag{2}$$

Writing the impurity mass in units of m_u, $m_I = M_I\ m_u$ and taking for M_I the value in atomic mass units as given in the atomic mass list of **NIST** (which takes into account the natural isotopic composition of the impurity), one obtains the kinetic energy of the impurity in K as

$$E_{kin,I} = \frac{1}{2} M_I \frac{m_u c^2}{(\hbar c)^2} v_{int}^2 \rightarrow \frac{1}{2} M_I \times 0.02061484\ v_{int}^2(\mathrm{K}) \tag{3}$$

since $\frac{m_u c^2}{(\hbar c)^2} = 0.02061484\ \mathrm{K}^{-1}\ \text{Å}^{-2}$. M_I values for the impurities used in some recent applications are e.g. $M_{Rb} = 85.4678$; $M_{Cs} = 132.9055$; $M_{Xe} = 131.293$

Often we evaluate the kinetic energy of the impurity in K from its velocity in hundreds m/s, so better have a simple-to-use expression

$$E_{kin,I}(\mathrm{K}) = \frac{1}{2} M_I m_u c^2 (\frac{v}{c})^2 \rightarrow 0.6013618 M_I (v_{\text{Å}/ps})^2 \tag{4}$$

For Cs at 200 m/s, $v^2 = 4$, $M_{Cs} = 132.9055$ and this yields $E_{kin,Cs} \sim 319.3$ K

6 Plotting Programs

```
sudo apt install gnuplot   (Linux)
sudo port install gnuplot  (MacOS)
```

References

[1] F. Dalfovo and A. Lastri. "Structural and dynamical properties of superfluid helium: A density-functional approach". *Physical Review B*, 52(2), 1995.

[2] J. Dupont-Roc, M. Himbert, N. Pavloff, and J. Treiner. "Inhomogeneous Liquid ^{4}He: A Density Functional Approach with a Finite-Range Interaction". *Journal of Low Temperature Physics*, 81(1/2), 1990.

[3] F. Ancilotto, M. Barranco, F. Caupin, R. Mayol, and M. Pi. "Freezing of ^{4}He and its liquid-solid interface from density functional theory". *Physical Review B*, 72(214522), 2005.

[4] Manuel Barranco, François Coppens, Nadine Halberstadt, Alberto Hernando, Antonio Leal, David Mateo, Ricardo Mayol, and Martí Pi. “Zero temperature DFT and TDDFT for ^{4}He: A short guide for practitioners”. Unpublished, 2017.