PROFESS 3.0 User's Manual

This manual that accompanies PROFESS (v3.0) in the CPC Program Library was updated by Mohan Chen, Junchao Xia, Johannes M. Dieterich and William C. Witt. It is based on the manual version 2.0.

Outline

In Section 1, we briefly introduce OFDFT methods. In Section 2, we report the new updates of PROFESS 3.0, based on PROFESS 2.0. In Section 3, we introduce how to obtain and install PROFESS. In Section 4, a quick start guide is provided. In Section 5, input and output files are introduced. Following that, we include a glossary of keywords in Section 6, as well as some troubleshooting help in Section 7. Finally, in Section 8, a list of references concludes this User's Manual.

1. Introduction

Orbital-free density functional theory (OFDFT) is a first-principles quantum mechanics method to find the ground-state energy of a system by variationally minimizing with respect to the electron density.[1] No orbitals are used in the evaluation of the kinetic energy (unlike Kohn-Sham DFT), and the method scales quasi-linearly with the size of the system. More precisely, since Fast Fourier Transforms (FFTs) are used to compute many of the terms, the algorithm scales as O(*Nlog(N)*), where *N* is a measure of the system size. In OFDFT, a material's electronic ground-state properties are completely determined by its 3-dimensional electron density and an external, *e.g.*, ionic, potential.

The Princeton Orbital-Free Electronic Structure Software (PROFESS) uses the OFDFT framework to model materials from the atomic scale to the mesoscale.[2,3] At the start of a simulation, PROFESS reads the various input files that describe the geometry of the system (ion positions and cell dimensions) and the types of elements (defined by electron-ion pseudopotentials), the desired actions for it to perform (minimize with respect to electron density and/or ion positions and/or cell lattice vectors, or molecular dynamics simulations), and the various options for the computation (such as which functionals to use). Based on these inputs, PROFESS sets up the computation and performs the appropriate optimizations to obtain the desired results. We will describe the format of these input files in further detail in the following sections.

Depending on the output setting, PROFESS can periodically report on the computation progress in output files. Cell and ion geometries, atomic forces, stresses on the cell, electron density, and the potential (first functional derivative of the energy with respect to the electron density) can also be output throughout the optimization, or just at the end of the optimization.

A climbing-image nudged elastic band (CINEB)[4,5] program – interfaced to PROFESS – finds saddle points with respect to ion movement. It takes the initial and final ion positions (and

whatever intermediate positions you would like to specify) as input, and proceeds to find the saddle point. It outputs the geometry of that state as well as its energy.

2. Changes from PROFESS v2.0

2.1 General updates

- Streamlining: most module files are reconstructed, including splitting large modules into several small ones but with clear functions. These reconstructions improve the readability of PROFESS, reduce the compilation time, and make implementation and contribution of new functions to PROFESS easier for external developers.
- The format of the output file is improved to be more informative and clearer.

2.2 Interfaces to external libraries

- FFTW 2.1.5 is replaced with FFTW3-API interfaces, F95-API in serial mode allowing for linking against either FFTW3 or a recent MKL and F2003-API in parallel mode.
- Spin-polarized Perdew-Burke-Ernzerhof (PBE)[6] is implemented through the Libxc library.[7]
- Import of a subset of the latest Numerical Analysis Library (NMS) with improvements for F95 standard compliance.

2.3 New kinetic energy density functionals

- Huang-Carter (HC) kinetic energy density functional (KEDF) is fully supported (with force and stress).[8]
- A Density Decomposition Method (DDM) with fixed localized density is supported (no force or stress yet).[9]
- Wang-Govind-Carter (WGC) decomposition (WGCD) KEDF is supported (with force, but no stress yet).[10]
- Enhanced von Weizsäcker (EvW)-WGC KEDF is supported (with force, but no stress yet).[11]
- Thomas-Fermi with G-enhanced von Weizsäcker (TFGvW)-KEDF model,[12] and many GGA functionals including Lee-Lee-Parr (LLP), DePristo-Kress (DK), DePristo-Kress 87 (DK87), Ou-Yang-Levy 1 (OL1), Ou-Yang-Levy 2 (OL2), Perdew-Wang 96 (PW86), Perdew-Wang 91 (PW91), Lembarki and Chermette (LC94), Tran-Wesowlowski (TW02), PBE2, E00, P92, Becke 86A (B86A), Becke 86B (B86B), and Thakkar (Thak) functionals.[13] The forces are available, while the stresses for these functionals are not yet implemented.

2.4 New functions

- Spin-polarized calculations are supported.[14] A similar density optimization algorithm comparable to Ref. 15 is implemented to handle two sets of densities. Note that the

reference density and therefore kernels of non-local KEDFs (except for the HC KEDF where the kernel for each spin channel is computed on-the-fly explicitly) are still simply derived from the total density, not from electron density of each spin channel.

- The Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS)[16] method is added for density optimization, which yields better stability and efficiency for spin-polarized and vacuum-containing systems.
- Molecular dynamics methods with three ensembles are implemented: the microcanonical NVE (constant number of particles N, constant volume V, and constant energy E), the canonical NVT (constant number of particles N, constant volume V, and constant temperature T), and the isothermal-isobaric NPT (constant number of particles N, constant pressure P, and constant temperature T) ensembles. Detailed algorithms can be found in the work by Martyna.[17] A reference application simulating the melting point of lithium can be found in Ref.18.
- An option to set the initial density as a superposition of atomic densities is added.

2.5 Problems fixed

- A bug in PROFESS 2.0 related to cutoff functions used to treat vacuum is fixed.
- Ion-ion interaction using the Ewald method (without the b-spline method) is now parallelized.

3. Obtaining and installing PROFESS

3.1 Obtaining the Code

CPC subscribers may download PROFESS v3.0 from the Computer Physics Communications Program Library. For current instruction of how to proceed, please refer to the CPC website at http://www.cpc.cs.qub.ac.uk/.

3.2 Compiling the Code

You must have an FFTW3-compliant library, including header files, installed. You will also need to download and compile your own Libxc library.

The PROFESS directory includes a Makefile and a directory called “Source” containing all the source files. A directory called “test” contains tests and other necessary codes.

The Makefile contains various compilation options for different platforms. The code has been tested primarily using Intel (version 13.0.1) and GNU compilers (versions 4.6+). In general, any compiler implementing the Fortran95 and 2003 standards should be usable. Before attempting to compile, check the compiler commands (for Fortran90 and C++ compilation) and corresponding flags that are uncommented in the Makefile (comment lines have “#” at the beginning of the line). If those are not appropriate for your system, check whether any of the commented-out architectures are more similar, and/or add your own flags and compiler names to fix any discrepancies.

Once the appropriate compilers and flags have been placed in the Makefile, one can build the serial version of PROFESS by typing:

```
make
```

on the command line. This will create the executable, named "PROFESS," for the serial version of the software, and a directory called "obj," where all the intermediate files and modules are placed.

To make the parallel version of the executable, one must type:

```
make parallel
```

The executable created in this case is named "pPROFESS" with object files place in "pobj."

To delete all working compilation files, start over, and recompile everything, type:

```
make clean
```

To make the executable for the CINEB saddle point search code (currently a different executable than PROFESS), one must type:

```
make cineb
```

and an executable "CINEB" will be generated. The CINEB program requires the serial version of PROFESS to run, so do not forget to compile the "PROFESS" executable, as well.

To build the utility to translate density file formats for different purposes, such as plotting the electron density, type:

```
make convert
```

When modifying any source code or the Makefile, or compiling on a new architecture, always try compiling with debug flags first. If the code runs, add in optimizations one at a time. Also, linking statically to the FFTW library, if possible, may allow the code to execute faster.

4. Quick start guide

"I Just Want to Run the Code. Just Tell Me How to Run the Code!"

4.1 PROFESS (serial version)

Here, we will assume minimal knowledge and walk you through the process of creating a simple test directory and running the code.

The executable for PROFESS is called "PROFESS." When you run it, it assumes that it will find at least 3 files in the working directory: an input file (.inpt), an ion file (.ion), and at least

one pseudopotential file (.recpot or .realpot). You must have these files or PROFESS will not run.

The OFDFT executable takes one argument, which is the name of the input file (without the .inpt suffix.)

The .inpt file contains keywords specifying the parameters and options you want the code to execute. The .ion file contains the geometry of your system. In addition, PROFESS requires a pseudopotential file for each ionic species (for describing the ion-electron interaction). Reciprocal space pseudopotentials must use the .recpot extension, while real-space pseudopotentials must use the .realpot extension. The formatting of these files is described in detail in Section 50.

For now, create a directory for the calculation you wish to run, for instance, "mytest." Inside that directory, create a file with the same name as the directory (though this is not required) and an .inpt extension, say, "mytest.inpt." Inside this file put the following line of text:

```
ecut 600
```

This is an input file stripped down to the bare minimum. You have just told PROFESS that you want it to use a plane-wave energy cutoff of 600 eV. If you don't specify at least this one line, the code will exit and print out an error message. That input is all that PROFESS *needs*. Of course, PROFESS will try to make guesses about all the parameters you did *not* specify, and sometimes you might not like the decisions it makes. For instance, this calculation will optimize the electron density only (holding ion geometry or cell lattice fixed). The option defaults are detailed further in Section 60 (keywords), and you should specify the options you want. We have strived to enable reasonable defaults, though, based on what is most commonly used.

Usually, it is a good practice to at least specify a few more basic things in the file, for instance:

```
method NCG
kinetic WGC
```

The 'method NCG' line tells the program you want to minimize the energy with respect to the electron density, using a variant of the conjugate gradient algorithm that uses the square root of the density, while keeping the electron density number conserved during line search. The keyword 'kinetic WGC' lets the program know you want to use the double density dependent WGC99 KEDF. There are many more (basic and more advanced) things you can specify, but we will leave it for now in the interest of simplicity.

Next, you will need to create an .ion file, for instance, "mytest.ion." For now it will be more convenient to name it the same base name (mytest) as the .inpt file, but there is an option to

specify in the input file the name of the associated .ion file. Add the following lines to "mytest.ion:"

```
%BLOCK LATTICE_CART
3.97 0 0
0 3.97 0
0 0 3.97
%END BLOCK LATTICE_CART
%BLOCK POSITIONS_FRAC
Al 0 0 0
Al 0.5 0.5 0
Al 0 0.5 0.5
Al 0.5 0 0.5
%END BLOCK POSITIONS_FRAC
%BLOCK SPECIES_POT
Al al_HC.lda.recpot
%END BLOCK SPECIES_POT
```

An .ion file has at least three sections: the first section describes the cell vectors in Å, the second section describes the ion positions (in this case, in fractional coordinates), and the third section stores the names of the pseudopotential files associated with the ion types. Your newly-written ion file represents a 4-atom face-centered cubic (FCC) aluminum cell.

The third file you will need is a local pseudopotential (LPS) file, one for each atomic element in your simulation. A variety of LPSs developed within the Carter group are available for download at:

http://www.princeton.edu/carter/research/local-pseudopotentials/

In the example .ion file above, we are using the Huang-Carter aluminum bulk LPS.[12] You can either download it from the website above or copy "al_HC.lda.recpot" from the provided "test" directory in the PROFESS package to your own "mytest" directory.

You should now be able to run this simple OFDFT calculation. Remember, PROFESS takes a single argument, basically the name of the input file without the .inpt suffix. So, if you go to your test directory and type, for instance:

```
/home/<username>/PROFESS/PROFESS mytest
```

PROFESS will perform the calculation that we have just set up, and should only take a few seconds. A message should appear upon program completion. Two files should also be generated: "test.err" and "test.out." The "test.err" file is the error file and tells you whatever problems occurred during processing. It should be empty. The "test.out" file is the output file that prints out the information about the run, the details of the calculation, and the energy at the electron density distribution that gives the minimum energy. (Note: by default, it will not output the electron density at the minimum energy, but you can request this information. See Section 6 for further details.)

4.2 Submitting PROFESS to a queue

Running the command above is also known as "running on the head node." This is generally discouraged except for systems with a small number of atoms (typically less than 10).We will walk you through an example for a PBS-style queuing system. Depending upon your cluster configuration and the queue setup, this example must be altered. Please check with your local systems administrator for the recommended submission setup.

Instead of the head node, you should submit your job to the queue. This is safer and, most of the time, quicker. Create a file in your test directory, e.g., call it "qPROFESS." This is a script file that will be executed by the queue. It can be as short as one line (or as complicated as you would like to make it). For now, it suffices to have one line in the file:

```
/home/<username>/PROFESS/PROFESS <your-test-filename>
```

in addition, you may need to make the submission script executable:

```
chmod u+x qPROFESS
```

Now you should be able to submit this job to the queue:

```
qsub –cwd qPROFESS
```

(The –cwd flag will run from the current directory you are in and not dump out a number of junk files into your home directory)

You can use the command "qstat" to monitor the status of your job, and the command "qdel <*process id-number*>" to kill the job, if desired. Type "man qsub" at the command prompt for more details.

Writing a script to run parallel PROFESS on multiple processes is a little trickier, as the submission script setup depends on the scheduler running on the computer. Here is an example of a script that will run a job on 8 cores on a supercomputer at Princeton:

```
#!/bin/bash
#PBS -l nodes=1:ppn=8,walltime=0:05:00
#PBS -q risk
#PBS -r n

UNITCELL=test

module load openmpi
cd $PBS_O_WORKDIR

mpiexec /home/<username>/PROFESS/pPROFESS $UNITCELL
```

Now that you have a taste for how PROFESS works, we will go into detail about the different abilities of the software. After a brief note about grid sizes and an introduction to

CINEB in the next sections, we will examine all the possible options for the input files, as well as the format of the input and output files, in Section 5.

4.3 Note about grid sizes in PROFESS

The lower limit to the grid density in the simulation is set using the keyword 'ECUT' or 'GDEN'. However, the actual grid size may be larger than requested to allow for more efficient FFTs. In particular, the grid will not be the multiple of any prime number larger than 7. A possible exception to this arises when using parallel PROFESS. With parallel PROFESS, the grid sizes in the 2nd and 3rd lattice directions are always divisible by the number of processors used for the computation for better load balancing and may be increased. It is therefore important to use a grid that is converged at the minimum number of CPUs.

4.4 CINEB quick start

To run the CINEB code, you need 5 files in one folder: the executable named OFDFT (rename the serial "PROFESS" or create a link to it), a CINEB.param file (see Section 5.7 for details), a CINEB.inp file (see Section 5.8 for details), a template ORBITALFREE.inp file (see Section 5.9 for details), and your pseudopotential files. Right now, you just type:

```
CINEB
```

within the directory to run the code. This may however change in future PROFESS versions.

When you do this, it should create two output files, one brief and one detailed. The brief one will be called cineb.out, while the verbose one is cinebimages.out.

As CINEB runs, the geometries of images are optimized in parallel, with one image per core. Note that we have not yet implemented the ability for a single image optimization to be parallelized when using CINEB, so each image's simulation size must be small enough to fit on a single core.

5. Input and output files

5.1 Input keywords file (.inpt)

This file is intended to describe all the parameters and options with which you may want to run the code.

Each line of the input file must contain one and only one instruction for the program, or be a comment. Comment lines start with the pound character (#). An instruction starts with a keyword (tells the program what option you wish to select: for instance, "ecut" says you are about to set the plane-wave kinetic energy cutoff) and may be followed by any number of options (for instance, "1200" if you want the plane-wave energy cutoff to be 1200 eV). An option is either a numerical value or a string of characters. It specifies what the keyword actually does. Only the first four characters of a keyword are read (so for instance, the keyword "geometryfile" will be interpreted the same as "geom" and "geomitryphile").

Similarly, only a truncated version of a literal option is read. Numerical values are read in their entirety.

Throughout the whole file, case (uppercase, lowercase) is unimportant. Also, if you specify conflicting options, sometimes the code will tell you if it is a particularly egregious offense, but more often than not it will run based on the last option you specified (whatever is farther down the file).

We give a list of the keywords in the Section 6, along with any possible and default values.

5.2 Geometry file (.ion)

The .ion file is where the information about the geometry of the system is specified. Generally, it is easiest to name the .ion file the same base name as the .inpt file, since it will be recognized immediately; otherwise, one can specify the name of the .ion file in the .inpt file.

Here is a sample .ion file:

```
%BLOCK LATTICE_CART
4.03 0 0
0 4.03 0
0 0 4.03
%END BLOCK LATTICE_CART
%BLOCK POSITIONS_FRAC
Al 0 0 0
Al 0.5 0.5 0
Al 0 0.5 0.5
Al 0.5 0 0.5
%END BLOCK POSITIONS_FRAC
%BLOCK SPECIES_POT
Al Al.recpot
%END BLOCK SPECIES_POT
```

Whitespace in the .ion file is ignored.

There are three sections (BLOCKS) in the .ion file above. Each block has a header and footer, which begin with the keyword “%BLOCK” and “%END BLOCK,” respectively. The sections need not be in the order presented above, though this is probably the most logical order.

The first section identifies the unit cell vectors. The unit cell can be cubic (as above) but can be as general as triclinic. The header in this case indicates that the lattice will be specified in Cartesian coordinates. Each line corresponds to the x, y, and z components of a lattice vector in Å.

Another option to specify the cell is to use %BLOCK LATTICE_ABS instead of %BLOCK LATTICE_CART. Instead of specifying the unit cell vectors in Cartesian coordinates, this will allow you to specify the length of each lattice vector in Å and three angles (in degrees):

```
%BLOCK LATTICE_ABS
4.03 4.03 4.03 90 90 90
%END BLOCK LATTICE_ABS
```

The second section identifies the ions in the system. In the above example, each line following the header %BLOCK POSITIONS_FRAC corresponds to an ion. The first field, "Al," consists of up to two letters and signifies the element symbol of the ion, in this case aluminum. This is followed by (in the above case) the fractional coordinates of the ion with respect to the cell lattice vectors specified above.

Ion coordinates can also be specified in absolute Cartesian coordinates that are independent of the cell lattice vectors. To do this, use the header %BLOCK POSITIONS_ABS or equivalently %BLOCK POSITIONS_CART instead (with the appropriate footer, of course).

The last section (%BLOCK SPECIES_POT) identifies the pseudopotential files to be used with the ions that are specified. (See below for a brief description of pseudopotentials.) For each ion type that is used in the system, there must be one pseudopotential file. Therefore, each line in this block consists of a one or two letter element name (*e.g.*, Al) followed by the file name of the corresponding pseudopotential. This pseudopotential file is expected to be available in the same directory as the .ion file.

Apart from these three mandatory blocks, an optional fourth block can be defined in the .ion file to indicate which ion positions are to be optimized. This block is bracketed by the header and footer %BLOCK ION_OPTIMIZATION and %END BLOCK ION_OPTIMIZATION. This block must be located after the ion positions block. During force minimization, ions are allowed to relax in directions indicated with "1," but are held frozen in directions indicated by "0:"

```
%BLOCK ION_OPTIMIZATION
0 0 0
1 1 1
1 0 1
1 1 1
%END BLOCK ION_OPTIMIZATION
```

In the example given here, the 1st ion is not optimized at all; it is held fixed in place. On the other hand, the 2nd and 4th ions can relax in all directions. The 3rd ion can only relax in the x and z directions (assuming an orthorhombic cell). If the ion_optimization block is completely missing, all ions are allowed to optimize their positions in all directions. If the ion_optimization block contains a smaller number of (non-whitespace) lines than the ion positions block, then PROFESS will quit with an error. If the ion_optimization block contains more, only enough lines are read to match the number of ions in the system, *i.e.*, additional lines are ignored.

Geometry files that are output from PROFESS have the same formatting as the input geometry files, but contain the geometry associated with the current optimization step. The filenames are either <*systemName*>.<#>.ion, where the number corresponds to a successful force minimization step, or <*systemName*>.ion.out for the final geometry.

Other optional blocks include a core density block (used in the density decomposition scheme with fixed localized density[9]) and an atomic density block (to set initial densities as a combination of atomic densities). Both are located within the corresponding keywords: %BLOCK SPECIES_CORE and %END BLOCK SPECIES_CORE, and %BLOCK SPECIES_RHOA and %END BLOCK SPECIES_RHOA. Within the block, the input is similar to the pseudopotential block, namely the element followed by its corresponding file. For example,

```
%BLOCK SPECIES_CORE
Al Al.CORE
%END BLOCK SPECIES_CORE
```

will be the input for the core density of Al. Additionally, you will have to prepare a file named "Al.CORE" (just be consistent with that specified in the .ion file), including the core density. The core densities and atomic densities must be provided in reciprocal space and they use all the same file format: the file should begin with one line with two numbers, the first one telling the number of following lines (or the number of grid points in reciprocal space) and the second for the increment or spacing between reciprocal space data points (Δg). After the first line, all following lines should also have two numbers, the first being g values and the second being the electron density (both in reciprocal space with atomic units $Bohr^{-1}$ and $Bohr^{-3}$). A sample file can be found in the Tests folder. The density may be converted from real space using a Fourier transform; a code (written in C++) to perform this conversion is also included under the "Source" directory, for more input/output information, you can simply compile the code using the command 'c++ PROFESS_CoreDensityG.cpp' and run ./a.out to take a look at the instructions.

5.3 Pseudopotential input file (.psp)

LPSs in reciprocal space follow the format of CASTEP™ .recpot files. This file format starts out with an arbitrary size comment block that ends with the "END COMMENT" line. The next line has two integers that correspond to the version number of the pseudopotential (ignored in PROFESS). The next line corresponds to the maximum g value of the pseudopotential in $Å^{-1}$. Then, the values of the pseudopotential are displayed in units of $eV*Bohr^3/Å^3$, three to a line, starting with the g=0 value. The values are evenly spaced in g, and the total number of values is divisible by three (so that the last line of values in the pseudopotential file contains the full three values.). The pseudopotential is terminated by a one-line terminator consisting of only "1000."

5.4 Electron density file (.den)

The electron density file is used for input as well as output. When read in, it will be the initial guess to the optimized density. When used for output, it corresponds to the latest electron density computed. If no electron density input file is provided, PROFESS begins optimization with a uniform electron density.

The density file contains the total electron density of the system. As such, this file is often large – well in the GB range. Thus, we recommend you store these files only temporarily, and delete them whenever they become unnecessary. To limit the total amount of space that this file takes up, you can use the 'RHOM' keyword to set a maximum file size. In that case, the file size will not exceed the requested maximum. If necessary, the electron density is linearly interpolated to output a smaller array. These smaller files can be useful for creating plots of electron density. However, the loss of detail means that these interpolated files are less useful as restart densities.

The density filename follows the form <*systemName*>.<#>.den, where # is some integer. If the density is output multiple times in the program, the # will start from 0 and count up.

In the density file, the dimensions of the array are listed first, and then electron density values are output (in atomic units) in column major order (i.e., we print the density for each spin; for each spin channel, the outer loop is for the z index, middle loop for y index, and inner loop for x index). However, since the file is written using direct access, there may be no space between adjacent values, making the file difficult to read without additional conversion. The density output file can be converted to Tecplot format (please see http://en.wikipedia.org/wiki/Tecplot for more details) using the RhoConvert utility.

5.5 General output file (.out)

The .out output file provides information about the calculation in a format that should be easy for the user to read.

One can control the output via the 'PRIN' command in the .inpt file. Since almost everything is printed into the .out file, we have taken measures to make .out file a bit easier to read. First, we have made it so that each line that is output will be categorized to a section and tagged. Lines dealing with density optimization start with a line ("|") character, lines dealing with geometry optimization start with a bracket ("[") character, and lines dealing with the cell lattice start with a semi-colon (";") character. An easy way to isolate all of the lines that deal with the geometry optimization is to grep the output file for this character:

```
grep "\[" <systemName>.out
```

When using keywords PRIN MIN DEN > 1 and/or PRIN MIN GEO > 1, information will be output at each iteration of the optimization. What follows are the headers associated with each optimization method and explanations of each column:

```
+-------------------------------------------------------------------------------+
|DL Step       Energy        Pot. Norm #LCG /#Line Linestep  #FFT   Time (s)  |
|     #         (eV)         (in code) Steps/Steps   Size   (cumul) (cumul)   |
+-------------------------------------------------------------------------------+
```

Above is the header for all electron density optimizations. "DL" is used as a comment prefix. "Step #" is the current iteration number. "Energy" is the total energy of the system at the current iteration. "Pot. Norm" is the norm of the potential (change in the Lagrange function – which is related to the total energy – with respect to the change in the variational parameter, *e.g.*, sqrt(rho)) in atomic units. "#LCG Steps" gives the number of iterations needed to find the Newton direction. For CG optimization, this value is always 0 because we do not use the Newton direction. "#Line Steps" gives the number of steps for the line search to converge and to satisfy the Wolfe conditions. "Linestep size" is the value of θ,[19] the line search variable. The last two columns, "#FFT" and "Time," give the cumulative number of Fourier transforms that were called as well as the total optimization time, respectively.

Occasionally, there will be problems with convergence in the algorithms. Perhaps the line search does not converge after some upper limit of iterations, or the direction chosen does not decrease the energy. Informational, warning, or error flags will then print in either one of the first two columns: "D" for directional issues and "L" for line search issues.

The D flags are:

M = Warning: maximum number of truncated Newton iterations exceeded
+ = Information: direction reset to the steepest descent direction
***** = Warning: residual dropped by less than 1% between TN iterations
< = Warning: Hessian is not positive definite
? = Error: unknown truncated Newton exit. There's probably a bug.

The L flags are:

m = Warning: Maximum number of line search iterations exceeded
w = Warning from line search algorithm (check .err or .out file)
e = Error in line search algorithm (check .err or .out file)
> = Error: energy change along line search is positive
? = Error: unknown truncated line search exit. There's probably a bug.

For ion geometry optimizations, the headings are slightly different depending on which optimization method is chosen. If conjugate gradient optimization (METH ION CON) is used, then the heading is as follows:

```
[-------------------------------------------------------------------------------]
[       Conjugate Gradient Ion Geometry Minimization Report                     ]
[-------------------------------------------------------------------------------]
[ Step Line  MaxForce    Energy   LineSearch  LS   SD?   Time                   ]
[  #  Step#  (eV/A)     (eV)         Step    Flag       (s)                    ]
[-------------------------------------------------------------------------------]
```

"Step #" gives the current CG iteration number, while "Line Step #" is the iteration of the line search (a nested loop within the CG iterative optimization). "MaxForce" gives the largest magnitude force felt by any one atom. "Energy" is the total energy of the system. "LineSearch Step" is the value of the variable used for the line search. "LS Flag" is a flag indicating the convergence status of the line search (more about that below). "SD?" indicates whether the steepest descent direction is used instead of the conjugate gradient direction. The direction is periodically reset because the geometry landscape is nonlinear, which can also reset itself if the CG direction looks bad. Finally, "Time" is the amount of time spent on the current iteration.

The line search uses the final iteration from the previous step as a starting point, and performs at least two additional line search steps per CG iteration. The "LS Flag" indicates the status of the line search as follows:

0 = Line search not yet complete. Continue line search.
1 = Line search algorithm converged.
2 = Line search algorithm exited because of other convergence thresholds (energy, atomic displacement, force cutoff) reached.
3 = Warning: line search exited by exceeding maximum number of iterations.
4 = Other warning from line search algorithm. Exited line search.
5 = Error from line search algorithm. Exited line search.

In the example below, two steps in the same CG iteration are shown. The second step converges. Note that any unconverged line search steps are marked by "%" symbol in the first column.

```
[%     3     1    4.927363E-02    -2.317591E+02    1.061E+01     0    F        0.048 ]
[      3     2    8.828586E-03    -2.317595E+02    1.061E+01     1    F        0.038 ]
```

The alternative ion optimization algorithm in PROFESS is Quickmin, which optimizes ion positions using a molecular dynamics-based scheme. The header for the Quickmin scheme (METH ION QUI), shown below, also has the "Step #," "MaxForce," "Energy," and "Time" columns defined the same as in CG ion optimization. However, there are two new columns: "Next Timestep" is the time step for the next step in the scheme, and "Next Velocity" is the sum of the magnitude of the velocities of all atoms used for the next step. The time step is adaptive but occasionally the step size is too large, as indicated by an increase in energies or forces pointing in an opposite direction. Steps that are too large will be indicated in the "Overstep?" column. Also, if the overstep is severe and results from that step are thrown out, output from that iteration will have the "%" symbol in the first column.

```
[-------------------------------------------------------------------------------]
[                  Quickmin Ion Geometry Minimization Report                    ]
[-------------------------------------------------------------------------------]
[ Step  Over    MaxForce        Energy        Next        Next         Time   ]
[   #   Step?    (eV/A)          (eV)       Timestep    Velocity       (s)   ]
[-------------------------------------------------------------------------------]
```

5.6 Other output files

One other output file will always be present. The error output file (.err) is where the error messages generated in PROFESS will be output. If this file is not empty, PROFESS has encountered one or more problems, which should be detailed within. Some other errors and warnings will be sent to standard output. You can redirect those messages to a log file, if desired.

One optional output is the force output file (.force). Forces at a given step can be output in the .out file, in a .force file, or in no file at all. After the header, force information is printed out in five columns. The first column contains the element name, the second column given the ion index, and the third through fifth columns give the forces in the x-, y-, and z-directions.

Another optional output is the potential output file (.pot). The potential (change in total energy with respect to the optimization parameter) is given in the same format as the .den electron density file, but has the .pot file extension instead.

The output file used to interface with CINEB is the .trans file. This is used only as an intermediate file which outputs the energy, forces, stress, and ion positions in the system and can be read and used by the CINEB code in a no-frills manner. (Note: Although the CINEB currently does not need the stresses, the stresses would be necessary for the solid state NEB (SSNEB) algorithm.[20])

5.7 CINEB.param file

The CINEB.param file is one of the three input files needed for CINEB calculations based on PROFESS; the other two files are CINEB.inp and ORBITALFREE.inp, respectively, as described below. There are only three command lines required in the CINEB.param file. However, these three lines must be written in a restricted order (first line is NUDGE_TYPE, second line is INIT_COND, and third line is FORCE_CUT). Here is an example of a CINEB.param file:

```
NUDGE_TYPE 1
INIT_COND 1
FORCE_CUT 0.01
```

The explanation of each option:
NUDGE_TYPE {integer}: Type of method that will be used. **-1:** No band. **0:** Elastic band. **1:** Nudged elastic band. **2:** Climbing image nudged elastic band.
INIT_COND {integer}: Initial conditions for first and last image. **1:** No geometry optimization for first and last images. **2:** Relax first and last images.
FORCE_CUT {real}: The maximum force (eV/Å) per atom for a relaxed band.

5.8 CINEB.inp file

The CINEB.inpt file is one of the three input files for CINEB calculations based on PROFESS. It contains geometry information for each image. Here is an example of a CINEB.inp file:

```
TITLE
3.43 0 0
0 3.43 0
0 0 3.43
AtomNumber 2
AtomType 1
ImageNumber 3
Li Li.PBE.recpot
F
Image 1
Li 0 0 0
Li 0.5 0.5 0.55
Image 2
Li 0 0 0
Li 0.48 0.48 0.50
Image 3
Li 0 0 0
Li 0.5 0.5 0.45
frozenAtoms
1 1 1
1 1 1
```

Explanations for above example:
Line 1 is a title that will not affect any calculations.
Lines 2-4 are the lattice vectors for a simulation cell (unit is Å).
Line 5 is the number of atoms in the cell.
Line 6 is the number of atom types.
Line 7 is the number of images.
Line 8 gives the pseudopotential file for a given atom type.
Line 9 specifies the type of atomic coordinates. 'F' stands for fractional coordinate while 'C' stands for Cartesian coordinate.
Starting from Line 10, one needs to write down all image geometries starting with a caption.
Finally, in the last section of the CINEB.inp file, one needs to write down indicators for the x, y, and z coordinates of each atom, to specify each coordinate as either moveable (1) or fixed (0).

5.9 ORBITALFREE.inp file

The ORBITALFREE.inpt file is one of the three input files for CINEB calculations using PROFESS. Here is an example of an ORBITALFREE.inp file:

```
TRAN ON
ECUT 600
PRINT FORCE SEP
```

This file is the same as the .inpt file for each image calculated by PROFESS. The keywords are described in full detail in Section 6. However, you will need to pay attention to three

parameters. First, the 'TRAN ON' is necessary for every CINEB calculation within PROFESS. This flag will turn on PROFESS, writing essential information to the .trans file. Information in the .trans file will be automatically read and analyzed by the CINEB code. Second, 'ECUT' or 'GDEN' is a necessary parameter to determine the energy cutoff in PROFESS. Third, 'PRINT FORCE SEP' is also required to print out the .force.out file. After finding the existence of '.force.out,' the CINEB code will collect the necessary data and then do the subsequent step.

6. Keywords

6.1 About keywords

The only mandatory keyword in PROFESS is 'ECUT' or 'GDEN,' which decides the grid density for the simulation. However, when only using one keyword, the calculation defaults to certain optimization algorithms and energy density functionals, which may not be the ones you want to use. To utilize PROFESS to your full advantage, more keywords should be specified. The most commonly-used keywords are MINI (whether to perform a single point or geometry optimization), METH (optimization algorithm), KINE (kinetic energy functional), EXCH (exchange-correlation functional), and SPLI (calculation method for ion-ion and ion-electron terms).

A full list of PROFESS keywords makes up the remainder of this section.

6.2 Grid Parameters

Keywords that determine the grid size and boundary conditions.

ECUT {real}

MANDATORY! Sets the kinetic **E**nergy **CUT**off for the plane wave basis to the value specified (in eV). This sets the density of the Fourier grid, and, correspondingly, the density of the real-space grid. Basically, the higher the 'ECUT' value, the more accurate the computation and the longer the computational time. The keyword 'GDEN' can be used to set the grid density directly as an alternative. If this is not set (and 'GDEN' is not set either), the simulation will not run.

GDEN {real}

Sets the real-space **G**rid **DEN**sity to the value specified, in points per Å. This option is an alternate to 'ECUT;' these keywords must not be used at the same time.

DIME {option}

Sets rules that the **DIME**nsions of the grid meshes must satisfy. Useful in some circumstances, but otherwise best left alone. Other options are:

ALL: No special parity. **(Default)**

EVE: Only even grid sizes are selected along the x, y, and z directions. Note that there is a small error for even grid sizes (due to how the mapping of q-vectors works in the fast Fourier transform library). It should not be significant; however, be aware of it.

ODD: Only odd grid sizes are chosen. There should be no error for odd grid sizes, but FFTs may take longer to compute.

TWO: The grid size in all three directions is restricted to powers of 2. The FFT routines work fastest with grid dimensions that are powers of two. Note that this keyword may increase your grid significantly.

6.3 Optimization Parameters

Keywords that determine the optimization methods and convergence criteria.

MINI {option}

Determines whether we are **MINI**mizing the energy, forces, and/or stresses. The various options are:

RHO or **DEN**: Only electron density, rho, is optimized. **(Default)**

ION or **GEO**: The ion positions inside the unit cell are relaxed to minimize forces, and electron density is optimized at each ion step.

CEL or **STR**: Lattice vectors of the simulation cell are optimized to minimize stresses. At each cell relaxation step, ion positions and electron density are optimized. Note: To preserve the fractional coordinates of the ions while relaxing the cell, use the keyword METH ION NONE (see below).

NVE: Perform molecular dynamics using the microcanonical ensemble (velocity Verlet integrator).

NVT: Perform molecular dynamics using canonical ensemble (Nosé-Hoover thermostat[21] and velocity Verlet integrator).

NPT: Perform molecular dynamics using isothermal-isobaric ensemble (Parinello-Rahman barostat,[22] Nosé-Hoover thermostat, and velocity Verlet integrator).

MAXI {option}

This keyword determines the maximal step number for minimizing the energy, forces, and/or stresses. The various options are:

RHO or **DEN {integer}**: Maximal density optimization steps. **(Default=500)**

ION or **GEO {integer}**: Maximal ion optimization steps. **(Default=200)**

EXT {integer}: Maximal extra electronic iteration steps, only useful when one uses the WGCD KEDF[10] or EvW KEDF.[11] **(Default=50)**

CEL or **STR {integer}**: Maximal cell relaxation steps. **(Default=100)**

MAG {integer}: Maximal number of steps for spin-polarized calculations to converge the magnetization. **(Default=100)**

METH {option}

This keyword specifies the **METH**od to use for density/ion/cell relaxation. Square root refers to the variable being optimized: the square root of the density. Acceptable values for the option are:

NON: No electron density optimization. This option only makes sense when no other optimization is requested and an initial electron density is supplied.

NTN: Square root truncated Newton minimization method with line search conserving total electron number (The prefix 'N' stands for electron number). **(Default)**

NCG: Square root conjugate gradient minimization method with line search conserving total electron number.

NBF: Square root BFGS method with line search conserving total electron number.

STN: Square root truncated Newton (TN) optimization for electron density with line search not conserving total electron number. ('METH NTN' is recommended instead of 'METH STN' for general usage)

SCG: Square root conjugate gradient (CG) minimization for electron density with line search not conserving total electron number. ('METH NCG' is recommended instead of 'METH SCG' for general usage)**HYB**: Hybrid STN and SCG method.

LOG: Log truncated Newton (TN) optimization. Use log(rho) as the optimization variable. This experimental addition to the arsenal has difficulty converging in the presence of vacuum.

ION or **GEO {option}**: Sets the ion relaxation method. This option takes one of the following sub-options:

NON: No ionic relaxation.

QUI: Quickmin algorithm.[23] A molecular-dynamics-type optimization scheme.

CON: Conjugate gradient ion optimization using the Dcsrch subroutine. **(Default)**

CG2: Conjugate gradient ion optimization based on home-made line search code; more robust than calling Dcsrch subroutine.

BFG: BFGS ion optimization method.

CEL {integer}: Whether to optimize the cell in a single dimension or in all dimensions. Integer=1, 2, or 3 corresponds to relaxing the cell only along the 1st, 2nd, or 3rd lattice vector, respectively. For an orthorhombic cell, this corresponds to relaxation in only the x, y, or z direction. If 'MINI CEL' is specified (cell lattice optimization) and this keyword is not used – or an integer other than 1-3 is given – cell relaxation occurs in all dimensions.

DENC {integer}

For electron **DEN**sity: select which **C**G algorithm to use with 'METH NCG.' Options:

1: Polak-Ribiere. Standard CG algorithm.

2: Hager-Zhang (generally faster).[24] **(Default)**

3. Steepest descent instead of CG.

IONC {integer}

For **ION** optimization: select which **C**G algorithm to use with 'METH ION CG2.' Options:

1: Polak-Ribiere (more robust).

2: Hager-Zhang (generally faster).[24] **(Default)**

3: Dai-Yuan (never tested).

CONV {integer}

Criterion used to check **CONV**ergence of electron density optimization. Valid if electronic optimization method 'METH NTN,' 'METH NBF,' and 'METH NCG' are used. (If 'METH MIX' is chosen instead, only the potential norm criterion is available.) Options include:

1: Total energy changes less than 'TOLE.' **(Default)**

2: Potential norm is less than 'TOLP.'

3: Both energy and potential must satisfy convergence criteria.

4: Considered converged if either energy or potential satisfies its convergence criterion.

MBFG {integer}

The number of storage steps in L-BFGS method, only used in 'METH NBF' density minimization method. **(Default=5)**

CALC {option}

CALCulate and output force and/or stress.

FOR: Once the system is optimized as requested using the 'MINI' keyword, the forces on the ions are computed and added to the appropriate output file.

STR: After complete energy minimization (as specified by the 'MINI' keyword), the stress will be computed and printed in the main output file. Equivalent to 'PRIN STR.' By default, 'CALC STR' does not take place.

TOLE {real}

Sets the **TOL**erance of the **E**nergy change (in eV) for determining the convergence of the electron density optimization. **(Default=10^{-6} a.u. or equivalently ~2.72E-5 eV)**

TOLP {real}

Sets the **TOL**erance of the **P**otential (in atomic units) for determining the convergence of electron density optimization. **(Default=1E-5 a.u.)**

TOLS {real}

Sets the **TOL**erance of the **S**tress (in atomic units) for determining the convergence with respect to cell vector optimization. **(Default=5E-7 a.u.)**

TOLF {real}

Sets the **TOL**erance of the maximum **F**orce (in eV/Å) on an ion when forces are considered to be converged. **(Default=5E-5 Hartree/bohr or equivalently ~2.57E-3 eV/Å)**

TIME {real}

Sets the initial size of the force minimization **TIME**step. For use with 'METH ION QUI.' **(Default=0.4)**

PREC {option}

The Garc á-Cervera **PREC**onditioner[25] for truncated Newton electron density optimization methods. This preconditioner generally speeds up the calculation when in use, but can lead to instability and convergence issues when the system contains large amounts of vacuum (surfaces, nanowires, clusters, molecules). Options: **ON** or **OFF**. **Default is OFF**. The preconditioner is only used with 'METH STN' or 'METH HYB' electron density optimization methods.

RHOU

The keyword **RHO U**niform forces electron density optimization to start from a uniform density at every step of an ion geometry optimization. This generally makes computations more expensive, since we do not take advantage of the optimized electron density from the previous geometry. However, it may be more stable when there are large changes in the ion configuration. If this keyword is not used, a uniform initial density is not enforced.

SPIN {integer}

Selects the number of **SPIN**s in the electron density. The value must be 1 (non-spin-polarized) or 2 (spin-polarized) to be physical. **(Default=1)**

MAGM {real}

The initial total magnetic moment of system, in unit of μB. **(Default=0.0)** If 'MAGM' is set along with 'FIXM,' the magnetic moment of system is fixed to 'MAGM' value.

FIXM

If set, the magnetic moment of system is constrained to be the initial value **MAGM** during the whole optimization. **(Default is false).**

6.4 Energy Functional Parameters

Keywords that set the kinetic and exchange-correlation energy density functionals and that determine how external energy and potential calculations are done.

KINE {option}

Sets the **KINE**tic energy density functional. Parameters that pertain to these options can be set using the PARA keyword. Choose one of the following options:

THO or **TF**: Thomas-Fermi functional, to be multiplied by the parameter 'PARA LAMB' (λ).[26,27]

VON or **VW**: von Weizsäcker functional, to be multiplied by the parameter 'PARA MU' (μ).[28]

TF+: The sum of Thomas-Fermi times 'PARA LAMB' (λ) and von Weizsäcker times 'PARA MU' (μ).

WT: The sum of the TF+ functional and the Wang-Teter kernel term (non-local term). The entire functional depends on 'PARA LAMB' (λ), 'PARA MU' (μ), 'PARA ALPH' (α), and 'PARA BETA' (β).[29,30]

WTV: Same at WT, except with a cutoff implemented for stability in vacuum. Takes setting additional 'RHST' and 'RHOV.' Note the stress calculation called using this functional is for the plain WT functional: WTV stress is not implemented (nor would it make sense).

WGC: The sum of the TF+ functional and the Wang-Govind-Carter kernel term. The functional takes parameters 'PARA LAMB' (λ), 'PARA MU' (μ), 'PARA ALPH' (α), 'PARA BETA' (β), and 'PARA GAMM' (γ). Note: For stability reasons, the electronic optimization methods LOG and MIX currently truncate the Taylor expansion of the WGC kernel at first order for both the energy and the potential at the beginning of the calculation. You may observe an increase in energy of the first step after the second order contribution is introduced.[31] **(Default)**

WGV: Same as WGC, except with a cutoff implemented for stability in vacuum. Takes additional 'PARA RHST' and 'PARA RHOV.' Note the stress calculation called using this functional is for the plain WGC functional: WTV stress is not implemented.

LQ: Thomas-Fermi plus von Weizsäcker plus the low q term. Developed by Jeng-Da Chai from the Weeks group. Highly experimental. Use at your own risk. Stress calculations using this functional are not implemented.[32,33]

HQ: Same as LQ but the low q term replaced with the high q term. Untested and not guaranteed to work. Stress calculations using this functional are not implemented.

CAT: KEDF with linear response of a free-electron gas. Possible input 'PARA CATG' (gamma in CAT KEDF). Stress calculations using this functional are not implemented.[34]

CAV: Same as CAT, except with a cutoff implemented for stability in vacuum. Takes additional 'RHST' and 'RHOV.' Stress calculations using this functional are not implemented.

HC: Huang-Carter KEDF.[8] The main parameters used to set up this KEDF are 'PARA RR,' 'PARA HCLA,' 'PARA BETA,' and 'PARA CUTR.'

DEC: WGC-decomposition (WGCD) KEDF.[10] The main parameters used to set up this KEDF are 'PARA M,' 'PARA SCFC,' and 'PARA RHOC.' The parameters used to set up the interaction KEDF part are 'PARA LAM' and 'PARA MU.'

DEV: WGC-decomposition (WGCD) KEDF[10] for vacuum.

EVT: EvW-WT KEDF based on WT.[11] The main parameters used to set up this KEDF are 'PARA KLOC,' 'PARA ALOC,' and 'PARA TOLK.'

EVC: EvW-WGC KEDF based on WGC.[11] The main parameters used to set up this KEDF are 'PARA KLOC,' 'PARA ALOC,' and 'PARA TOLK.'

GGA: Multiple single-point GGA functionals. Additional string is needed after 'GGA' (*e.g.*, KINE GGA LLP). Available functionals include (input characters in parentheses): TF, vW, TFλvW (TF+), Lee-Lee-Parr (LLP), DePristo-Kress (DK), DK87, Ou-Yang-Levy 1 (OL1), OL2, Perdew-Wang 86 (PW86), PW91, Lembarki and Chermette (LC94), Tran-Wesolowski (TW), PBE2, E00, P92, Becke 86A (B86A), Becke 86B (B86B), and Thakkar (Thak).[13]

vW+: vWGTF model where *G* can adopt either analytical or numerically interpolated forms.[12]

UDDM {real}

Use Density Decomposition Method (DDM);[9] tests have only been performed for HC KEDF + DDM. In principle, DDM uses a fixed localized density and it can be used to combine with any kind of KEDF (described by 'KINE' keyword) to treat the delocalized density. The parameters used to set up the interaction KEDF are 'PARA ATF' and 'PARA BVW.'

PARA {option}

This command allows you to set the values of **PARA**meters in the KEDFs.

LAMB: Specifies the prefactor for the Thomas-Fermi kinetic energy functional. Defaults: 0 for vW functional, not defined for CAT/CAV, 1 for all others. 'LAMB' is also used to specify the TF coefficient of the interaction KEDF in the WGCD method.

MU: Specifies the prefactor for the von Weizsäcker kinetic energy functional. Defaults: 0 for TF, 1/9 for TF+, not defined for CAT/CAV, 1 for all others. 'MU' is also used to specify the VW coefficient of the interaction KEDF in the WGCD method.

ALPH: Specifies the parameter alpha in the WT/WGC KEDF and associated vacuum cutoff KEDFs. Default is $\frac{5+\sqrt{5}}{6}$ for WGC and 5/6 for WT.

BETA: Specifies the parameter beta in the WT/WGC KEDF and associated vacuum cutoff KEDFs. Default is $\frac{5-\sqrt{5}}{6}$ for WGC and 5/6 for WT. This parameter is also used as the 'β' parameter in the HC KEDF.

GAMM: Specifies the parameter gamma in the WGC KEDF; a proper range is 2-4.[35] **(Default=2.7)**

CATG: The gamma parameter in the CAT KEDF. Default is 1.4, which is optimal for aluminum in the diamond and fcc phases, and for Al fcc (111/100/110) surface energies.

RHO0: Specifies the reference density for the linear response in the linear-response-based kinetic energy functionals. Default: The average electron density of the system. Unit is electron number per $Å^3$.

RHOS: Specifies the (constant) reference density for the Taylor expansion in the WGC kinetic energy functional. (In general, this should be set equal to 'PARA RHO0.') Default: The average electron density of the system. Unit is electron number per $Å.^3$

MRHO: If 'PARA RHOS' is not specified, 'PARA MRHO' is used as a multiplier of 'PARA RHOS,' mostly used in WGCD. **(Default=1.0)**

PBEC: When the gradient of the density on a real space grid is smaller than this value, the gradient corrections for exchange-correlation potential and energy will be set to zero. Our PBE-XC is not stable in vacuum, possibly causing trouble in density optimization. If you

need to use this, a setting of 1E-3 is good. **(Default=0, which means no cutoff in PBE XC)**

AL5: Coefficient multiplying the WGC Taylor expansion term. Used only when keyword WGCT is set to -5.[36] **(Default=1.0)**

BE5: Coefficient multiplying the WGC Taylor expansion term. Used only when keyword WGCT is set to -5.[36] **(Default=1.0)**

RR: Ratio between bins for interpolating kernel, only used in Huang-Carter KEDF.[8] **(Default=1.15)**

HCLA: Lambda value used in Huang-Carter KEDF.[8] **(Default=0.01)**

CUTR: Density in HC KEDF smaller than 'CUTR' will be set to 0 during interpolation steps.[8] **(Default=0.0)**

ATF: Coefficient of TF term in interaction KEDF term, used only in DDM.[9] **(Default=0.8)**

BVW: Coefficient of VW term in interaction KEDF term, used only in DDM.[9] **(Default=0.2)**

M: This parameter is only used to tune the scale function in WGCD KEDF.[10] **(Default=0.0)**

SCFC: Convergence criteria of maximal difference between $F_{i+1}(r)$ and $F_i(r)$; *i* is the extra iteration index required to converge F(r). 'SCFC' is used only in WGCD KEDF.[10] **(Default=1.0e-4)**

RHOC: Used to tune F(r) and then rho0 when vacuum treatment is introduced. RHOC is only used in the WGCD KEDF. **(Default=6.84e-3)**

KLOC: K parameter in EvW KEDF.[11] **(Default=0.0)**

ALOC: A parameter in EvW KEDF.[11] **(Default=0.0)**

TOLK: Tolerance for convergence of the K parameter in EvW KEDF.[11] **(Default=1.0e-8)**

CP: Coefficient of the penalty term in GGA functionals. Only needed for vW and TF+ KEDFs.[12] **(Default=0)**

VWMO: Model type for vWGTF KEDFs. **1** for analytical *Gvsd*; **2** for analytical *ELFvsd*; and **3** for interpolated *ELFvsd*. See Ref.132 for details. **(Default=1)**

WGCT {integer}

Order of the **WGC T**aylor expansion. While the second order expansion is generally recommended, in some cases it is not a very physical term. This term contributes very little in aluminum and it diverges for silicon or gallium cases. In some cases, you may need WGCT= -3 to make the WGC functional converge.[36] **0**: only the 0th order of WGC kernel will be used.

1: WGC kernel expanded to 1st order only.

2: WGC kernel expanded to 2nd order. **(Default)**

-3: WGC kernel expanded to 2nd order, but set $\frac{\partial^2\omega}{\partial\rho^2(r)}=0$.

-4: WGC kernel expanded to 2nd order, but set $\frac{\partial^2\omega}{\partial\rho(r)\partial\rho(r')}=0$.

-5: WGC kernel expanded to 2nd order, but $\frac{\partial^2\omega}{\partial\rho^2(r)}$ and $\frac{\partial^2\omega}{\partial\rho(r)\partial\rho(r')}$ terms are multiplied by coefficients 'PARA AL5' and 'PARA BL5,' respectively.

RHOV {real}:

The electron density at which the vacuum cutoff function quickly goes to zero, in atomic units. **(Default=10^{-6})**

RHST {real}:

Controls how fast the vacuum cutoff function goes to zero. **(Default=10^{-6})**

HOLD {option}

HOLD ρ_0 and/or ρ^*(used in WT or WGC functionals) to their initial values instead of allowing them to change whenever the cell volume changes. This should be used when performing a cell optimization likely to increase/decrease the cell size or the amount of vacuum in the system. Default is off: ρ_0 and ρ^* values are not held fixed. 'HOLD' is automatically turned on if 'PARA RHO0' or 'PARA RHOS' are set to fixed values.

RHO0: Holds the average density ρ_0 of a given system to its initial value.

RHOS: Holds the average density ρ^* that is used to do Taylor expansion in WGC KEDF to its initial value.

EXCH {option}

This command sets the **EXCH**ange-correlation functional. The option should be one of the following:

LDA: Local density approximation of Perdew and Zunger as exchange, Ceperley and Alder as correlation.[37,38] **(Default)**

PBE: Perdew-Burke-Ernzerhof (PBE) GGA exchange-correlation functional.[39]

EWAL {option} {real}

This keyword sets the options for ion-ion calculations using an **EWAL**d summation. Values should not need to be changed from their defaults. This keyword can be used in conjunction with 'SPLI AL' or 'SPLI II' below.[40,41]

TOL: This option sets the Ewald error tolerance in eV. **(Default=10^{-10})**

MAX: Maximal interaction cutoff distance allowed for ions in real space, unit is Å. **(Default=12.00)**

ETA: This option sets the eta increment in $Å^{-1}$. **(Default=0.01)**

REA: The value of this option sets the real-space cutoff increment in $Å^{-1}$. **(Default=0.01)**

SPLI {option} {integer}

This keyword selects cardinal b-**SPLI**ne approximations for the ion-ion and/or ion-electron terms. The exact computation of these terms scales quadratically with the system size and can slow down large calculations tremendously. The B-spline approximations reduce the scaling to near-linear and remain sufficiently accurate. The integer corresponds to the order of the spline and should be even. Higher values will provide a better approximation but can take longer to compute. Orders 10-14 are found to be a good compromise in practice.[3]

AL: Use cardinal b-splines at the specified order for all (both ion-electron and ion-ion) computations. **(Default)**

IE: Use cardinal b-splines at the specified order for ion-electron computations only.[42]

II: Use cardinal b-splines at the specified order for ion-ion computations only.[43]

NO: Do not use any cardinal b-splines. This option should not be followed with an integer value.

6.5 Molecular Dynamics Parameters

Specify parameters used in molecular dynamics simulations.

MDOP {path}

The file path for dumping geometry and velocity files. **(Default is the directory that .inpt and .ion files exist in)**

RSTM {integer}

Restart molecular dynamics from the prepared atom file and velocity file.
<0: Restart molecular dynamics from the selected step, note that here we use negative values to avoid confusion when this value is set to 1. For example, if one wants to start from step N (integer), set this value to -N and then PROFESS will read restart from geometry file 'ion.restart.N' and velocity file 'vel.restart.N.'
0: Turn off this function. **(Default)**
1: Restart from specific geometry file ion.restart and velocity file vel.restart.
>1: These values are not valid.

NVEM {integer}
Parameters for NVE ensemble.
TEMP {real}: Initial temperature, unit is Kelvin.
TIME {real}: Total time of simulation, unit is second.
DTIM {real}: Time step, unit is second.

NOSE {option}
Parameters for NVT ensemble with Nosé-Hoover thermostat.
TEMP {real}: Temperature, unit is Kelvin.
TIME {real}: Total time of simulation, unit is second.
DTIM {real}: Time step, unit is second.
QMAS {real}: Mass of thermostat, in atomic units.(5.0×10^3 works well for a 128-atom Li cell,[18] for example. For other systems, users should do tests by choosing different masses before doing any production runs. This suggestion also applies to the barostat mass in the NPT MD simulations.)
NRES {integer}: Parameters to set an '*n*'th order of Trotter factorization[44] of the time-evolution operator associated with the Nosé-Hoover thermostat. 'NRES' stands for the '*n*', which can be chosen as an integer number larger than 1. A larger '*n*' can lead to more accurate trajectories. **(Default=3)**
NYOS {integer}: Parameters to further set an '*m*'th order of factorization of time-evolution operator according to Yoshida's scheme.[45] For each different '*m*' order, a different set of weighting parameters are used. Available '*m*' values are 1, 3, or 5 in the current version of code. **(Default=3)**

NPTM {option}
Parameters for NPT ensemble.
TEMP {real}: Temperature, unit is Kelvin.
TIME {real}: Total time, unit is second.
DTIM {real}: Time interval, unit is second. Typical length is 0.1 fs, which is written as 1.0e-15.
QMAS {real}: Nosé-Hoover thermostat mass, in atomic units. (e.g., 5.0×10^3 works well for a 128-atom Li cell.)
BMAS {real}: Barostat mass, in atomic unit. (e.g., 2.5×10^7 works well for 128-atom Li cell.)[18]
EXTP {real}: External pressure, unit is Pa. **(Default=0)**
NRES {integer}: this variable is used in NPT but with the same meaning as 'NOSE NRES'.
NYOS {integer}: this variable is used in NPT but with the same meaning as 'NOSE NYOS'.
CONS {integer}: CONStrain type, here we assume the simulated cell is rectangular.
<0: No constraints.
1: Set the non-diagonal elements in stress tensor to be zero. (mostly tested)
3: Keep the cell fixed in x direction.
4: Keep the cell fixed in y direction.

5: Keep the cell fixed in z direction.
6: Keep the cell fixed in x and y directions.
7: Keep the cell fixed in x and z directions.
8: Keep the cell fixed in y and z directions.

VELR {integer}

Used only in molecular dynamics, **R**escale the initial **vel**ocities of atoms to target temperature.
1: Turn on this function.
0: Turn off this function.

MSDI {integer}

Used only in molecular dynamics. Print out **M**ean **S**quare **D**isplacements beyond this 'integer' step.

MSDA

Used only in molecular dynamics. Print out **M**ean **S**quare **D**isplacement**s** for each **A**tom at each molecular dynamics step.

6.6 Input/Output Parameters

Specify input and output file names and formats. Specify amount of detail to output.

GEOM {filename}

The **GEOM**etry file contains all the information necessary to describe the system studied, including the cell vectors, the types of all atoms present, their positions, and the names of the corresponding pseudopotentials. (Optionally, information on whether each atom should be optimized or held frozen can also be included.) By default, the name of the geometry file is assumed to be the name as the .inpt file, but with an .ion extension (*e.g.*, test.ion). Through the GEOM option, one can change the name of the geometry file to something else (*e.g.*, GEOM test2.txt).

OUTP {filename}

Used to specify the name of the **OUTP**ut file. By default, the name of the output file is assumed to be the name as the .inpt file, but with an .out extension (*e.g.*, test.out).

ERRO {filename}

Used to specify the name of the **ERRO**r file. By default, the name of the output file is assumed to be the name as the .inpt file, but with an .err extension (*e.g.*, test.err).

RHOF {filename}

Short for **RHO** (density) **F**ile. Reads in a density from a file compatible with the current version of PROFESS. The RHOF keyword should be used to specify the name of the density file. Without this keyword, electron density optimizations start from a uniform electron density. 'RHOF' and 'OLDD' should not be used at the same time.

RHOA

If this keyword is used, a superposition of atomic densities specified in the .ion file will be used as initial density.

OLDD or **OLDR**

Short for **OLD D**ensity or **OLD R**ho. Reads in density from a file previously generated by PROFESS. The input density file must share the same name as the .inpt (*e.g.*, test.den). Without this keyword, computation would start from a uniform electron density. This option must not be used together with 'RHOF' (see above).

RHOR {option} {integer}

Short for **RHO R**epeat. This command specifies how many times the density file specified by the 'RHOF/OLDD/OLDR' keyword above should be repeated in a given direction when constructing the starting electron density. It can be useful in some specific cases to generate an initial guess for the electron density of a very large, nearly periodic system. By default, the density file will only be read in once.
X: repeat {integer} number of times in x-direction.
Y: repeat {integer} number of times in y-direction.
Z: repeat {integer} number of times in z-direction.

RHOM {real}

Short for **RHO M**ax. This keyword specifies the maximum size (in MB) of the outputted electron density file. If the full-size electron density array does not fit in the file, a smaller array is generated and output by interpolating points from the full-size array. **(Default=2,000)**

PRIN {option}

This command will determine what you want to **PRIN**t out. By default, an output (.out) file will contain information about the simulation setup and the final energies.
MIN {option} {integer}: Print out various minimizer details to the output file (.out).
RHO or **DEN**: Details about the density optimization.
0: No density optimization report is output. **(Default)**
1: Outputs a header and footer summarizing the electron density optimization.
2: Output from 1, plus a line summarizing each optimization step
3: Output from 2, plus the contribution of each functional to the total energy
ION or **GEO**: Details about the ion optimization.
0: No geometry optimization report is output. **(Default)**
1: Output only the total time for the geometry optimization.
2: Output from 1, plus a line summarizing each optimization step
3: Output from 2, plus the total forces are output at each step.
4: Output from 3, plus the ion-ion and ion-electron components of the forces are also output.

CEL {integer}: Outputs the geometry file every *integer* steps for cell optimization and also outputs the final (relaxed) geometry to .final.ion. By default, the final geometry is output if 'MINI ION' or 'MINI CELL' is chosen.

EWA: Prints details about the Ewald energy calculation in the output each time it is run. By default, the Ewald energy information does not print.

STR: Prints the stress tensor and the lattice vectors in the output file at the end of the computation. Equivalent to 'CALC STR.' By default, stress does not print in .out.

RES: Prints the electron density at each optimization step to a .den file, for ease in restarting in case the calculation fails. Only needed for hard-to-converge structures. By default, this does not print.

RHO or **DEN**: Prints the electron density only at the end of the run. By default, this does not print.

ION or **GEO {integer}**: Outputs the geometry file every 'integer' steps and also outputs the final (relaxed) geometry to .ion.out. By default, the final geometry is output if 'MINI ION' or 'MINI CELL' is chosen.

FOR {option}: Whether to output forces to a separate .force file or output forces to the .out file. This keyword also makes sure that the forces are evaluated and printed out at the end the run.
SEP: Print the forces to a separate file.
SAM: Print the forces to the .out file. **(Default)**

MDI {integer}: Print out MD geometry file and velocity file every 'integer' steps into a directory. The path of the directory is specified by 'MDOP' option as discussed above.

6.7 Other Calculation Parameters

TRAN {option}

Outputs the energies, forces, and stresses necessary for input into the **TRAN**sition state finder CINEB depending on whether option is **ON** or **OFF**. Outputs this information into a file with extension .trans. **(Default is off)**

NNDI {real}

Stands for **N**earest **N**eighbour **Di**stance. If the distance between any two atoms is shorter than the value specified using this keyword, the code will stop. Distance is in Å**.** This can be used to check that the ion configuration does not become unphysical. A negative real number with this keyword indicates that the check is turned off. **(Default is off)**

NNBI {real}

Use binning to compute the minimum nearest neighbor distance of any two atoms. Use in conjunction with 'NNDI.' Bins allow the neighbor distance computation to scale linearly, since if two bins are far from each other, atoms within those bins do not need to be compared to see whether or not they are too close. Binning is especially advantageous for larger systems. A positive number turns binning on; all other numbers turn binning off. **(Default is on)**

7. Troubleshooting

PROBLEM: The compiler cannot link to FFTW3 functions or misses header files.

Make sure a FFTW3 compliant library is properly installed and both the library files, either the static or dynamic ones depending on linking, and the header files are accessible for the compiler.

If the header file is missing, add the directory containing the F95 and/or F2003 header files (depending on serial or parallel compilation) using a -I/path/to/include/directory option for your compiler.

If compilation works but there is a runtime problem due to dynamic linking of the FFTW libraries, you can check which library – if any – you are linking to, by typing:

```
echo $LD_LIBRARY_PATH
```

at the command line. This returns a list of directories, separated by colons. The directory that contains the FFTW3 libraries should be listed before any directories containing other versions of FFTW3. To give the correct directory the highest priority, add the following line to your job submission scripts:

```
LD_LIBRARY_PATH=<yourfftw3directory>:${LD_LIBRARY_PATH}
```

PROBLEM: Small simulations can be run without a problem, but larger ones give a segmentation fault.

This may be related to stack size and overflows. In the bash submission script, make stack sizes on the nodes unlimited by inserting the line, "ulimit –s unlimited" (without the quotes). If using cshell, add the line, "limit stacksize unlimited."

PROBLEM: I cannot reproduce the same results when running on different computers or when using different numbers of processors.

The exact same results should not be expected, since the FFTW library performs the FFTs using different algorithms, depending on the architecture of the computer. Different algorithms may also be used when using various numbers of processors. Generally, final energy results from different computer setups should be considered the same if they match to eight or more significant figures. Please note the above discussed grid size artifacts when using a parallel compilation.

PROBLEM: When using the WGC functional, the energy of the system becomes extremely negative and the potential is huge.

The WGC functional is known to be unstable with densities that deviate very far from the uniform electron density, due to the Taylor series it employs to calculate the kernel. This instability has been observed when there exist regions of very low or very high electron density. The most common sign of this instability is when the energy becomes unphysically negative, while the norm of the potential becomes large.[46] There is as of yet no surefire solution for solving this problem.

However, there are simulation setups that can either exacerbate or help alleviate these issues. First, check that the values for ρ_0 or ρ^*, either calculated automatically during the simulation or input using 'PARA RHO0' and 'PARA RHOS' (see Section 06.4), are close to the average bulk density for the material being studied. If they are too far off, the Taylor series will have problems. Second, check that the ion configuration does not place any two atoms unphysically close to each other. (Keyword 'NNDI' is

helpful for this check.) If they are too close, this will cause extremely uneven electron density, as well. Finally, if those checks do not help, the KEDF WGV is more stable in vacuum, or a different order of the Taylor expansion (keyword 'WGCT') can be used.

8. References

[1] Y. A. Wang and E. A. Carter, "Orbital-Free Kinetic Energy Density Functional Theory," in "Theoretical Methods in Condensed Phase Chemistry," S. D. Schwartz, Ed., within the series "Progress in Theoretical Chemistry and Physics," Kluwer, 117-84 (2000).

[2] G. Ho, V. L. Ligneres, and E. A. Carter, "Introducing PROFESS: A new program for orbital-free density functional theory calculations," *Comput. Phys. Commun.*, **179**, 839 (2008). L. Hung, C. Huang, I.Shin, G. Ho, V. L. Ligneres, and E. A. Carter, "Introducing PROFESS 2.0: A parallelized, fully linear scaling program for orbital-free density functional theory calculations," *Comput. Phys. Comm.*, **181**, 2208 (2010).

[3] L. Hung and E. A. Carter, "Accurate Simulations of Metals at the Mesoscale: Explicit Treatment of 1 Million Atoms with Quantum Mechanics," *Chem. Phys. Lett.*, **475**, 163 (2009).

[4] G. Henkelman, G. Jóhannesson, H. Jónsson, "Methods for finding saddle points and minimum energy paths," Theoretical Methods in Condensed Phase Chemistry (ed Schwartz, S. D.) 269-302 (Kluwer, 2000).

[5] G. Henkelman, B. P. Uberuaga, H. Jónsson, "A climbing image nudged elastic band method for finding saddle points and minimum energy paths," *J. Chem. Phys.*, **113**, 9901-9904 (2000).

[6] J. P. Perdew, K. Burke and M. Ernzerhof, "Generalized gradient approximation made simple," *Phys. Rev. Lett.*, **77**, 3865 (1996).

[7] M. Marques, M. Oliveira, T. Burnus, "Libxc: a library of exchange and correlation functionals for density functional theory," *Comp. Phys. Commu.*, **183**, 2272 (2012).

[8] C. Huang and E. A. Carter, "Nonlocal orbital-free kinetic energy density functional for semiconductors," *Phys. Rev. B*, **81**, 045206 (2010).

[9] C. Huang and E. A. Carter, "Toward an Orbital-free Density Functional Theory of Transition Metals Based on an Electron Density Decomposition," *Phys. Rev. B*, **85**, 045126 (2012).

[10] J. Xia and E. A. Carter, "Density-Decomposed Orbital-Free Density Functional Theory for Covalently Bonded Molecules and Materials," *Phys. Rev. B*, **86**, 235109 (2012).

[11] I. Shin and E. A. Carter, "Enhanced von Weizsäcker Wang-Govind-Carter Kinetic Energy Density Functional for Semiconductors," *J. Chem. Phys.*, **140**, 18A531 (2014).

[12] J. Xia and E. A. Carter, "Single-point Kinetic Energy Density Functionals Based on a Pointwise Kinetic Energy Density Analysis," submitted.

[13] G. Chan, A. Cohen and N. Handy, "Thomas–Fermi–Dirac–von Weizsäcker models in finite systems," *J. Chem. Phys.*, **114**, 631 (2001); A. Berk, "Lower-bound energy functionals and their application to diatomic systems," *Phys. Rev. A*, **28**, 1908 (1983); N. Govind, J. Wang and H. Guo, "Total-energy calculations using a gradient-expanded kinetic-energy functional," *Phys. Rev. B*, **50**, 11175 (1994); K. Yonei, "Energy levels for an extended Thomas-Fermi-Dirac potential," *J. Phys. Soc. Jpn.*, **22**, 1127 (1967); Y. Tomishima and K. Yonei, "Solution of the Thomas-Fermi-Dirac Equation with a Modified Weizsäcker Correction," *J. Phys. Soc. Jpn.*, **21**, 142 (1966); D. A. Kirzhnits, *Sov. Phys. JETP*, **5**, 64 (1957); H. Lee, C. Lee and R. G. Parr, "Conjoint gradient correction to the Hartree-Fock kinetic- and exchange-energy density functionals," *Phys. Rev. A*, **44**, 768 (1991); A. E. Depristo and J. D. Kress, "Kinetic-energy functionals via Padé approximations," *Phys. Rev. A*, **35**, 438 (1987); H. Ou-Yang and M. Levy, "Approximate noninteracting kinetic energy functionals from a nonuniform scaling requirement," *M. Int. J. Quantum Chem.*, **40**, 379 (1991); J. P. Perdew and Y. Wang, "Accurate and simple density functional for the electronic exchange energy: Generalized gradient approximation," *Phys. Rev. B*, **33**, 8800 (1986); D. J. Lacks and R. G. Gordon, "Tests of nonlocal kinetic energy functionals," *J. Chem. Phys.*, **100** 4446 (1994); J. P. Perdew, J. A. Cevary, S. H. Vosko, K. A. Jackson, M. R. Pederson, D. J. Singh and C. Fiolhais, "Atoms, molecules, solids, and surfaces: Applications of the generalized gradient approximation for exchange and correlation," *Phys. Rev. B*, **46**, 6671 (1992); A. Lembarki

and H. Chermette, “Obtaining a gradient-corrected kinetic-energy functional from the Perdew-Wang exchange functional,” *Phys. Rev. A,* **50**, 5328 (1994); F. Tran and T. A. Wesolowski, “Link between the kinetic- and exchange-energy functionals in the generalized gradient approximation,” *Int. J. Quantum Chem*., **89**, 441 (2002); V. V. Karasiev, S. B. Trickey and F. E. Harris, “Born–Oppenheimer Interatomic Forces from Simple, Local Kinetic Energy Density Functionals,” *J. Comput.-Aided Mater. Des.*, **13**, 111 (2006); M. Ernzerhof, “The role of the kinetic energy density in approximations to the exchange energy,” *J. Mol. Struct.: THEOCHEM,* **501**, 59 (2000); J. P. Perdew, “Generalized gradient approximation for the fermion kinetic energy as a functional of the density,” *Phys. Lett. A*, **165**, 79 (1992); A. D. Becke, ”On the large‐gradient behavior of the density functional exchange energy,” *J. Chem. Phys.*, **85**, 7184 (1986); A. D. Becke, ”Density functional calculations of molecular bond energies,” *J. Chem. Phys.*, **84**, 4524 (1986); A. J. Thakkar, “Comparison of kinetic-energy density functionals,” *Phys. Rev. A*, **46**, 6920 (1992).

[14] J. Xia, C. Huang, I. Shin, and E. A. Carter, "Can Orbital-Free Density Functional Theory Simulate Molecules?," *J. Chem. Phys.*, **136**, 084102 (2012).

[15] H. Jiang and W. Yang, “Conjugate-gradient optimization method for orbital-free density functional calculations,” *J. Chem. Phys.* **121**, 2030 (2004).

[16] J. Nocedal, “Updating Quasi-Newton Matrices with Limited Storage,” *Math. Comp.*, **35**, 773-782 (1980). D. C. Liu and J. Nocedal, “On the Limited Memory Method for Large Scale Optimization,” *Mathematical Programming B*, **45**, 503-528 (1989).

[17] G. J. Martyna, M. E. Tuckerman, D. J. Tobias and M. L. Klein, *Mol. Phys.*, **87** (5), 1117 (1996).

[18] M. Chen, L. Hung, C. Huang, J. Xia, and E. A. Carter, “The Melting Point of Lithium: An Orbital-Free First-Principles Molecular Dynamics Study,” *Mol. Phys.*,**111**, 3448 (2013).

[19] L.Hung, C. Huang and E. A. Carter, “Preconditioners and Electron Density Optimization in Orbital-Free Density Functional Theory,” *Comm. Comp. Phys.*,**12**, 135 (2012).

[20] K. J. Caspersen and E. A. Carter, "Finding Transition States for Crystalline Solid-Solid Phase Transformations," *Proc. Natl. Acad. Sci*., **102**, 6738 (2005).

[21] S. Nosé, “A unified formulation of the constant temperature molecular dynamics methods,” *J. Chem. Phys.*, **81**, 511 (1984). W.G. Hoover, “Canonical dynamics: Equilibrium phase-space distributions,” *Phys. Rev. A*, **31**, 1695 (1985).

[22] M.Parrinello and A.Rahman, “Crystal Structure and Pair Potentials: A Molecular-Dynamics Study,” *Phys. Rev. Lett.*, **45**, 1196 (1980).

[23] H. Jónsson, G. Mills, and K. W. Jacobsen, “Classical and Quantum Dynamics in Condensed Phase Simulations,” edited by B. J. Berne, G. Ciccotti, and D. F. Coker World Scientific, Singapore, 1998, pp. 385–404.

[24] W. W. Hager and H. Zhang, “A New Conjugate Gradient Method with Guaranteed Descent and an Efficient Line Search,” *SIAM J. Optim.* **16**, 170-192 (2005).

[25] C. J. García-Cervera, “An efficient real space method for orbital-free density-functional theory,” *Comm. Comp. Phys.*, **2**, 334-357 (2007).

[26] L. H. Thomas, *Proc. Cambridge Philos. Soc.*, **23**, 542 (1927).

[27] E. Fermi, *Z. Phys.*, **48**, 73 (1928).

[28] C. F. von Weizsäcker, *Z. Phys.*, **96**, 431 (1935).

[29] L. Wang and M. P. Teter, “Kinetic-energy functional of the electron density,” *Phys. Rev. B*, **45**, 13196 (1992).

[30] Y. A. Wang, N. Govind and E. A. Carter, "Orbital-Free Kinetic Energy Functionals for the Nearly-Free Electron Gas," *Phys. Rev. B*, **58**, 13465 (1998). Erratum: *Phys. Rev. B*, **64**, 129901-1 (2001).

[31] Y. A. Wang, N. Govind, and E. A. Carter, "Orbital-Free Kinetic-Energy Density Functionals with a Density-Dependent Kernel," *Phys. Rev. B*, **60**, 16350 (1999). Erratum: *Phys. Rev. B*, **64**, 089903-1 (2001).

[32] J. Chai and J. D. Weeks, “Orbital-free density functional theory: Kinetic potentials and ab initio local pseudopotentials,” *Phys. Rev. B*, **75**, 205122 (2007).

[33] J. Chai, V. L. Lignères, G. Ho, E. A. Carter and J. D. Weeks, “Orbital-free density functional theory: Linear scaling methods for kinetic potentials, and applications to solid Al and Si,” *Chem. Phys. Lett.*, **473**, 263 (2009).

[34] E. Chacón, J. E. Alvarellos and P. Tarazona, “Nonlocal kinetic energy functional for nonhomogeneous electron systems,” *Phys. Rev. B.*, **32**, 7868 (1985). D. García-Aldea and J. E. Alvarellos, “Kinetic-energy

density functionals with nonlocal terms with the structure of the Thomas-Fermi functional," *Phys. Rev. A,* **76**, 052504 (2007).

[35] G. Ho, V. L. Ligneres, and E. A. Carter, "Analytic form for a nonlocal kinetic energy functional with a density-dependent kernel for orbital-free density functional theory under periodic and Dirichlet boundary conditions," *Phys. Rev. B*, **78**, 045105 (2008).

[36] I. Shin, A. Ramasubramaniam, C. Huang, L. Hung and E. A. Carter, "Orbital-free density functional theory simulations of dislocations in aluminum," *Philos. Mag.*, **89**, 3195 (2009).

[37] D. M. Ceperley and B. J. Alder, "Ground state of the electron gas by a stochastic method," *Phys. Rev. Lett.*, **45**, 566 (1980).

[38] J. P. Perdew and A. Zunger, "Self-interaction correction to density-functional approximations for many-electron systems," *Phys. Rev. B*, **23**, 5048 (1981).

[39] J. P. Perdew, K. Burke and M. Ernzerhof, "Generalized gradient approximation made simple," *Phys. Rev. Lett.*, **77**, 3865 (1996).

[40] A. Y. Toukmaji and J. A. Board Jr., "Ewald summation techniques in perspective: a survey," *Comput. Phys. Commun.* **95**, 73-92 (1996).

[41] N. Karasawa and W. A. Goddard, III. "Acceleration of convergence for lattice sums," *J. Phys. Chem.* **93**, 7320 (1989).

[42] N. Choly and E. Kaxiras, "Fast method for force computations in electronic structure calculations," *Phys. Rev. B* **67**, 155101 (2003).

[43] U. Essmann, L. Perera, M. L. Berkowitz, T. Darden, H. Lee and L. G. Pedersen, "A smooth particle mesh Ewald method," *J. Chem. Phys.* **103**, 8577 (1995).

[44] H. F. Trotter, "On the product of semi-groups of operators," *Proc. Natl. Acad. Sci*., **10**, 545 (1959).

[45] H. Yoshida, "Construction of higher order symplectic integrators," *Phys.Lett. A*, **150**, 262 (1990).

[46] B. Zhou, V. L. Lignères and E. A. Carter, "Improving the orbital-free density functional theory description of covalent materials," *J. Chem. Phys.*, **122**, 044103 (2005).