Finite-temperature DFT calculations using the Ensemble-DFT method in ONETEP

Álvaro Ruiz Serrano
August 2013

Extended to spin relaxation by Kevin Duff
April 2018

Extension to include Grand Canonical Ensemble by Arihant Bhandari
Sep 2020

Pulay Mixing and Input Trial Step by Gabriel Bramley
November 2020

Contents

1 Basic principles

This manual describes how to run finite-temperature calculations using ONETEP. A recent implementation uses a direct minimisation technique based on the Ensemble-DFT method [2]. The Helmholtz free energy functional is minimised in two nested loops. The inner loop performs a line-search in the space of Hamiltonian matrices (in a similar fashion as described in Ref. [3]), for a fixed set of NGWFs. Then, the outer loop optimises the NGWFs psinc expansion coefficients for a fixed density kernel. For a more detailed description and discussion of this method in ONETEP, see Ref. [4].

Using the NGWF representation, the Helmholtz free energy functional becomes:

$$A_{\mathcal{T}}\left[\{H_{\alpha\beta}\},\{|\phi_\alpha\rangle\}\right] = E\left[\{H_{\alpha\beta}\},\{|\phi_\alpha\rangle\}\right] - \mathcal{T}S\left[\{f_i\}\right]. \tag{1}$$

where $\{H_{\alpha\beta}\}$ is the NGWF representation of the Hamiltonian matrix, $\{|\phi_\alpha\rangle\}$ is the current set of NGWFs, $\mathcal{T}$ is the electronic temperature, $E\left[\{H_{\alpha\beta}\},\{|\phi_\alpha\rangle\}\right]$ is the energy functional and $S\left[\{f_i\}\right]$ is an entropy term. The occupancies of the Kohn-Sham states, $\{f_i\}$ are calculated from the energy levels, $\{\epsilon_i\}$, using the Fermi-Dirac distribution:

$$f_i\left(\epsilon_i\right) = \left(1 + \exp\left[\frac{\epsilon_i - \mu}{k_\mathrm{B}\mathcal{T}}\right]\right)^{-1}. \tag{2}$$

where μ is the Fermi level. To obtain the energy eigenvalues, $\{\epsilon_i\}$, the Hamiltonian matrix is diagonalised as:

$$H_{\alpha\beta}M^\beta{}_i = S_{\alpha\beta}M^\beta{}_i\epsilon_i, \tag{3}$$

where $\left\{S_{\alpha\beta}\right\}$ are the elements of the NGWF overlap matrix, and $\left\{M^\beta{}_i\right\}$ are the expansion coefficients of the Kohn-Sham eigenstates in the NGWF basis set. At the moment, this is a cubic-scaling operation that requires dealing dense matrices, which makes it memory-demanding.

2 Free- and fixed-spin EDFT

By default in spin polarized runs, the total occupancy of each spin channel is held fixed; each spin channel has its own Fermi level determined by this constraint. Alternatively the whole system can be held at one Fermi level dictated by the conservation of the total number of electrons in the system, allowing the net spin to freely relax.

Free-spin EDFT should be appropriate for most applications unless there's a reason to hold the system fixed at a given net spin. As with any minimization with potentially

many minima, the final state may depend on initial conditions. As a special case, free-spin EDFT may not be able to symmetry-break a system that wants to have any kind of spin polarization but that is initialized to have 0 net spin. The general advice for simple systems like basic ferromagnets (though this should not replace good system-specific judgment) is to slightly over-specify the expected net spin on each atom and hold the spin fixed for a few iterations before being allowed to relax. For example a cobalt cluster is expected to have a net spin per atom lower than that of an isolated atom, that decreases to bulk-like as a function of cluster size. A good initialization may be to give each atom atomic-like net spin and hold the net spin fixed for 3-5 NGWF CG iterations, then allow it to relax.

3   Compilation

By default, ONETEP is linked against the LAPACK library [5] for linear algebra. The LAPACK eigensolver DSYGVX [6], can only be executed in one CPU at a time. Therefore, EDFT calculations with LAPACK are limited to small systems (a few tens of atoms). Calculations on large systems are possible if, instead, ONETEP is linked against SCALA-PACK library [7] during compilation time. The SCALAPACK eigensolver, PDSYGVX, can be run in parallel using many CPUs simultaneously. Moreover, SCALAPACK can distribute the storage of dense matrices across many CPUs, thus allowing to increase the total memory allocated to a given calculation in a systematic manner, simply by requesting more processors. For the compilation against SCALAPACK to take effect, the flag -DSCALAPACK must be specified during the compilation of ONETEP.

4   Pulay Mixing EDFT

In default EDFT, the Hamiltonian is updated using a damped fixed point update routine:

$$H_{\alpha\beta}^{(m+1)} = H_{\alpha\beta}^{(m)} + \lambda\, R[H_{\alpha\beta}^{(m)}] \tag{4}$$

Where the λ defines the mixing parameter and residual is defined as:

$$R[H_{\alpha\beta}^{(m)}] = \tilde{H}_{\alpha\beta}^{(m)} - H_{\alpha\beta}^{(m)} \tag{5}$$

Where $\tilde{H}_{\alpha\beta}^{(m)}$ is the diagonlised Hamiltonian obtained at step m. At a sufficiently low value of λ, most systems will achieve convergence, but at an increasingly slow rate as the system increases in size. Convergence can be accelerated using quasi-Newton update methods such as Broyden or Pulay methods, the latter of which is implemented in EDFT as an alternative to the damped fixed point method.

The implementation in ONETEP uses a similar logic to other DFT implementations of Pulay's method, except the Hamiltonian is optimised instead of the density:

$$H_{\alpha\beta}^{(m+1)} = \sum_{j=m-n+1}^{m} c_j H_{\alpha\beta}^{(j)} + \lambda \sum_{j=m-n+1}^{m} c_j R[H_{\alpha\beta}^{(j)}] \tag{6}$$

Where the history length is defined n and the co-efficients c_j are obtained through the procedure outlined by Ref. [10]. For the systems tested, this method leads to improved convergence, especially for larger metallic systems. Further information can be found in Ref. [11].

5   Increased Calculation Speed Using Fixed Step Sizes

As described in Section 4, λ defines the step length taken at each inner loop iteration. In the default algorithm, an optimal λ value which gives the greatest decrease in the Lagrangian is determined by a line search routine. Although this improves the robustness of the algorithm, the line search requires two or more energy evaluations per inner loop step to obtain the optimum λ value. If λ varies very little over the course of the calculation, this can double the computational expense of each inner loop iteration for a negligible increase in the accuracy for each step.

Alternatively, one can fix the λ to a reasonable value for a significant speed-up by ensuring only one energy evaluation is performed per inner loop iteration. However, this option is less robust than the default line search algorithm, as the fixed λ value may produce either sub-optimal energy decreases or energy increases for certain steps. Furthermore, if λ is chosen to be too high, your answer may diverge from the ground state by taking several consecutive positive Lagrangian steps (A warning will be provided if this occurs too often). Conversely, convergence will be very slow if λ is chosen to be too low. λ is set with the `edft_trial_step` keyword, which switches from the line search algorithm if greater than 0, and uses the fixed λ value specified.

User input values of λ can be determined by running a standard EDFT calculation for a single NGWF iteration with line search and plotting the 'step' value printed at each iteration (in VERBOSE output mode). The safest option is to choose a value close to the minimum step value, but a slightly higher value can be selected, especially if larger step values are common. The first two steps of your calculation choose λ with line search regardless of your input, as optimal step sizes for these iterations are significantly larger than subsequent inner loop iterations. As such, these two iterations should be disregarded from your λ value selection analysis. As step sizes which yield stable convergence are system dependent, it is recommended to manually determine different λ values for systems with large differences in species or size.

6   Commands for the inner loop

6.1   Basic setup

- `edft:  T/F`
  [Boolean, default `edft:  F`].
  If true, it enables Ensemble-DFT calculations.

- `edft_maxit:   n`
  [Integer, default `edft_maxit:   10`].
  Number of EDFT iterations in the ONETEP inner loop.

- `edft_smearing_width:   x` *units*.
  [Real physical, default `edft_smearing_width:   0.1 eV`].
  Sets the value of the smearing width, $k_B\mathcal{T}$, of the Fermi-Dirac distribution. It takes units of energy (eV, Hartree) or temperature. For example,
  `edft_smearing_width:   1500 K`
  will set $\mathcal{T} = 1500$ degree Kelvin.

- `edft_update_scheme:   damp_fixpoint/pulay_mix`.
  [Character, default `dft_update_scheme:   damp_fixpoint`].
  Defines the mixing scheme for EDFT in the ONETEP inner loop.

- `edft_ham_diis_size:   x`
  [Integer, default `edft_ham_diis_size:   10`].
  Specifies the maximum number of Hamiltonians used from previous iterations to generate the new guess through Pulay mixing.

- `spin:   x`
  [Real, default `spin:   0.0`].
  For EDFT runs this value does not need to be an integer. Because we are considering an ensemble of states it can have any real value between $-\text{nelec}/2$ to $\text{nelec}/2$. Make sure you have enough bands to cover the more populated spin channel.

- `edft_spin_fix`
  [Integer, default `edft_spin_fix:   -1`].
  Control for whether the net spin of the system should remain fixed at `spin`, or relax during the run. Any negative number will fix the net spin. Nonnegative numbers n will hold the net spin fixed for n iterations then let it relax for the rest of the calculation.

- `edft_trial_step`
  [Integer, default `edft_trial_step:   0`].
  Sets the value of λ, which fixes the step size in the EDFT inner loop, and switches off the line search for optimum λ values. If set to 0, the normal line search routine is used.

6.2   Tolerance thresholds

- `edft_free_energy_thres:   x` *units*.
  [Real physical, default `edft_free_energy_thres:   1.0e-6 Ha/Atom`].
  Maximum difference in the Helmholtz free energy functional per atom between two consecutive iterations.

- `edft_energy_thres:` x *units*.
  [Real physical, default `edft_energy_thres: 1.0e-6 Ha/Atom`].
  Maximum difference in the energy functional per atom between two consecutive iterations.

- `edft_entropy_thres:` x *units*.
  [Real physical, default `edft_entropy_thres: 1.0e-6 Ha/Atom`].
  Maximum difference in the entropy per atom functional between two consecutive iterations.

- `edft_rms_gradient_thres:` x.
  [Real, default `edft_rms_gradient_thres: 1.0e-4`].
  Maximum RMS gradient $\dfrac{d\tilde{A}_{\mathcal{T}}}{df_i}$.

- `edft_commutator_thres:` x *units*.
  [Real physical, default `edft_commutator_thres: 1.0e-5 Hartree`].
  Maximum value of the Hamiltonian-Kernel commutator.

- `edft_fermi_thres:` x *units*.
  [Real physical, default `edft_fermi_thres: 1.0e-3 Hartree`].
  Maximum change in the Fermi energy between two consecutive iterations.

6.3 Advanced setup

- `edft_extra_bands:` n.
  [Integer, default `edft_extra_bands: -1`].
  Number of extra energy bands. The total number of bands is equal to the number of NGWFs plus `edft_extra_bands`. When set to a negative number, no extra bands are added.

- `edft_round_evals:` n.
  [Integer, default `edft_round_evals: -1`].
  When set to a positive integer value, the occupancies that result from the Fermi-Dirac distribution are rounded to n significant figures. This feature can reduce some numerical errors arising from the grid-based representation of the NGWFs.

- `edft_write_occ:` T/F.
  [Boolean, default `edft_write_occ: F`].
  Save fractional occupancies in a file.

- `edft_max_step:` x.
  [Real, default `edft_max_step: 1.0`].
  Maximum step during the EDFT line search.

7   Commands for the outer loop

The standard ONETEP commands for NGWF optimisation apply to the EDFT calculations as well. The only flag that is different is:

- `ngwf_cg_rotate: T/F`. [Integer, default `ngwf_cg_rotate: T`]. This flag is always true in EDFT calculations. It ensures that the eigenvectors $M^{\beta}{}_{i}$ are rotated to the new NGWF representation once these are updated.

8   Restarting an EDFT calculation

- `write_hamiltonian: T/F`.
  [Boolean, default `write_hamiltonian: F`].
  Save the last Hamiltonian matrix on a file.

- `read_hamiltonian: T/F`.
  [Boolean, default `read_hamiltonian: F`].
  Read the Hamiltonian matrix from a file, and continue the calculation from this point.

- `write_tightbox_ngwfs: T/F`.
  [Boolean, default `write_tightbox_ngwfs: T`].
  Save the last NGWFs on a file.

- `read_tightbox_ngwfs: T/F`.
  [Boolean, default `read_tightbox_ngwfs: F`].
  Read the NGWFs from a file and continue the calculation from this point.

  NOTE: if a calculation is intended to be restarted at some point in the future, then run the calculation with
  `write_tightbox_ngwfs: T`
  `write_hamiltonian: T`
  to save the Hamiltonian and the NGWFs on disk. Two new files will be created, with extensions `.ham` and `.tightbox_ngwfs`, respectively. Then, to restart the calculation, set
  `read_tightbox_ngwfs: T`
  `read_hamiltonian: T`
  to tell ONETEP to read the files that were previously saved on disk. Remember to keep a backup of the output of the first run before restarting the calculation.

  NOTE: the density kernel is not necessary to restart an EDFT calculation. However, it is necessary to calculate the electronic properties of the system, once the energy minimisation has completed. To save the density kernel on a file, set:
  `write_denskern: T`
  to generate a `.dkn` file containing the density kernel. To read in the density kernel, set

```
read_denskern:  T
```

9   Controlling the parallel eigensolver

Currently, only the SCALAPACK PDSYGVX parallel eigensolver is available. A complete manual to this routine can be found by following the link in Ref. [8]. If ONETEP is interfaced to SCALAPACK, the following directives can be used:

- `eigensolver_orfac:  x.`
  [Real, default `eigensolver_orfac:  1.0e-4`].
  Precision to which the eigensolver will orthogonalise degenerate Hamiltonian eigenvectors. Set to a negative number to avoid reorthogonalisation with the SCALAPACK eigensolver.

- `eigensolver_abstol:  x.`
  [Real, default `eigensolver_abstol:  1.0e-9`].
  Precision to which the parallel eigensolver will calculate the eigenvalues. Set to a negative number to use SCALAPACK defaults.

The abovementioned directives are useful in calculations where the SCALAPACK eigensolver fails to orthonormalise the eigenvectors. In such cases, the following error will be printed in the input file:

```
(P)DSYGVX in subroutine dense_eigensolve returned info= 2.
```

Many times (although not always) this error might cause the calculation to fail. If this situation occurs, set
```
eigensolver_orfac:  -1
eigensolver_abstol:  -1
```
in the input file and restart the calculation. SCALAPACK will not reorthonormalise the eigenvectors. Instead, an external Löwdin orthonormalisation process [9] will be triggered. This is usually more efficient for larger systems.

10   Grand Canonical Ensemble DFT

In simulations of electrochemical electrodes, the electrons can freely exchange between the electrode and the electrical circuit. So, there is no constraint on the number of electrons N. Rather, the electrode potential U is fixed, with respect to a reference electrochemical potential μ_{ref} which fixes the chemical potential of electrons μ:

$$\mu = \mu_{ref} - eU \tag{7}$$

Typical experiments use a standard hydrogen electrode as the reference electrode with $\mu_{ref}^{SHE} = -4.44$ eV. Once the chemical potential of electrons is fixed, the number of

electrons changes as a dependent variable according to the Fermi-Dirac distribution in eq. (2).

$$N = \sum_i f_i \tag{8}$$

Thermodynamically, this corresponds to switching the electrons from the finite-temperature, fixed-number canonical ensemble to the finite-temperature, fixed-potential grand-canonical ensemble. Correspondingly, the relevant free energy minimized at equilibrium is the grand potential[1]:

$$\Omega = A - \mu N \tag{9}$$

The following keywords are used for the grand-canonical ensemble DFT:

- `edft_grand_canonical:  T/F.`
  [Boolean, default `edft_grand_canonical:  F`].
  Switch to fixed-potential grand-canonical ensemble.

- `edft_reference_potential:  x` *units*.
  [Real physical, default `edft_reference_potential:  -4.44 eV`].
  Set the reference potential μ_{ref}. If no units are given, atomic units are considered: Ha (hartrees).

- `edft_electrode_potential:  x` *units*.
  [Real physical, default `edft_electrode_potential:  0.0 V`].
  Set the electrode potential U. If no units are given, atomic units are considered: Ha/e, hartrees per elementary charge.

- `edft_nelec_thres:  x.`
  [Real, default `edft_nelec_thres:  1.0e-06 per atom`].
  Convergence threshold on the change in number of electrons per spin channel per atom.

References

[1] R. Sundararaman, W. Goddard, T. Arias. Grand canonical electronic density-functional theory: Algorithms and applications to electrochemistry *J. Chem. Phys.*, 146(11):114104, 2017.

[2] N. Marzari, D. Vanderbilt, and M. C. Payne. Ensemble density-functional theory for ab initio molecular dynamics of metals and finite-temperature insulators. *Phys. Rev. Lett.*, 79(7):1337–1340, 1997.

[3] C. Freysoldt, S. Boeck, and J. Neugebauer. Direct minimization technique for metals in density functional theory. *Phys. Rev. B*, 79(24):241103, 2009.

[4] Á. Ruiz-Serrano and C.-K. Skylaris. A variational method for density functional theory calculations on metallic systems with thousands of atoms. *J. Chem. Phys.*, 139(5):054107, 2013.

[5] Lapack. http://www.netlib.org/lapack/.

[6] Lapack DSYGVX eigensolver. http://netlib.org/lapack/double/dsygvx.f.

[7] ScaLapack. http://www.netlib.org/scalapack/.

[8] ScaLapack PDSYGVX eigensolver. http://www.netlib.org/scalapack/double/pdsygvx.f.

[9] Per-Olov Lowdin. On the non-orthogonality problem connected with the use of atomic wave functions in the theory of molecules and crystals. *J. Chem. Phys.*, 18(3):365–375, 1950.

[10] G. Kresse and J. Furthmüller Efficient iterative schemes for *ab initio* total-energy calculations using a plane-wave basis set. *Phys. Rev. B*, 54:11169, 1996.

[11] N. Woods, M. Payne and P. Hasnip Computing the self-consistent field in Kohn–Sham density functional theory *J. Phys.: Condens. Matter*, 31:453001, 2019.