Worksheet 2: Finite-size errors in silicon (90 minutes)

Neil D. Drummond

QMC2015, Kolkata, India
23rd–27th March, 2015

1 Finite-size errors

Expectation values obtained in a finite simulation cell differ from the corresponding values in an infinite crystal. In this worksheet we will illustrate this point and consider some ways in which single-particle finite-size effects and Coulomb finite-size effects in QMC calculations can be dealt with.

Single-particle finite-size errors in QMC are the analogue of $\mathbf{k}$-point sampling errors in DFT calculations. One can therefore assess their magnitude by studying the convergence of the DFT energy with respect to the $\mathbf{k}$-point grid. Moreover, by making a sensible choice of $\mathbf{k}$-vector grid, one can greatly reduce single-particle finite-size errors.

Finite-size errors due to the long-range nature of the Coulomb interaction and the fact that correlations are confined to a single simulation cell can be addressed by extrapolation, the use of the model periodic Coulomb (MPC) interaction and the addition of corrections.

We will apply some of these techniques to calculate the total energy per unit cell of silicon in the diamond structure. You will also learn how to generate and use a blip representation of plane-wave orbitals (and see why this is important).

1.1 Generating blip orbitals

1.1.1 Running a VMC calculation with plane-wave orbitals

Go to the `Silicon222` directory. Two sets of plane-wave orbitals for a 16-atom cell of silicon can be found here: (i) `pwfn.data_Gamma` contains DFT orbitals generated using a $2 \times 2 \times 2$ $\mathbf{k}$-point mesh that includes $\mathbf{k} = \mathbf{0}$ (i.e., the simulation-cell Bloch vector $\mathbf{k}_s$ is at Γ) and (ii) `pwfn.data_L` contains a $2 \times 2 \times 2$ $\mathbf{k}$-point mesh that does not include $\mathbf{k} = \mathbf{0}$ ($\mathbf{k}_s$ is at the L point of the supercell Brillouin zone).

Let's work with the orbitals at Γ first. Copy `pwfn.data_Gamma` to `pwfn.data`.

The `input` file should be set up to run a short VMC calculation using plane-wave orbitals: open the `input` file using your favourite text editor and ensure that **atom_basis_type** is "plane-wave", **runtype** is "vmc", **vmc_nstep** is 200 and **vmc_nconfig_write** is 0. As usual, use CASINOHELP to discover the meaning of any keywords you are unsure about.

Now use RUNQMC to run CASINO. When CASINO has finished, take a note of the total CPU time, which can be found near the end of the `out` file. Use REBLOCK to analyse the data in `vmc.hist` and take a note of the total energy. Then type *clearup*.

1.1.2 Converting plane-wave orbitals to blip orbitals

Using a plane-wave basis to represent the orbitals in a QMC calculations, as we have just done, is very inefficient. Each time an electron moves, every orbital must be evaluated at the electron's new position, and this involves evaluating $\mathcal{O}(N)$ plane waves for each orbital. If the orbitals are represented by B-splines on a real space grid (known as a "blip" representation) then the time taken to evaluate each orbital is independent of system size.

To generate a "blip" representation of the orbitals in `pwfn.data`, type *blip*. You will be asked to enter the "grid multiplicity", which determines the fineness of the spline grid in real space[1]. Choose a multiplicity of 2, which is a fairly typical value. BLIP will then ask you to enter the number of points to be used in a Monte Carlo estimate of the overlap α between the blip and plane-wave representations of the orbitals. Hopefully the overlaps will be close to 1, implying that the numerical values of the blip and plane-wave representations of the orbitals are similar. The overlaps of the Laplacians and the three components of the gradients of the orbitals are also computed. Choose e.g. 1000 points for the test. (If you wish to skip this test, choose 0 points.) BLIP will then ask you if you wish to test the orbitals by evaluating the kinetic energy of each blip orbital numerically. This can take a long time, so answer "N". The kinetic-energy test allows you to compare the kinetic energies of the blip and plane-wave orbitals, which should be very similar if the grid multiplicity is large enough. Finally, BLIP will ask you if you wish to translate atoms for a reduced-periodicity calculation. We are performing a 3D-periodic calculation, so please enter "N".

When BLIP has finished, a `bwfn.data` file containing the blip orbitals should have appeared.

1.1.3 Running a VMC calculation with blip orbitals

To tell CASINO to use the blip orbitals in `bwfn.data`, change the value of the **atom_basis_type** keyword in `input` from "plane-wave" to "blip". Then run CASINO, as you did for the plane-wave orbitals, and use REBLOCK to obtain the energies with their error bars. Do the energies obtained with the plane-wave and blip representations of the orbitals agree? How do the CPU times compare?

When you are convinced that representing plane-wave orbitals in a blip basis is a good idea, type *clearup -all*.[2]

1.1.4 Varying the grid multiplicity (longer activity—optional)

Study the effect of varying the grid multiplicity by producing a graph of VMC energy against the multiplicity. In order for the energy differences to be statistically significant, you will need to increase the number of VMC iterations (**vmc_nstep** in the `input` file). You should find that the VMC energy obtained with blip orbitals converges to the energy obtained with the plane-wave orbitals as the multiplicity gets large.

Can you work out what prevents one from using an arbitrarily large grid multiplicity in practice?

1.2 Optimising a Jastrow factor

We will now try to optimise a Jastrow factor for these orbitals. First, open the `correlation.data` file with a text editor. To get rid of the current Jastrow factor, delete all the lines containing parameter values between "Parameter values ; Optimizable (0=NO; 1=YES)" and "END SET 1" for each of the u, χ and f terms.

Now let's optimise the Jastrow factor. In the `input` file, make sure that **runtype** is set to "vmc_opt" and that **vmc_nconfig_write**=1000 and **vmc_nstep**=5000, meaning that 1000 configurations will be written out for use in the optimisation[3]. A number of different optimisation methods are available in CASINO; use the **opt_method** parameter to select the method that you would like to use[4]. In the first instance, use **opt_method**="varmin_linjas", which is by far the most rapid optimisation method, although it cannot optimise the cutoff lengths in the Jastrow factor. Note that **opt_cycles**=2, meaning that two cycles of VMC configuration generation followed by optimisation will be performed. Make sure that **atom_basis_type** is set to "blip", so that the blip orbitals in `bwfn.data` are used. When you are happy with the `input` file, use RUNQMC to start the CASINO calculation.

Once CASINO has finished, use ENVMC to select the `correlation.out.x` file that corresponds to the lowest energy. Rename this file as `correlation.data`, then type *clearup -all*. Optimisation finished!

[1] If the $\mathbf{G}$ vectors of the plane-wave calculation form an $n \times n \times n$ mesh then the grid of points in real space is an $mn \times mn \times mn$ mesh, where m is the grid multiplicity.

[2] When CASINO reads the blip orbitals in `bwfn.data` it produces binary file(s) called `bwfn.data.b`x holding the blip orbitals. These are more compact and can be read more quickly in subsequent runs; however if `bwfn.data` is updated then one must be careful to delete and regenerate the `bwfn.data.b`x files. By default CLEARUP does not delete the `bwfn.data.b`x files; the "-all" flag tells it to do so.

[3] In general it is best to use at least 10000 configurations, but we are cutting corners here.

[4] Use CASINOHELP to check the meaning of `input` keywords that you are unsure about.

1.3 Careful optimisation (longer activity—optional)

Try to improve the Jastrow factor. You could investigate the effects of increasing the expansion orders (i.e. using more parameters), adding a plane-wave expansion of interparticle distance[5], using more configurations in the optimisation, or using a different optimisation method such as linear least-squares energy minimisation.

It is fairly easy to do a reasonably good job of optimising a wave function, but doing a very good job can be time-consuming.

1.4 VMC calculation

Now set up the `input` file to perform a long VMC calculation. (Set **runtype**="vmc", **vmc_nstep**=20000 and **vmc_nconfig_write**=0.) Then run CASINO using RUNQMC.

When it has finished, use REBLOCK to analyse the VMC data in `vmc.hist`, and take a note of the total energy. Then type *clearup -all*.

1.5 Comparing results at Γ and L

Now replace the orbitals at Γ by the orbitals at L: delete the `bwfn.data` and any `bwfn.data.b*` files, copy `pwfn.data_L` to `pwfn.data` and use BLIP to generate a blip representation of those orbitals. In principle one should re-optimise the Jastrow factor after changing the offset of the $\mathbf{k}$-vector mesh; in practice this is usually unnecessary, however. Use RUNQMC to run CASINO and analyse the results using REBLOCK. Do you get the same energy as you did with the orbitals at Γ?

Given that the DFT energies of the $2 \times 2 \times 2$ $\mathbf{k}$-point grids at Γ and L are -7.74128 and -7.83477 a.u. per primitive cell, respectively, compared with the converged DFT energy of -7.84152 a.u. per primitive cell, it would clearly be best to use L in your calculations.

How does the difference of VMC energies at Γ and L compare with the difference of DFT energies? Hence estimate the size of the single-particle finite-size error in your VMC data at L.

2 Model periodic Coulomb interaction

2.1 Generating the input file for the MPC interaction

The MPC interaction can only be evaluated if the Fourier transformations of the charge density and $1/r$ within the minimum image convention are known. In order to generate an `mpc.data` file holding these data, set **runtype** to "gen_mpc" in `input`, and then run CASINO using RUNQMC. Have a look at the `out` file to make sure that everything has worked OK, then type *clearup -all*. An `mpc.data` file should have appeared in the directory.

2.2 VMC data with the MPC interaction

Now calculate the VMC energy using the MPC interaction. Set the **interaction** parameter to "ewald_mpc" instead of "ewald". Perform reblocking analysis on the data in `vmc.hist`, and note down the MPC energy obtained.

3 Extrapolation to infinite system size

3.1 Calculating the VMC energy of a 54-atom cell

Input files for a 54-atom cell at L can be found in the `Silicon333` directory. The Jastrow factor in the `correlation.data` file has been optimised, so you just need to (i) generate blip orbitals, (ii) generate the `mpc.data` file, (iii) run a VMC calculation and (iv) analyse the results using REBLOCK. Depending on which laptop you are using, you may have to cut corners with the optimisation.

[5] To generate a plane-wave expansion p in electron–electron separation, type *make_p_stars*. It will ask you for the number of dimensions (3), the lattice (FCC) and the number of stars (start off with a small number, e.g. 4). Then copy and paste the p term into the Jastrow factor in `correlation.data`, e.g. between "END F TERM" and "END JASTROW".

3.2 Infinite-system limit

The finite-size errors in the Ewald and MPC energies per primitive cell fall off as $1/N$, i.e. $E_\infty = E_N + b/N$ where b is a constant and E_N is the energy per unit cell in an N-electron simulation cell. Use this expression to calculate E_∞ using the Ewald results and the MPC results. Hopefully you get the same answer...

Plot E_N against $1/N$ for both the Ewald interaction and the MPC interaction. You should find that the finite-size errors with the MPC are smaller than the finite-size errors obtained with the Ewald interaction.