Worksheet 2: Finite-size errors in silicon (1 hour)

Neil Drummond and Pablo López Ríos

2007 Summer School on Computational Materials Science
University of Illinois at Urbana-Champaign, USA
Friday 13th July, 2007

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-ranged 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. We will also learn how to perform wave-function optimisation using CASINO.

First optimisation calculation

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 L). Note that twisted boundary conditions must be applied in case (ii), although the orbitals can still be chosen to be real.

First, copy `pwfn.data_Gamma` to `pwfn.data`, and use BLIP to generate a `bwfn.data` file containing the orbitals represented by blips (see Worksheet 1 if you're not sure how to do this).

We will now try to optimise a Jastrow factor for these orbitals. Open the `correlation.data` file with your favourite text editor and have a look at the Jastrow factor:

- Three terms are present: an electron–electron term u, an electron–nucleus term χ and an electron–electron–nucleus term f.
- The form of the Jastrow factor is given in the CASINO manual. Basically each term is a polynomial function of the interparticle distances, and the polynomial expansion coefficients are optimisable parameters.
- The number of terms in the polynomials is controlled by the "expansion orders".

- The Kato cusp conditions are imposed, determining some of the parameter values. Only the free parameter values are shown in `correlation.data`.
- Each term goes smoothly to 0 at a cutoff length, which is itself an optimisable parameter.
- All free parameters in `correlation.data` are followed by a flag specifying whether they are fixed or optimisable.
- Different parameters can be used for different combinations of spins. The behaviour is controlled by the spin-dependence flag. For example, for the u term, if this flag is 0 then the same u term is used between all pairs of electrons; if it is 1 then different u terms are used for parallel- and antiparallel-spin electrons; if it is 2 then different u terms are used for both-up, both-down and opposite-spin electrons.

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 **nwrcon**=**nmove**=1000, meaning that 1000 configurations will be generated by VMC and written out for use in the optimisation[1]. The **corper** parameter should be given a large value, e.g. 10, to ensure that the VMC-generated configurations are not serially correlated. 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[2]. "varmin_linjas" is by far the most rapid optimisation method, although it cannot optimise the cutoff lengths in the Jastrow factor. Note that **opt_cycles**=3, meaning that three cycles of VMC configuration generation followed by optimisation will be performed. When you are happy with the `input` file, use RUNQMC to start the CASINO calculation.

At each cycle of the optimisation, CASINO will write out a `correlation.out.x` file, which holds the Jastrow factor produced in that cycle. To examine the results of the optimisation, type *ve*; this presents you with a summary of the energy and variance obtained in each VMC configuration-generation calculation, and the name of the file containing the corresponding Jastrow factor. (You can make use of VE while CASINO is still running.) The energy and variance usually decrease sharply in the first cycle; thereafter they remain relatively stable.

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

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[3], 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] In general it is best to use at least 10000 configurations.

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

[3] 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".

VMC calculation

Now set up the `input` file to perform a long VMC calculation. (Set **runtype**="vmc", **nwrcon**=0, **corper**=4 and **nmove**=5000.) 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*.

Comparing results at Γ and L

Now repeat all of the above steps[4] for the plane-wave data in `pwfn.data_L`. Do you get the same energy?

Given that the DFT energies of the $2\times2\times2$ $\mathbf{k}$-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? Estimate the size of the single-particle finite-size error in your data.

Model periodic Coulomb interaction

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*. An `mpc.data` file should have appeared in the directory.

VMC data with the MPC

Now calculate the VMC energy using the MPC interaction. Set the **interaction** parameter to "mpc" instead of "ewald"[5]. Perform reblocking analysis on the data in `vmc.hist`, and note down the energy obtained.

Extrapolation to infinite system size

Calculating the VMC energy of a 54-atom cell

Input files for a 54-atom cell 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.

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.

[4] Instead of deleting the parameters in the Jastrow factor before re-optimising it, you could just start off with the parameters you optimised at Γ. In fact you should find that the Jastrow factor you optimised at Γ is perfectly good at L.

[5] For later reference, if you want to calculate the MPC and Ewald energies at the same time, set **interaction** to "ewald_mpc".

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.