compmatphys.org : project topic 2 (2022-2023)

When doing DFT calculations, we want to keep numerical noise under control. Only when a predicted value – e.g. the total energy of a crystal – is up to a known level free of numerical noise, it makes sense to compare it to experiments or to more advanced calculations. That is why you do convergence testing: if the size of the basis set or the density of the k-mesh would not be sufficiently large, you get predictions with random numerical noise on top of it. That is also why code comparison studies are done (e.g. https://doi.org/10.1126/science.aad3000): if multiple independent DFT codes predict the same value for a given quantity (within some error bar), you can trust that the numerical noise is at most of the level of the differences among these codes.

One aspect of numerical noise that has not systematically been investigated for DFT codes so far, is the impact of the choice of the unit cell on numerical noise. There is an infinite number of choices for a primitive unit cell of a given crystal, and on top of that you can represent the same crystal also by an infinite number of non-primitive cells (supercells). If all of these cells represent the same crystal, then all of them should lead to the same value for, for instance, the total energy. However, the extent to which this is true or not, has never been systematically studied. Is it really so that choosing a weird primitive cell with very small or very large angles leads to the same energy as the standard choice? If the answer to this question would be 'no', then the numerical uncertainty on DFT predictions may be larger than currently assumed.

Well, let's find out…!

Task 1 : basic convergence testing

You'll examine one of the simplest metals, fcc-Al. It is a case that is often used for testing purposes. Search a cif file (an experimental one or a calculated one) with a lattice parameter close to the optimal one (it does not need to be spot on, within 5% is OK). Make sure you have a primitive cif file, with just one atom. Do a convergence study for the basis set size and k-mesh, to inspect how the total energy behaves with respect to these two convergence parameters. What is the remaining numerical noise on this calculation?

Note: for later convenience, you better specify the k-mesh quality in a lattice-independent way (hints how to do so will be added here later)

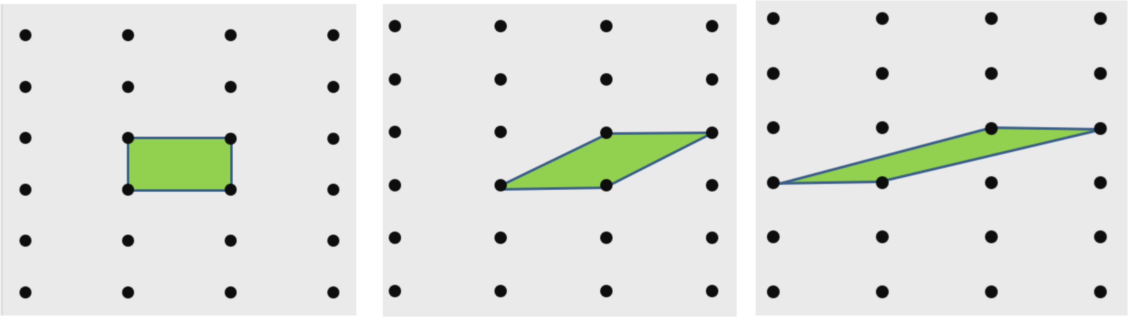
Task 2 : E(V) scan and EOS fit

With a good and efficient basis set and k-mesh determined in the previous task, do a E(V) scan of this crystal. Determine the equilibrium volume, bulk modulus and its pressure derivative by a Birch-Murnaghan fit. Make a cif file for the equilibrium case – this is the crystal volume with which you will continue.

Task 3 : constructing alternative unit cells

With the hints/links/info give here, familiarize yourself with the procedure to generate a cif file for any choice of primitive cell:

This series of 3 pictures show a rectangular bravais lattice, with three choices of primitive cells. You can convince yourself that each cell represents 1 bravais lattice point (that's why it is called primitive), and that the area of the 3 cells is identical.



The CELL_PARAMETERS block in the Quantum Espresso input file is a matrix that contains as its columns the 3 components of each of the 3 vecotrs that span the unit cell (in alat units). This would be the values for a cubic cell that is aligned with the axes of the axes system:

```
CELL_PARAMETERS {alat}
  1.000000000000000   0.000000000000000   0.000000000000000
  0.000000000000000   1.000000000000000   0.000000000000000
  0.000000000000000   0.000000000000000   1.000000000000000
```

Figure out a procedure by which you can convert any choice of primitive cell into the proper input matrix (note: any choice of primitive cell will contain 1 aluminum atom only, and that atom can stay at position (0, 0, 0). The CELL_PARAMETER block is the only thing that needs to be changed. You can visualize your input files by xcrysden, to see whether they do indeed represent the cell you had in mind, and whether this still leads to a proper fcc-Al crystal.

Task 4 : numerical noise for primitive unit cell choices

Calculate the total energy for the equilibrium volume, for a list of primitive cells. Try to build systematic series of primitive cells, like the series of which the start is illustrated in the 3 pictures under task 3. Don't be afraid to go to extreme cases, where some of the cell angles become very small. Calculate the total energy for each cell, keeping strictly the same basis set size and same k-mesh density (i.e. the optimal ones you determined before). Inspect what happens to the total energy: does it stay constant? Does it vary? If it varies, how much? Or does it start varying only after a certain point in your series? …

Task 5 : numerical noise for supercells

Make a few supercells, like the conventional one (4 atoms), 2x1x1 primitive cells (2 atoms), 2x2x1 primitive cells (4 atoms), 2x2x2 primitive cells (8 atoms), 2x1x1 conventional cells (8 atoms), … and inspect again the total energy (per atom) : does it stay constant? Does it vary? If it varies, how much?

Task 6 : combining supercells and alternative cell choises

Depending on the outcomes of tasks 4 and 5, it could be useful to inspect for a few of the supercells what happens if you construct a supercell of the same size with a different choice of lattice vectors (as in task 4).

Task 7 : formulating conclusions

Formulate general conclusions on the behaviour you observed. How large do you estimate the numerical noise due to unit cell choice to be?

Some additional hints:

If your calculation crashes, complaining about not being able to allocate sufficient memory, then this is what you can do :

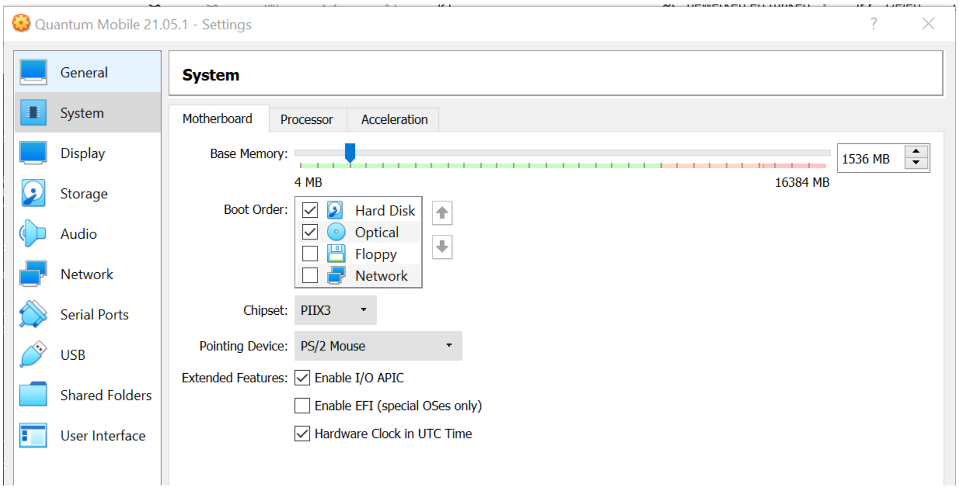
If you open the output file by a text editor (nano, full command is “nano name.out”), you’ll see somewhere in the initial part an estimation of the memory that will be used. Does that make sense for your laptop or not?

You can play with the k-mesh (go down to as small as 1x1x1) and basis set size (encut, even a small value of 5 or 10 Ry is useful to see whether the calculation can run), and inspect how that affects the predicted memory use.

The virtual machine has a built-in max memory use, in order not to crash your laptop. You can change this by clicking the gear symbol before launching Quantum Mobile, then system/motherboard, and use the slider to increase the amount of memory it is allowed to use (compare that with the estimate from the out file, and you’ll see whether or not you have a chance it will run). See screenshot underneath.

If you select the ‘processor’ tab next to ‘motherboard’, you can use more cpu’s of your laptop. You can test on a smaller case whether or not this has any effect. (I did not try this with Quantum Mobile yet, it may be you’ll need to launch the calculation via “mpirun -np 4 pw.x ….” in order to use 4 cpus, or a different number.)

If you cannot get anything running, or in case you hit the ceiling for a given basis set size or kmesh, consider continuing on the HPC (access instructions available in the “let’s play” tile, first week)



Base Memory: 1536 MB

Boot Order:
☑ Hard Disk
☑ Optical
☐ Floppy
☐ Network

Chipset: PIIX3

Pointing Device: PS/2 Mouse

Extended Features:
☑ Enable I/O APIC
☐ Enable EFI (special OSes only)
☑ Hardware Clock in UTC Time

How to install the pseudopotential library once and for all

Go to https://www.materialscloud.org/discover/sssp/table/efficiency and click the 'Pseudo' button to download the library. It will download a tar.gz file with a long name, here abbreviated as SSSP.tar.gz.

Bring this file via the shared folder to Quantum Mobile, and copy it from there to the top folder where you have your subfolders with calculations. Unzip and unfold the file there by the commands:

```
gunzip SSSP.tar.gz
```

```
tar -xvf SSSP.tar
```

Now you'll have a folder with one pseudopotential file per element. Take note of the name of the files for the elements you need (UPF files).

In your calculation folder, where you have your Quantum Espresso input file, set the pseudo_dir variable to the location of this pseudopotential folder, for instance:

```
pseudo_dir='/home/max/work/my-pseudo-library',
```

Now you can list under the ATOMIC_SPECIES block the name of the UPF files you have take note of.