TCHInt tutorial

Pablo López Ríos, 18 April 2023.

Brief introduction to the command-line

If you are unfamilar with the UNIX command line, here is a little crash course of the things you will need. Throughout the tutorial pay close attention to command details (*e.g.*, the presence or absence of spaces) and feel free to ask for further help if you need to.

The command line and logging in:

- At the command line you can type a shell command and its arguments, and press `Enter` to run it, *i.e.*, `command arguments` `Enter`, *e.g.*, `sbatch jobscript` or `./run.sh`.
- To access the Dragon HPC cluster, you will need to open the terminal application on your laptop and type `ssh -X username@dragon.phys.p.lodz.pl`.

Directories (folders) and files:

- The shell tracks a *working directory*, which at the start of the shell session is your personal home directory. One "goes into" a different directory with `cd directory`, *e.g.*, `cd ~/tutorial` or `cd test2`, and `cd ..` to "leave" a directory.
- The contents of the current working directory are shown with `ls`.
- Directories can be created with `mkdir directory` and deleted with `rmdir directory` (provided they are empty).
- Files can be copied with `cp existing-file-name new-file-name`, moved (or renamed) with `mv old-file-name new-file-name`, and deleted with `rm file`,

The NANO editor:

If you have no preferred terminal-based editor, you may use `nano` to view and edit files:

- To open a file, type `nano file`.
- Edit the file, noting that the cursor is moved using the ← ↑ ↓ → keys.
- To save the file, hold `Ctrl` and press `O`.
- To exit the editor, hold `Ctrl` and press `X` (followed by `Y` or `N` to write unsaved changes or not, respectively, if prompted).

1 Activity: transcorrelated integral generation with xTC

In this activity we will run a quick TCHINT calculation, after which you are encouraged to play around with the input and re-run. TCHINT has been compiled and set up for you, and we will use the local Dragon HPC cluster to run the calculation.

In this tutorial we will produce transcorrelated integral files for the all-electron N_2 molecule with the cc-pVDZ basis set. An `input.molden` file has been pre-generated using PYSCF, and a Jastrow factor has been optimized with variance minimization using CASINO. We will use the mean-field "xTC" approximation, so TCHINT will produce an `FCIDUMP` but no `TCDUMP` file, and we shall use the coarsest integration grid setting of `"grid_lvl": 0` for speed. Note that subsequent NECI calculations will use separately-obtained `FCIDUMP`s using a finer `"grid_lvl": 2`; you will not need to use the output files from this tutorial in later tutorials (although you could).

- Copy the tutorial input files to a working directory by typing

  ```
  mkdir ~/tchint-tutorial
  cd ~/tchint-tutorial
  cp ~ws-p.lopez/tutorial/* .
  ```

  and then:

  - Have a look at the directory contents with `ls`.
  - Have a look at the TCHINT input file with `nano tchint.json`.
  - Have a look at the orbital data file with `nano input.molden`.
  - Have a look at the Jastrow factor definition and parameters with `nano parameters.casl`.
  - Have a look at the batch submission script with `nano jobscript`.

- Run the job with `sbatch jobscript`. This might take a bit to start, so do `ls` every few seconds until you see an `out` file appear.

  - Have a look at the output file with `nano out`, locate the system parameters (number of electrons, orbitals, grid points, ...), the non-TC and TC reference energy, and the time taken by this run.
  - (Optional) Have a look at the `FCIDUMP` and `FCIDUMP.bare` files and see if you can spot the expected similarities and differences.

- (Optional) You may run TCHINT again by setting up the input files in a separate directory:

  ```
  mkdir ~/tchint-tutorial-2
  cd ~/tchint-tutorial-2
  cp ~ws-p.lopez/tutorial/* .
  ```

  and so on. Try some of the following:

  - Try setting `"grid_lvl": 1` in `tchint.json`
    - Compare the reported runtime with that of the original run.
    - Compare the reported reference energy with that in the original run.
  - Try setting `"lmat_mean_field": false` in `tchint.json`
    - Compare the runtime with the that of the mean-field run.
    - Compare the number of matrix elements in the `FCIDUMP` (approximately the number of lines in it) with that in the `TCDUMP` (reported in the output file).
    - Compare the reported reference energy with that in the mean-field run.