Instructions for Magic Castle HPC Users

Key Information

Magic Castle implements an HPC cluster that is very similar to the general purpose clusters of the Digital Research Alliance of Canada.
The Magic Castle instance for the Toronto Bioinformatics Hackathon is called **deep-lionfish**.

- 1 login nodes without GPU and 2 compute nodes with 1 GPU each.
- SSH login: **deep-lionfish.c3.ca**
- Jupyter Hub: **https://jupyter.deep-lionfish.c3.ca**
- Documentation: **https://docs.alliancecan.ca** (most of which applies).
- Support: **Email courses@scinet.utoronto.ca**
- Your username:
- Your password:
- Instruction video: https://scinet.courses/1374
- Each account has its own home, scratch, and project directories, visible on all nodes.
- Cluster life time: **until October 1st, 2024.**

The `deep-lionfish` cluster runs on Arbutus, a national academic cloud service hosted at the University of Victoria. It was set up by SciNet, another partner and hosting site of the Alliance. Magic Castle was developed primarily by members of Calcul Québec, also an Alliance partner.

Terminal access

- Open a terminal/ssh client.
- Type "`ssh USERXX@deep-lionfish.c3.ca`" (not an email address!)
- Type the password.
- You are now on a **login node**, which you share it with all other users.
- You can edit files and do small tests on this.
- From here you can request dedicated resources.
- To copy data to `deep-lionfish.c3.ca` from a local terminal, use `scp` or `rsync`.

Jupyter Hub access

- Alternatively, the **Jupyter Hub** is a way to interact with the cluster from your browser.
- Visit the URL: `https://jupyter.deep-lionfish.c3.ca`
- Log in with your username and password. Leave the OTP field empty.
- Next, ask for dedicated resources.
- Use the Jupyter Lab environment for all cases except maybe VS code.
  The Jupyter Lab supports notebooks, uploads and downloads and (also) a terminal.

Compute resources

- Real work does must be done on **compute nodes**.
- Compute nodes are administered by a scheduler called SLURM.
- You get to compute nodes via the Jupyter Hub, or the `salloc` or `sbatch` commands.
- You must specify the resources you want the scheduler to give you, in particular
    - The number of cpu cores (for sbatch/salloc: `-n` followed by a number)
    - The number of GPUs (for sbatch/salloc: `-G` followed by a number)
    - The amount of (host) memory (for sbatch/salloc: `--mem=...`)
    - The duration of your resource request (for sbatch/salloc: `--time=...`)

Software:

- CC software (https://docs.alliancecan.ca/wiki/Available_software) is available, but excluding commercial software.
- For Python packages, start from one of the Python modules, create virtual environments in your $HOME directory, and install packages with `pip`.
- Use of Anaconda and its variants is highly discouraged.
- You can install R packages in your own $HOME.
- If you plan to install software binaries, follow the steps on https://docs.alliancecan.ca/wiki/Installing_software_in_your_home_directory

Running Python notebooks from the Jupyter Lab

- This environment has several “launchers”. One of this is plain Python.
- Inside a virtual environment that you want to show up as a launcher, execute this:
  ```
  $ pip install ipykernel
  $ python -m ipykernel install --user --name ENV --display-name ENV
  ```
- You may need to refresh the Jupterhub to see the launcher called “ENV” appear.

Running R notebooks from the Jupyter Lab

- In a terminal, load one of the R modules (e.g. “module r/4.3.1”)
- Start R, and in R, execute the following commands:
  ```
  > install.packages(“IRkernel”)
  > IRkernel::installspec()
  ```
- You may need to refresh the Jupterhub to see the launcher called “R” appear.

Running Rstudio from the Jupyter Lab

- Go to the ‘softwares’ section, find `rstudio-server/4.2.1` and click ‘load’.
- Now click on the Rstudio Launcher (and be a bit patient).