Running Density Functional Theory Calculations Using Orca on the C1 Cluster

Toby McConville

January 16, 2026

Connecting to the C1 Cluster

With MacOS or Linux, open the terminal.
On windows, use Windows Linux Subsystem to connect.
Run the command:
ssh-keygen -t rsa -b 4096 -f /.ssh/id_rsa
chmod 700 /.ssh
chmod 600 /.ssh/id_rsa
chmod 644 /.ssh/id_rsa.pub

UWaterloo Authentication

Before using your key on C1, you must register it at:

authman.uwaterloo.ca

Follow these steps:

1. Go to authman.uwaterloo.ca
2. Log in with your WatIAM credentials
3. Navigate to SSH key management
4. Copy contents of `id_rsa.pub` and paste
5. Save your registration

View Your Public Key `cat ~/.ssh/id_rsa.pub`

Adding Your SSH Key to C1

Automatic Method

```
ssh-copy-id -i ~/.ssh/id_rsa.pub username@c1.sci.uwaterloo.ca
```

- You'll be asked for your password **once**
- Your public key will be added to `~/.ssh/authorized_keys` on C1

Manual Method (If ssh-copy-id unavailable)

```
cat ~/.ssh/id_rsa.pub | ssh username@c1.sci.uwaterloo.ca
'mkdir -p ~/.ssh && cat >> ~/.ssh/authorized_keys'
```

Test your connection:

```
ssh username@c1.sci.uwaterloo.ca
```

SSH Config: Create Connection Shortcuts

Why use SSH config?

- Type `ssh c1cluster` instead of `ssh username@c1.sci.uwaterloo.ca`
- Store settings for multiple servers
- No need to remember full hostnames

Create/edit config file: `nano ~/.ssh/config` **Add this configuration:**

```
Host c1cluster
    HostName c1.sci.uwaterloo.ca
    User your_username
    IdentityFile ~/.ssh/id_rsa
```

Set correct permissions: `chmod 600 ~/.ssh/config`

Now connect with: `ssh c1cluster`

Useful Unix Commands for File Management

Command	Description
`pwd`	Print working directory
`ls`	List all files in the current directory
`ls *.inp`	List all files with extension `.inp` in the present directory
`ls -l`	List all files in current directory with detailed information
`cp file1 file2`	Copy a file with name "file1" to a file with name "file2"
`cp -r dir1 dir2`	Copy a directory with name "dir1" to a directory with name "dir2"
`mkdir calc`	Make the directory "calc"
`cd calc`	Change into the directory calc
`cd ..`	Change to a lower directory level
`rm *.tmp`	Remove all files with extension `.tmp` (CAREFUL!)
`mv file1 file2`	Rename the file "file1" to "file2"
`find . -name "*.inp"`	Find all files in all subdirectories with extension `.inp`

Useful Unix Commands for dealing with the contents of text files

Command	Description
`cat file`	Print contents of the file (without pausing on screen boundaries)
`more file`	Print contents of the file pausing when screen end is reached. Pressing RETURN scrolls down one line, SPACE on screen
`grep -i "string" file`	Search in file for "string" and print all lines containing this string
`sed "s/string1/string2" file`	Search in file for all occurrences of string1 and replace them with string2 and print the result on the screen

Viewing Files with `less`

Usage: `less filename` (e.g., `less HY_opt.out`)

Command	Description
Navigation	
`Space` or `f`	Move forward one page
`b`	Move backward one page
`Down arrow`	Move forward one line
`Up arrow`	Move backward one line
`g`	Go to beginning of file
`shift g`	Go to end of file
Searching	
`/pattern`	Search forward for "pattern"
`?pattern`	Search backward for "pattern"
`n`	Go to next search match
`N`	Go to previous search match
Other	
`q`	Quit/exit less
`h`	Display help

Tip: Try `/FINAL` to jump to final energy in an ORCA output file

First Connection to the Cluster

Connect using your SSH key: `ssh c1cluster` **You should see:**

- Welcome message
- Cluster information
- Your home directory location

Verify your location:

```
pwd             # Print working directory
ls -la          # List all files including hidden
df -h .         # Check available disk space
```

Setting Up ORCA Environment

ORCA is already installed on the cluster at:
`/scic/app/cpu/orca/6.0.0` **Add ORCA to your PATH (temporary):**

```
export PATH=/scic/app/cpu/orca/6.0.0:$PATH
```

Verify ORCA is now accessible:

```
which orca            # Should show: /scic/app/cpu/orca/6.0.0/orca
```

Organizing Your Work

Create a directory structure:

```
cd ~
mkdir -p Lab_BasisSet/HY
cd Lab_BasisSet/HY
```

Recommended structure:

```
Lab_BasisSet/
 HY/
    HY_pc0.inp
    HY_pc0.out
 (we'll compare to reference data for other basis sets)
```

Example of an ORCA Input File - With water

Use nano to create an input file:

```
nano h2o.inp
```

Simple water molecule example:

```
# Water molecule optimization
! B3LYP SVP Opt TightSCF
* xyz 0 1
 O    0.000000    0.000000    0.117790
 H    0.000000    0.755453   -0.471161
 H    0.000000   -0.755453   -0.471161
*
```

Save with `Ctrl+O`, exit with `Ctrl+X` (nano)

Running ORCA

Batch submission:
Create a submission script `submit.sh`:

```
#!/bin/bash
#SBATCH --job-name=h2o_opt
#SBATCH --time=01:00:00
#SBATCH --ntasks=1
#SBATCH --mem-per-cpu=4000
module load orca/6.0.0
orca h2o.inp > h2o.out
```

Submit the job:

```
sbatch submit.sh
```

Note: Type squeue in the terminal to look at your queued jobs

Understanding ORCA Output

Sections to look for:

1. **Input Echo** - Confirms your input
2. **Geometry** - Initial and final structures
3. **SCF Convergence** - Electronic structure iterations
4. **Final Single Point Energy** - Total energy
5. **Optimization** - Geometry changes (if Opt keyword)
6. **Timings** - Computational cost

Checking for success:

```
grep "HURRAY" h2o.out
# or
grep "SUCCESS" h2o.out
```

Transferring Files: Cluster Local Computer

Download files from cluster:

```
# From your local computer:
scp c1cluster:~/Lab_BasisSet/HY/*.out ./
```

Upload files to cluster:

```
# From your local computer:
scp myfile.inp c1cluster:~/Lab_BasisSet/HY/
```

For entire directories:

```
scp -r c1cluster:~/Lab_BasisSet ./local_folder/
```

Important Files Generated by ORCA

File	Description
`.out`	Main output file with results
`.gbw`	Binary wavefunction file
`.xyz`	Final geometry in XYZ format
`.prop`	Property file (MOs, densities)
`.hess`	Hessian matrix (from NumFreq)
`_trj.xyz`	Optimization trajectory
`.tmp`	Temporary files (can be deleted)

Cleanup temporary files:

```
rm *.tmp
```

Best Practices

1. **Always use descriptive filenames**
   - Good: `HY_pc0_opt.inp`
   - Bad: `test1.inp`
2. **Keep a calculation log**

   ```
   echo "$(date): Started HY optimization" >> logfile.txt
   ```

3. **Test with small basis sets first**
   - pc-0 for quick testing $\rightarrow$ larger basis for final results
4. **Archive completed calculations**

   ```
   tar -czf Lab_results.tar.gz Lab_BasisSet/
   ```

5. **Backup important results locally**

What is a Basis Set?

Think of basis sets like the resolution of a camera:

- Larger basis set = more "pixels" to describe electrons
- More accurate, but computationally slower

The pc-series basis sets:

Basis Set	Size	Speed	Quality
pc-0	Small	Very Fast	Initial testing
pc-1	Medium	Fast	Good geometries
pc-2	Large	Moderate	High accuracy
pc-3	Very Large	Slow	Near-perfect

Results should converge as basis sets get larger

- Bond lengths stop changing significantly
- Energies stabilize
- We look for when results "stop improving"

Today's Exercise: Learning from a "Bad" Calculation

Strategy: Run ONE calculation with small basis set, compare to converged results **Workflow:**

1. Build HY and PY molecules in Avogadro
2. Create ORCA input files with pc-0 basis set
3. Run YOUR calculations (10 min)

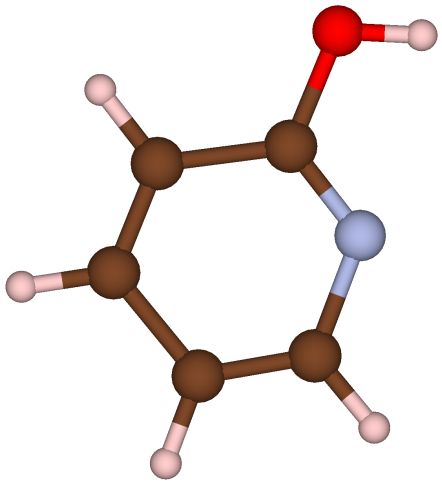
Learning Goals:

- Experience computational workflow hands-on
- See dramatically wrong predictions from small basis sets
- Compare to experimental literature values

The Two Tautomers

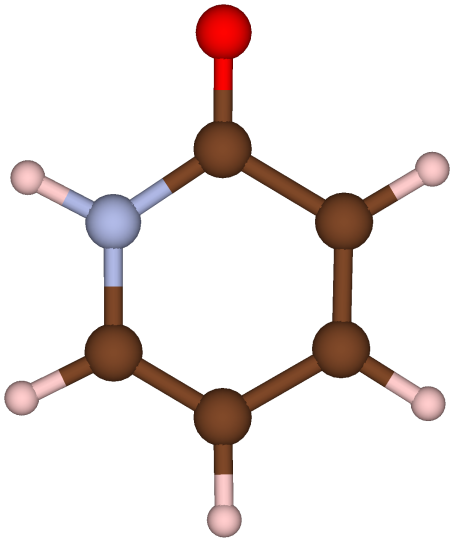
2-Hydroxypyridine (HY)

- Enol form
- O-H group
- Aromatic N



2-Pyridone (PY)

- Keto form
- C=O group
- N-H group



The Question: Which form is more stable?

Step 1: Build Molecules in Avogadro

Launch Avogadro on your local computer Build HY (2-Hydroxypyridine):

1. Draw Tool → Draw benzene ring
2. Click on one C → Change to N (periodic table button)
3. Click on C next to N → Draw O
4. Click on O → Add H
5. Add H to remaining carbons
6. Extensions → Optimize Geometry (Force Field: UFF)

Save coordinates:

- File → Save As → `HY.xyz`

Build PY the same way:

- Start with pyridine ring (N in ring)
- Make C=O double bond (next to N)
- Add H to N
- Optimize and save as `PY.xyz`

Step 2: Transfer XYZ Files to Cluster

From your local computer, upload to cluster:

```
scp HY.xyz $USER@c1cluster:~/Lab_BasisSet/HY/
scp PY.xyz $USER@c1cluster:~/Lab_BasisSet/HY/
```

Or: Copy-paste method

1. Open `HY.xyz` in a text editor locally
2. Copy the coordinates
3. SSH to cluster
4. Create file with nano and paste

Verify files are there:

```
ssh c1cluster
cd ~/Lab_BasisSet/HY
ls -l
```

Step 3: Create ORCA Input File for HY

Check your XYZ file exists:

```
ls HY.xyz
```

Create ORCA input file:

```
nano HY.inp
```

Input file contents:

```
! B3LYP pc-0 TightOPT
! Opt Freq
%maxcore 2000
xyzfile 0 1 HY.xyz
```

Notes:

- `xyzfile` tells ORCA to read coordinates from HY.xyz
- Format: `xyzfile charge multiplicity filename`
- Must have both HY.inp and HY.xyz in same directory

Save: Ctrl+O, then Ctrl+X

Step 4: Create ORCA Input File for PY

Check your XYZ file exists:

```
ls PY.xyz
```

Create ORCA input file:

```
nano PY.inp
```

Input file contents:

```
! B3LYP pc-0 TightOPT
! Opt Freq
%maxcore 2000
xyzfile 0 1 PY.xyz
```

Save: Ctrl+O, then Ctrl+X

Step 5: Submit Your Calculations

Create submission script for HY:

```
nano HY_run.sh
```

Script contents:

```
#!/bin/bash
#SBATCH --job-name=HY_pc0
#SBATCH --time=01:00:00
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=1
#SBATCH --mem-per-cpu=4000
module load orca/6.0.0
/scic/app/cpu/orca/6.0.0/orca HY.inp > HY.out
```

Repeat for PY
Submit both jobs:

```
sbatch HY_run.sh
sbatch PY_run.sh
```

Monitor progress:

```
squeue -u $USER          # Check job status
tail -f HY.out           # Watch output (Ctrl+C to exit)
```

Step 6: Analyze Results and Fill Table

Extract final energies from output files:

```
grep "FINAL SINGLE POINT ENERGY" HY.out
grep "FINAL SINGLE POINT ENERGY" PY.out
```

Example calculation:

```
HY energy: HY Eh
PY energy: PY Eh
E = E(PY)Eh - E(HY)Eh
Convert to kJ/mol: 1Eh ~ 2625.5kJ/mol
```

Extract bond lengths:

```
grep -A 15 "INTERNAL COORDINATES (ANGSTROEM)" HY.out | tail -17
Bonds positions are listed with xyz coordinates
Pick bond length of first listed H atom
```

Fill in table in handout with pc-0 results:

- Energy difference (kJ/mol)
- Bond lengths (O-H, N-H, etc.)

Step 7: Check Your Results

Are your calculations done?

```
grep "ORCA TERMINATED NORMALLY" *.out
```

Extract YOUR final energies:

```
grep "FINAL SINGLE POINT ENERGY" *.out
grep "FINAL SINGLE POINT ENERGY" *.out
```

Compare to reference:

- Reference HY: -323.456 Eh
- Reference PY: -323.466 Eh

If values don't match:

- Check geometry and basis set input
- Ask TA for help

Step 8: Extract Bond Lengths

ORCA creates optimized geometry as .xyz file:

```
ls *.xyz
# Should see: HY_pc0.xyz  PY_pc0.xyz
```

Download to your computer:

```
# From your local computer:
scp c1cluster:~/Lab_BasisSet/HY/*.xyz ./
```

Open in Avogadro:

1. Open Avogadro
2. File → Open → Select HY_pc0.xyz
3. Click on O atom, then H atom
4. Bond length displays at bottom
5. Record: O-H = _____ Å

Reference value: O-H = 0.992 Å with pc-0

Summary of steps

Building:

- Avogadro → Draw molecule → Save as .xyz

Transfer:

- `scp file.xyz c1cluster: /Lab_BasisSet/HY/`

Setup:

- `export PATH=/scic/app/cpu/orca/6.0.0:$PATH`
- `nano HY_pc0.inp` - Create input
- Paste coordinates from .xyz (skip first 2 lines)

Running:

- `orca HY_pc0.inp > HY_pc0.out &`
- `tail -f HY_pc0.out` - Monitor

Extracting:

- `grep "FINAL SINGLE POINT ENERGY" HY_pc0.out`
- Download .xyz files and open in Avogadro for bond lengths