CHEM4301/5301 - Theory of Chemical Bonding Linux Tutorial

Alastair J. Price, Erin R. Johnson

Winter 2021

Required items for the course:

- Access to PC running Windows, OSX, or a distribution of Linux (recommended).
- High-speed internet connection.

Recommended Items:

- Webcam.
- Microphone.
- "Exploring Chemistry with Electronic Structure Methods: A Guide to Using Gaussian" by Foresman and Frisch.
- Avogadro `https://avogadro.cc/`

This will be an outline of getting started and how to do the computational chemistry components within this course. This outline should provide all of the steps to install and access the computational resources at Dalhousie University within the Johnson group. If you have any questions or issues with the course itself, you should first reference the course syllabus, before emailing Professor Johnson. All other questions related to how to access the computer resources should be directed to the Teaching Assistant, Alastair Price.

This document will be of particular importance for the final two assignments, which will use the Gaussian electronic-structure program. Students are expected to perform all Gaussian calculations independently and should save their input and output files on the Linux server. Students must have unique input and output files. Copying files from another student will result in a mark of zero on the related assignment.

1 SSH and SCP

Secure Shell (SSH) will be used widely in this course to navigate to and from the server. SSH is a cryptographic network protocol for operating networked computers securely over an unsecure network. This allows for a remote login from one computer to another, and has many alternative options for authentication. In this course, we will be utilising password based authentication in order to access the servers. The other major protocol that we will be using in this course is the secure copy protocol (SCP). Like SSH, this is a cryptographic network protocol but, instead of operating the networked computer, it acts as a means for file transfer between two networked computers. Other methods, such as `sftp` and `rsync` do exist (and can be used), but will not be discussed in this document.

If you will be working with a windows machine, continue reading in subsection 1.1; for OSX based computers continue on to subsection 1.2 and, for linux, subsection 1.3

1.1 Windows SSH and SCP

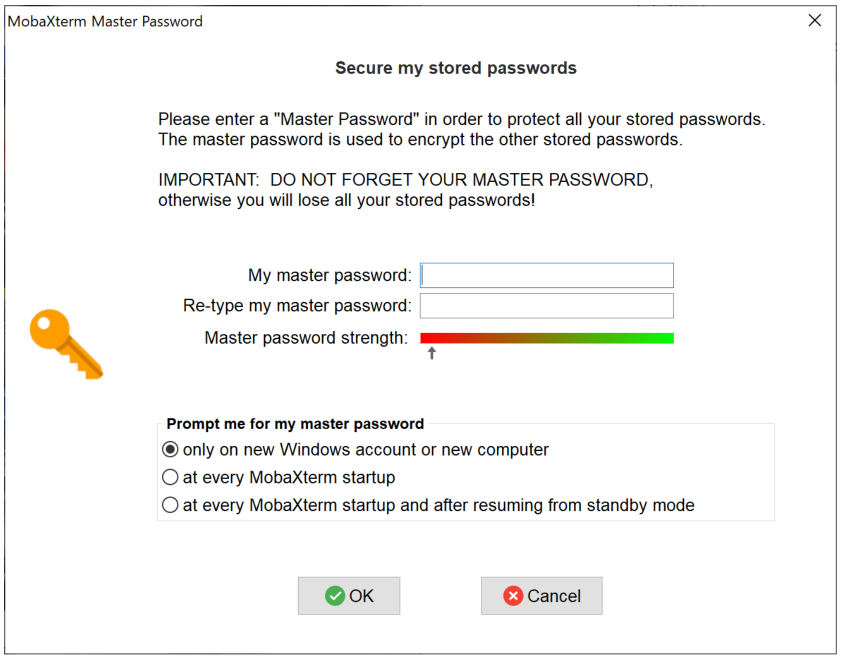
Historically, the ability to SSH has not been included within windows. Recent versions of Windows 10 have included a SSH client as an optional feature, but it is not installed by default. The two main methods available to users of windows machines are the PuTTY https://www.putty.org/ terminal and what will be demonstrated in this document, MobaXterm https://mobaxterm.mobatek.net/. MobaXterm is recommended due to its ease of use and graphical user interface (GUI) familiar to most Windows users.

MobaXterm provides windows all of the important network tools, such as SSH and SCP, as well as many basic unix commands that will be used within this course. Additionally, MobaXterm contains a graphical SFTP browser, bypassing the need to utilise SCP and negating the requirement to further install a Xserver to use any graphical content. MobaXterm can be downloaded from https://mobaxterm.mobatek.net/download.html. When downloading, make sure you first select the Home edition (which is free) before selecting the MobaXterm Home edition (Installer edition) https://mobaxterm.mobatek.net/download-home-edition.html. Once selected, a dialogue box should appear to begin the download with a zip directory containing the MobaXterm client.

1.1.1 Installation of MobaXterm

Once downloaded, use your favourite unzip tool to extract all the files from the directory. Some examples of zip tools are 7zip https://www.7-zip.org/, winrar https://www.win-rar.com/start.html?&L=0, or WinZip https://www.winzip.com/win/en/zip-file.html. After extraction, double click the Windows Installer package and follow the prompts to completion. After MobaXterm is fully installed, it can be opened by double clicking the icon or navigating to the executable within the program files.

Upon first execution of MobaXterm, the user will be prompted with a dialogue box requesting a strong master password, as in Fig. 1.



MobaXterm Master Password

Secure my stored passwords

Please enter a "Master Password" in order to protect all your stored passwords.
The master password is used to encrypt the other stored passwords.

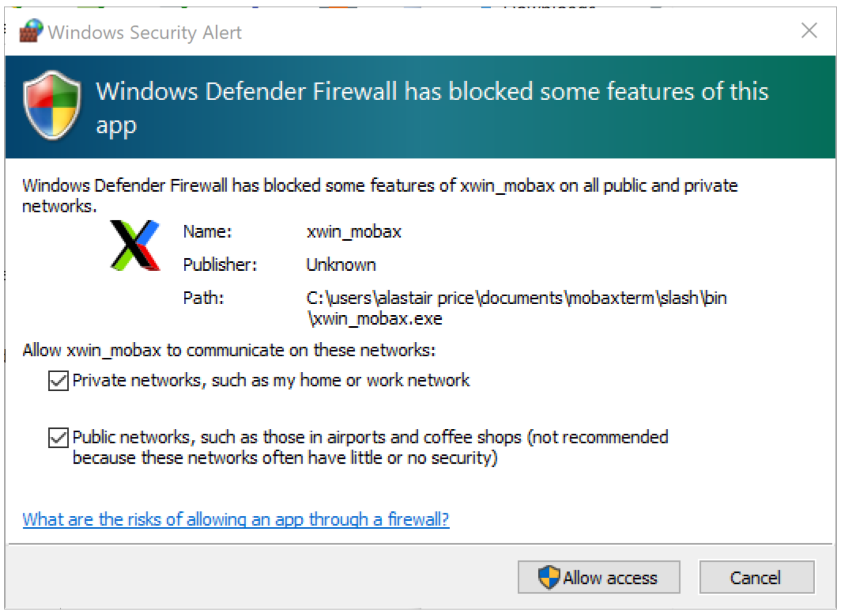
IMPORTANT: DO NOT FORGET YOUR MASTER PASSWORD,
otherwise you will lose all your stored passwords!

My master password:
Re-type my master password:
Master password strength:

Prompt me for my master password
[✓ only on new Windows account or new computer]
at every MobaXterm startup
at every MobaXterm startup and after resuming from standby mode

Figure 1: MobaXterm Master password; once set it must not be forgotten, or all passwords will be lost.

After setting a strong password and clicking OK, you will likely be prompted with a windows defender dialogue requesting network access. It is recommended that you give MobaXterm the fullest network access by clicking both the private and public networks before clicking Allow, as seen in Fig. 2.


Windows Security Alert

Windows Defender Firewall has blocked some features of this app

Windows Defender Firewall has blocked some features of xwin_mobax on all public and private networks.

Name:	xwin_mobax
Publisher:	Unknown
Path:	C:\users\alastair price\documents\mobaxterm\slash\bin \xwin_mobax.exe

Allow xwin_mobax to communicate on these networks:

☑ Private networks, such as my home or work network

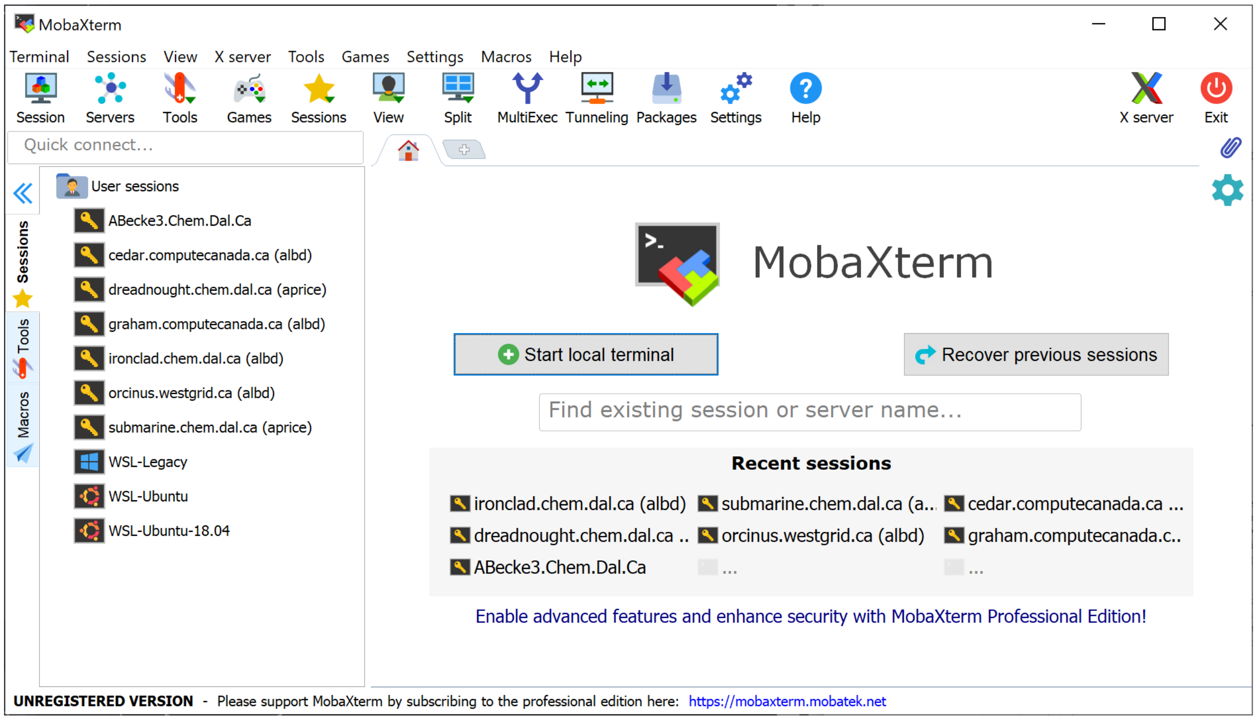
☑ Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)

What are the risks of allowing an app through a firewall?

Figure 2: This step will be specific to your version and windows setup.

1.1.2 MobaXterm use

Once MobaXterm has started, you will be greeted with a screen similar to that shown in Fig. 3. From this point, we will start with clicking the start local terminal button in the centre of the screen. This will open a new tab or terminal, looking like Fig. 4, which will allow the use of all of the tools and methods described below.



MobaXterm
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
Sessions
Tools
Macros
MobaXterm
Start local terminal
Recover previous sessions
Find existing session or server name...

User sessions
- ABecke3.Chem.Dal.Ca
- cedar.computecanada.ca (albd)
- dreadnought.chem.dal.ca (aprice)
- graham.computecanada.ca (albd)
- ironclad.chem.dal.ca (albd)
- orcinus.westgrid.ca (albd)
- submarine.chem.dal.ca (aprice)
- WSL-Legacy
- WSL-Ubuntu
- WSL-Ubuntu-18.04

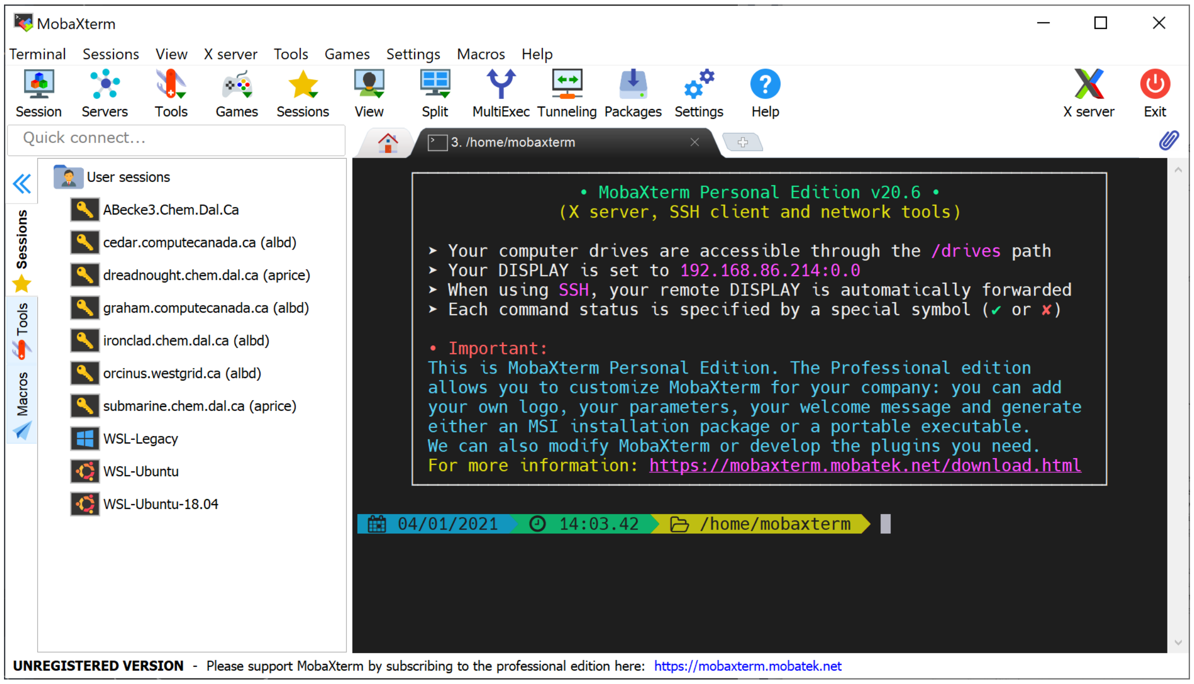
Recent sessions

ironclad.chem.dal.ca (albd)	submarine.chem.dal.ca (a...	cedar.computecanada.ca ...
dreadnought.chem.dal.ca ..	orcinus.westgrid.ca (albd)	graham.computecanada.c..
ABecke3.Chem.Dal.Ca	...	...

Enable advanced features and enhance security with MobaXterm Professional Edition!

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

Figure 3: The start screen of mobaXterm.


MobaXterm
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
X server Exit
Quick connect...
3. /home/mobaxterm
User sessions
ABecke3.Chem.Dal.Ca
cedar.computecanada.ca (albd)
dreadnought.chem.dal.ca (aprice)
graham.computecanada.ca (albd)
ironclad.chem.dal.ca (albd)
orcinus.westgrid.ca (albd)
submarine.chem.dal.ca (aprice)
WSL-Legacy
WSL-Ubuntu
WSL-Ubuntu-18.04
Sessions
Tools
Macros

```
              • MobaXterm Personal Edition v20.6 •
            (X server, SSH client and network tools)

 ➤ Your computer drives are accessible through the /drives path
 ➤ Your DISPLAY is set to 192.168.86.214:0.0
 ➤ When using SSH, your remote DISPLAY is automatically forwarded
 ➤ Each command status is specified by a special symbol (✔ or ✘)

 • Important:
 This is MobaXterm Personal Edition. The Professional edition
 allows you to customize MobaXterm for your company: you can add
 your own logo, your parameters, your welcome message and generate
 either an MSI installation package or a portable executable.
 We can also modify MobaXterm or develop the plugins you need.
 For more information: https://mobaxterm.mobatek.net/download.html

 04/01/2021  14:03.42  /home/mobaxterm
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

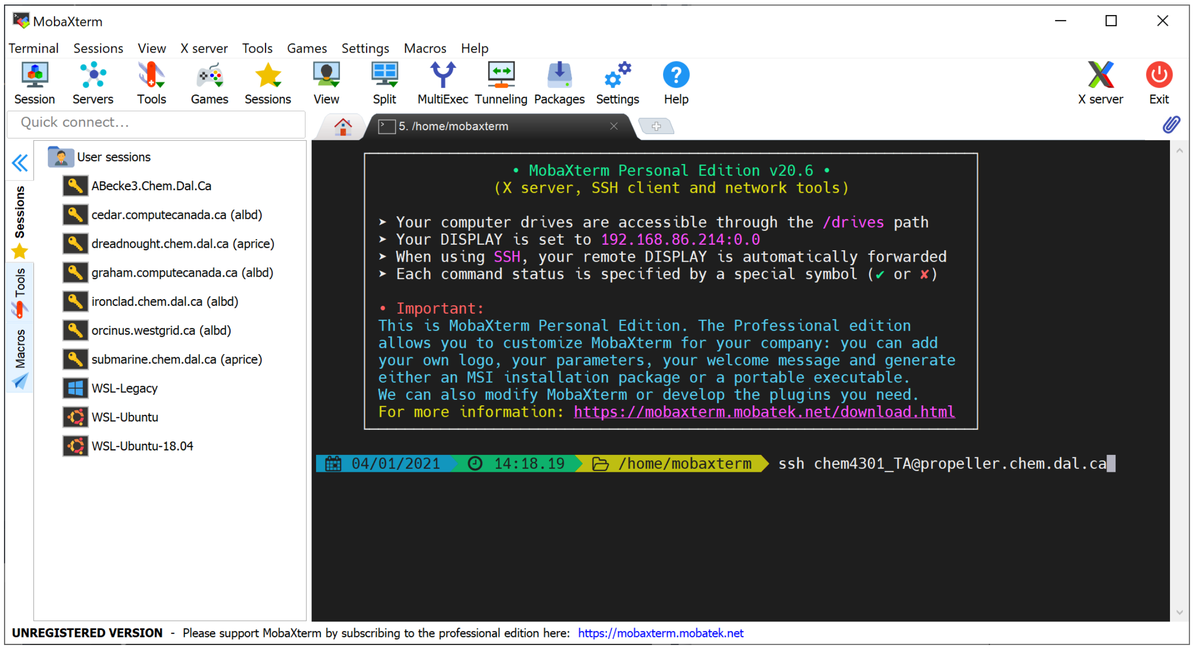
Figure 4: MobaXterm Terminal.

All SSH sessions will require a few basic pieces of information in order to connect. The first will be your username, which will be determined and provided for you by the course instructor or TA (the example will use `chem4301_TA`). The second is the location of the SSH server, which in our case will be propeller at Dalhousie (`@propeller.chem.dal.ca`). Finally, you will need the password, which should be changed upon your first login, using the `passwd` command. The command that will be used to login is `ssh chem4301_TA@propeller.chem.dal.ca`, as shown in Fig. 5. Here, `chem4301_TA` should be replaced by your specific username.

Upon hitting enter on the ssh command above, you will either be shown `Warning: Permanently added 'propeller.chem.dal.ca' (RSA) to the list of known hosts.` and/or prompted with a request for your password, as in Fig. 6. You will be able to ignore or type in yes to the above warning and then proceed to enter your password. Of note for security purposes, it will appear that no password is being entered. DON'T PANIC, proceed to enter the full password before hitting enter. At this point, you can choose to save the password by clicking yes on the dialogue box that appears. Congrats, you have now connected remotely to propeller at Dalhousie via ssh. The prompt should look like Fig. 7.

1.1.3 SCP on Windows

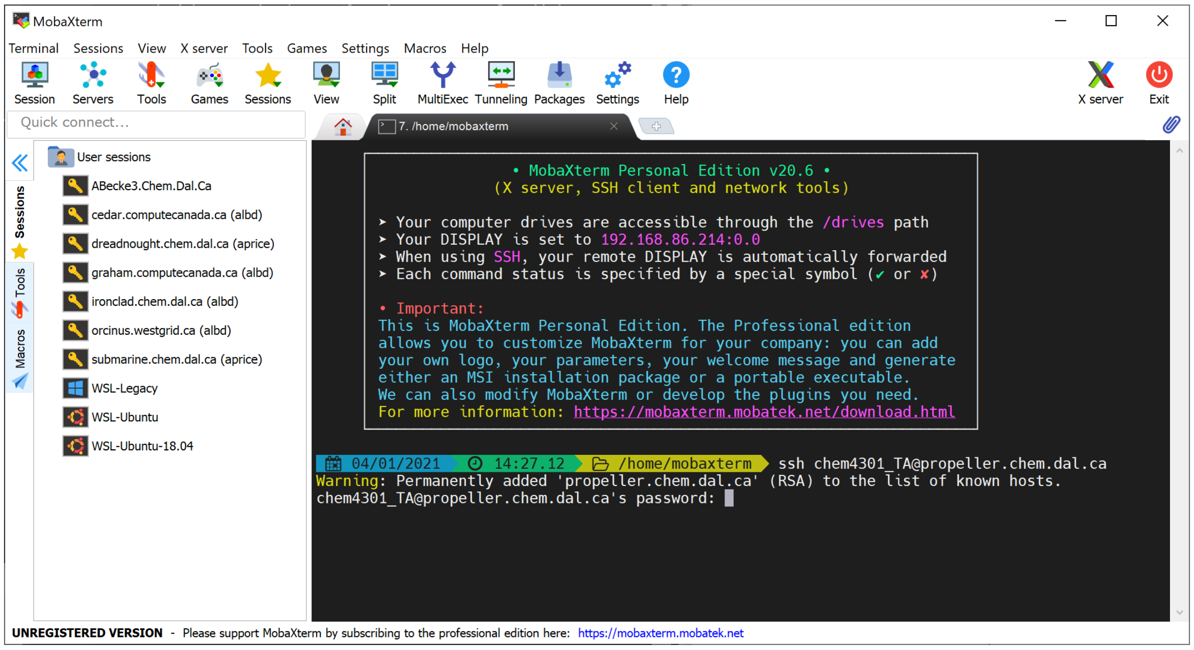
Moving on from SSH, the other major tool that will be used is SCP. SCP will allow you to copy files from one computer (such as your own) to the server in the lab (and many more). The syntax for the `scp` command is very similar to that of the `ssh` command above. Starting from the same place as in Fig. 4, we will scp a file from the server to your home computer. Note you will only be able to execute the `scp` command from your home computer, unless you have a static IP or a registered domain name, which is necessary to be able to connect to it from the lab server. The `scp` command has the default syntax of `scp [OPTION] [user@]SRC_HOST:]file1 [user@]DEST_HOST:]file2`, where `[OPTION]` is one or more of the scp options, such as cipher, ssh configuration, ssh port, limit, recursive and many more (we will be primarily using the `-r` option), the `[user@]SRC_HOST:]file1` will be the source file and location, and `[user@]DEST_HOST:]file2` is the destination of the file. For example, to transfer the file `file.txt` from the server to your computer, the command will be `scp chem4301_TA@propeller.chem.dal.ca:/home/chem4301_TA/file.txt .` as can be seen in Fig. 8.


MobaXterm
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
5. /home/mobaxterm
User sessions
ABecke3.Chem.Dal.Ca
cedar.computecanada.ca (albd)
dreadnought.chem.dal.ca (aprice)
graham.computecanada.ca (albd)
ironclad.chem.dal.ca (albd)
orcinus.westgrid.ca (albd)
submarine.chem.dal.ca (aprice)
WSL-Legacy
WSL-Ubuntu
WSL-Ubuntu-18.04
Sessions Tools Macros

```
• MobaXterm Personal Edition v20.6 •
(X server, SSH client and network tools)

➤ Your computer drives are accessible through the /drives path
➤ Your DISPLAY is set to 192.168.86.214:0.0
➤ When using SSH, your remote DISPLAY is automatically forwarded
➤ Each command status is specified by a special symbol (✔ or ✘)

• Important:
This is MobaXterm Personal Edition. The Professional edition
allows you to customize MobaXterm for your company: you can add
your own logo, your parameters, your welcome message and generate
either an MSI installation package or a portable executable.
We can also modify MobaXterm or develop the plugins you need.
For more information: https://mobaxterm.mobatek.net/download.html

 04/01/2021   14:18.19   /home/mobaxterm  ssh chem4301_TA@propeller.chem.dal.ca
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

Figure 5: MobaXterm Terminal ssh command.


MobaXterm
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
7. /home/mobaxterm
User sessions
ABecke3.Chem.Dal.Ca
cedar.computecanada.ca (albd)
dreadnought.chem.dal.ca (aprice)
graham.computecanada.ca (albd)
ironclad.chem.dal.ca (albd)
orcinus.westgrid.ca (albd)
submarine.chem.dal.ca (aprice)
WSL-Legacy
WSL-Ubuntu
WSL-Ubuntu-18.04
Sessions Tools Macros

```
• MobaXterm Personal Edition v20.6 •
(X server, SSH client and network tools)

➤ Your computer drives are accessible through the /drives path
➤ Your DISPLAY is set to 192.168.86.214:0.0
➤ When using SSH, your remote DISPLAY is automatically forwarded
➤ Each command status is specified by a special symbol (✔ or ✘)

• Important:
This is MobaXterm Personal Edition. The Professional edition
allows you to customize MobaXterm for your company: you can add
your own logo, your parameters, your welcome message and generate
either an MSI installation package or a portable executable.
We can also modify MobaXterm or develop the plugins you need.
For more information: https://mobaxterm.mobatek.net/download.html

 04/01/2021   14:27.12   /home/mobaxterm  ssh chem4301_TA@propeller.chem.dal.ca
Warning: Permanently added 'propeller.chem.dal.ca' (RSA) to the list of known hosts.
chem4301_TA@propeller.chem.dal.ca's password:
```

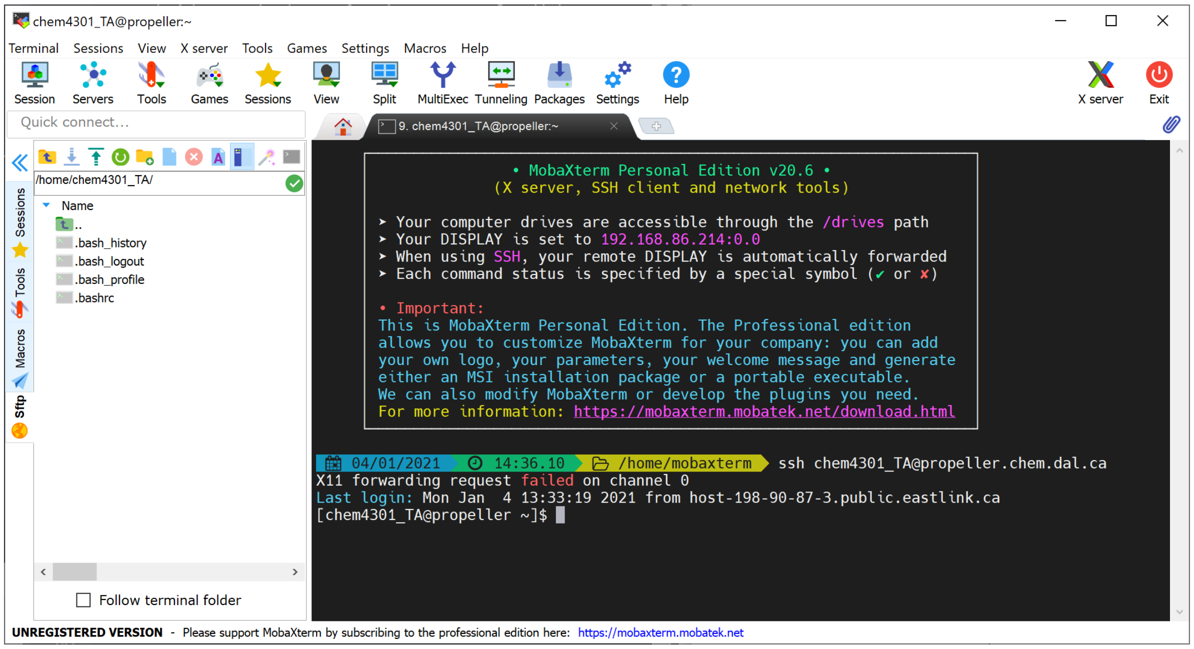
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

Figure 6: MobaXterm Terminal.

After hitting `Enter`, this will copy the file to your local machine and place it in your current directory (folder), represented by `./` as can be seen in Fig. 9. The opposite can be done, and sending a file from your local machine to the server is achieved by the command

```
scp file.txt chem4301_TA@propeller.chem.dal.ca:/home/chem4301_TA/
```

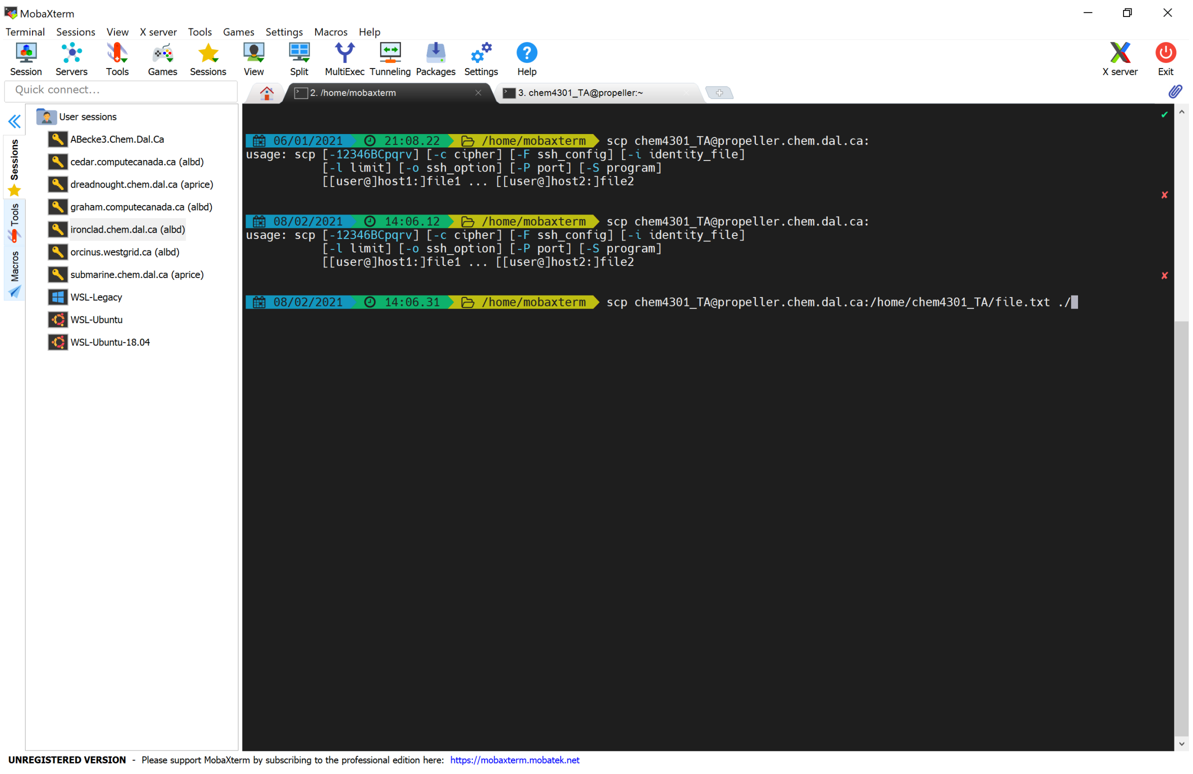
This will place a file named `file.txt` from your local computer back onto the server. **Be careful** if you have different versions of files that have the same names on both your local machine and the server, since this command can and will overwrite the file at the destination with the source (with great power comes great



```
  • MobaXterm Personal Edition v20.6 •
(X server, SSH client and network tools)

➤ Your computer drives are accessible through the /drives path
➤ Your DISPLAY is set to 192.168.86.214:0.0
➤ When using SSH, your remote DISPLAY is automatically forwarded
➤ Each command status is specified by a special symbol (✔ or ✘)

• Important:
This is MobaXterm Personal Edition. The Professional edition
allows you to customize MobaXterm for your company: you can add
your own logo, your parameters, your welcome message and generate
either an MSI installation package or a portable executable.
We can also modify MobaXterm or develop the plugins you need.
For more information: https://mobaxterm.mobatek.net/download.html

 04/01/2021  14:36.10  /home/mobaxterm  ssh chem4301_TA@propeller.chem.dal.ca
X11 forwarding request failed on channel 0
Last login: Mon Jan  4 13:33:19 2021 from host-198-90-87-3.public.eastlink.ca
[chem4301_TA@propeller ~]$ 
```

chem4301_TA@propeller:~
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
9. chem4301_TA@propeller:~
/home/chem4301_TA/
Name
..
.bash_history
.bash_logout
.bash_profile
.bashrc
Sessions Tools Macros Sftp
Follow terminal folder
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

Figure 7: MobaXterm Terminal.

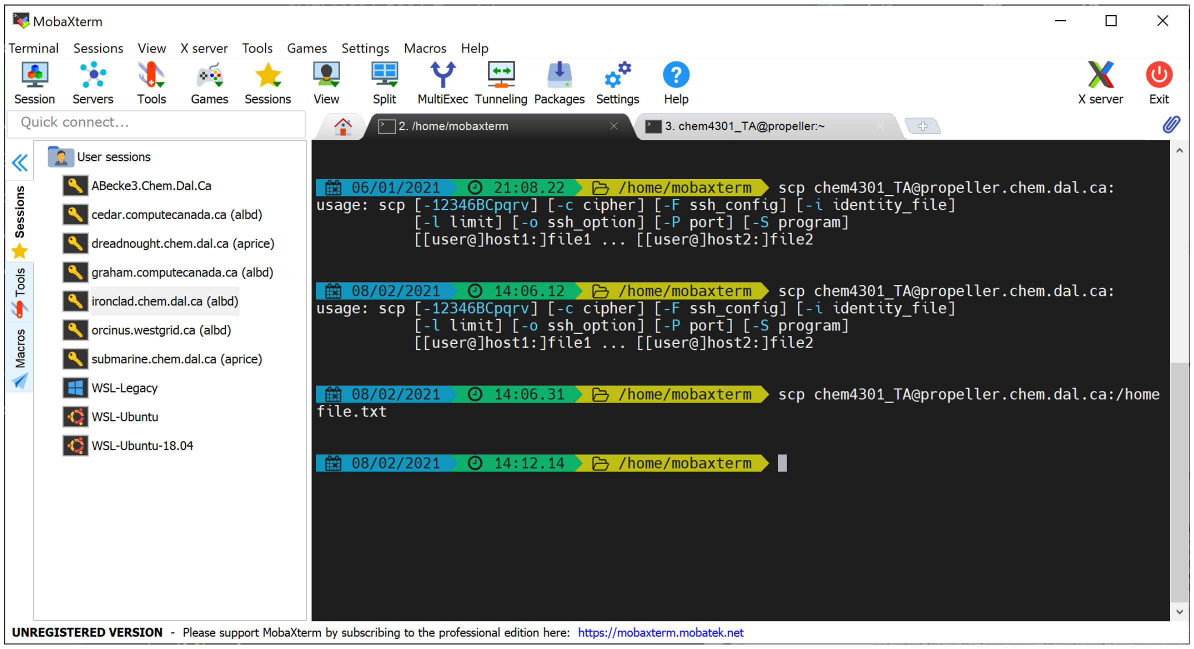


```
 06/01/2021  21:08.22  /home/mobaxterm  scp chem4301_TA@propeller.chem.dal.ca:
usage: scp [-12346BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
           [-l limit] [-o ssh_option] [-P port] [-S program]
           [[user@]host1:]file1 ... [[user@]host2:]file2

 08/02/2021  14:06.12  /home/mobaxterm  scp chem4301_TA@propeller.chem.dal.ca:
usage: scp [-12346BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
           [-l limit] [-o ssh_option] [-P port] [-S program]
           [[user@]host1:]file1 ... [[user@]host2:]file2

 08/02/2021  14:06.31  /home/mobaxterm  scp chem4301_TA@propeller.chem.dal.ca:/home/chem4301_TA/file.txt ./
```

MobaXterm
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
2. /home/mobaxterm
3. chem4301_TA@propeller:~
User sessions
ABecke3.Chem.Dal.Ca
cedar.computecanada.ca (albd)
dreadnought.chem.dal.ca (aprice)
graham.computecanada.ca (albd)
ironclad.chem.dal.ca (albd)
orcinus.westgrid.ca (albd)
submarine.chem.dal.ca (aprice)
WSL-Legacy
WSL-Ubuntu
WSL-Ubuntu-18.04
Sessions Tools Macros
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

Figure 8: MobaXterm Terminal SCP command.

responsibility). As noted earlier, if you want to transfer whole directories (otherwise known as folders to Windows users), this can be done by using the `-r` flag with the SCP command. For example, if I wanted to transfer my whole home directory, including `file.txt`, from propeller to my local computer, the command


MobaXterm
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
X server Exit
Quick connect...
2. /home/mobaxterm
3. chem4301_TA@propeller:~
User sessions
ABecke3.Chem.Dal.Ca
cedar.computecanada.ca (albd)
dreadnought.chem.dal.ca (aprice)
graham.computecanada.ca (albd)
ironclad.chem.dal.ca (albd)
orcinus.westgrid.ca (albd)
submarine.chem.dal.ca (aprice)
WSL-Legacy
WSL-Ubuntu
WSL-Ubuntu-18.04
Sessions
Tools
Macros
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

```
 06/01/2021  21:08.22  /home/mobaxterm  scp chem4301_TA@propeller.chem.dal.ca:
usage: scp [-12346BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
           [-l limit] [-o ssh_option] [-P port] [-S program]
           [[user@]host1:]file1 ... [[user@]host2:]file2

 08/02/2021  14:06.12  /home/mobaxterm  scp chem4301_TA@propeller.chem.dal.ca:
usage: scp [-12346BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
           [-l limit] [-o ssh_option] [-P port] [-S program]
           [[user@]host1:]file1 ... [[user@]host2:]file2

 08/02/2021  14:06.31  /home/mobaxterm  scp chem4301_TA@propeller.chem.dal.ca:/home
file.txt

 08/02/2021  14:12.14  /home/mobaxterm 
```

Figure 9: MobaXterm SCP command result.

would be `scp -r chem4301_TA@propeller.chem.dal.ca:/home/chem4301_TA .`

Depending on your experience with the command line, you may also use the command `rsync`, which is a more modern form of scp. The transferring of files will be one of the major commands used in this course, as it will allow for local editing and imaging of your Gaussian input and output files.

1.2 OSX SSH and SCP

OSX/MacOS, unlike Windows, has a built-in terminal environment, allowing you to work in the same way you would do with GNU/Linux, but on your home computer. To begin, you will need to open the terminal via spotlight search (this being the easiest method) with `Command-Space` and then typing in terminal, as can be seen in Fig. 10.

Once you have your terminal opened, if you are running MacOS Catalina or greater you will likely need to switch your shell from Zsh to Bash, which can be done simply by typing `bash`. Although changing shell is not 100% necessary, this tutorial has not been tested with Zsh. After this, all of the examples from Sec. 1.1.2 for SSH and SCP will be the same between all operating systems.

1.3 Linux SSH and SCP

As for the Linux users remaining, the commands of `ssh`, `scp`, and `rsync` will likely be already known to you. In the event that they are new, I recommend looking back at the MobaXterm commands, as they will be almost identical for all operating systems once the terminal has been opened.

2 Terminal Basics

For this section, the commands will be similar between all three platforms (Linux, Unix, and Windows) as you will be either using a terminal emulator or be connected directly to the server. The basic commands that we will be using are `ls`, `mkdir`, `cd`, `pwd`, `cp`, `mv`, `more`, `less`, `cat`, `head`, `tail`, `grep`, `awk`, and `sed`. These commands are fundamental to the understanding and use of the command line with Bash, which is the default login shell found in most Linux distributions.

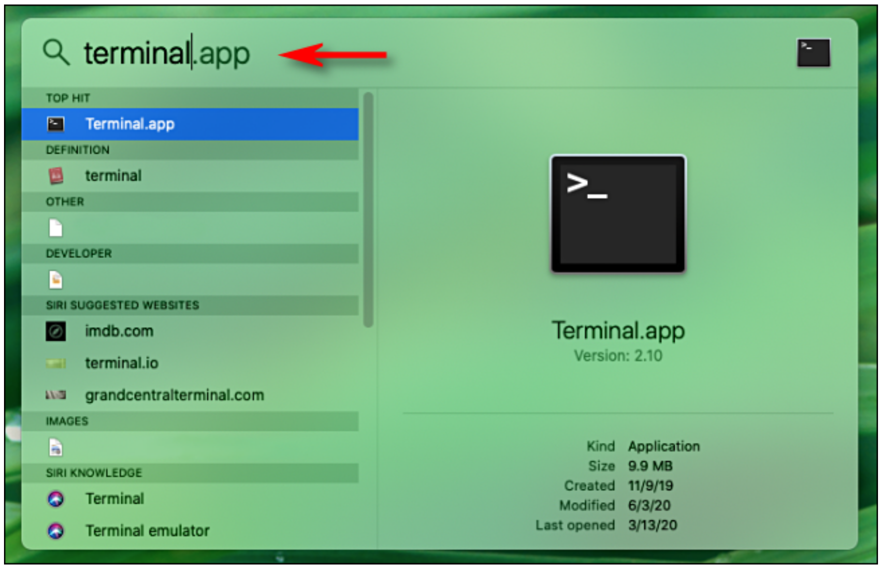
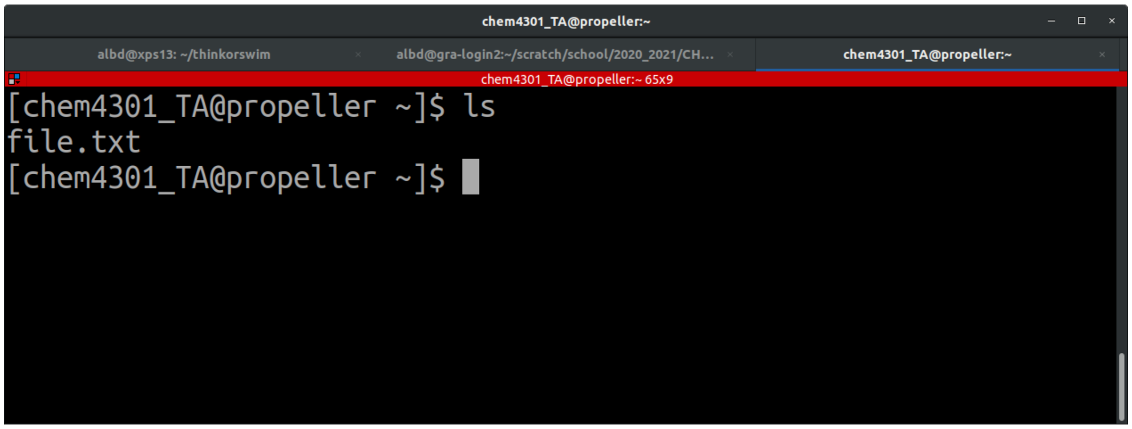


Figure 10: Spotlight Search for Terminal.

2.1 The `ls` command

The `ls` command is possibly one of the most common and important commands that you will use when working with the command line interface (or the terminal). The `ls` command allows you to see the files and sub-directories that are located in your current working directory. For example, the ls command syntax has the form `ls [option(s)] [file]`. Executing the `ls` command in the home directory on propeller will give Fig. 11.


chem4301_TA@propeller:~
albd@xps13: ~/thinkorswim
albd@gra-login2:~/scratch/school/2020_2021/CH...
chem4301_TA@propeller:~
chem4301_TA@propeller:~ 65x9

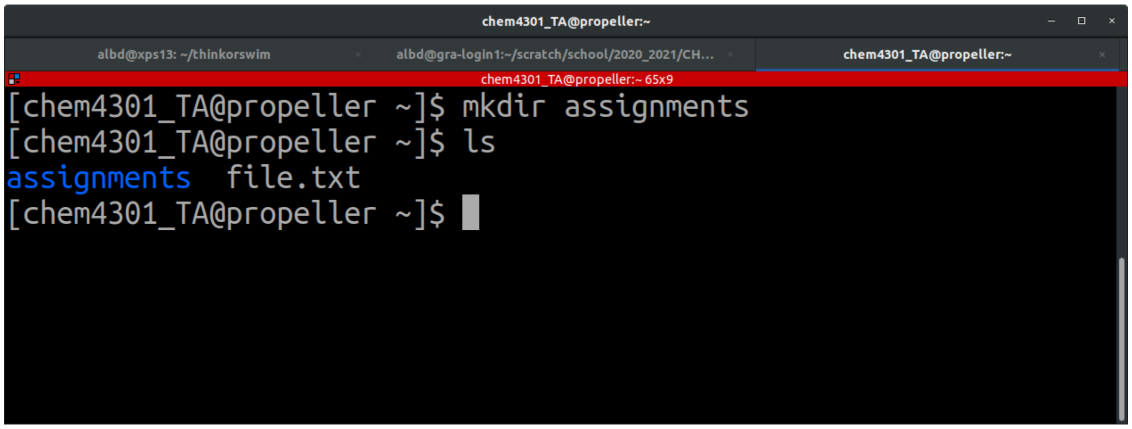
```
[chem4301_TA@propeller ~]$ ls
file.txt
[chem4301_TA@propeller ~]$ 
```

Figure 11: The `ls` command.

In this case, the `ls` command has listed the only file in the directory, which is `file.txt` from the scp example earlier. Some common options for the `ls` command are `ls -a`, which shows all of the files and directories (even the hidden ones), and `ls -l`, which provides the long list format containing information such as the read/write-ability of the file, the owner, when it was last changed, etc. The `ls` command, like many other commands, can be executed in one directory yet pointed at another, which we will see later.

2.2 The `mkdir` command

The `mkdir` command does what it says on the tin – it makes directories or `mkdir`. It has the syntax `mkdir [options] directory_name(s)` and any number of directories can be made with this command simultaneously. If one wanted to create a new directory on propeller named `assignments`, the command would be `mkdir assignments`. As shown in Fig. 12, the new directory can be seen in blue after typing in the above `ls` command.

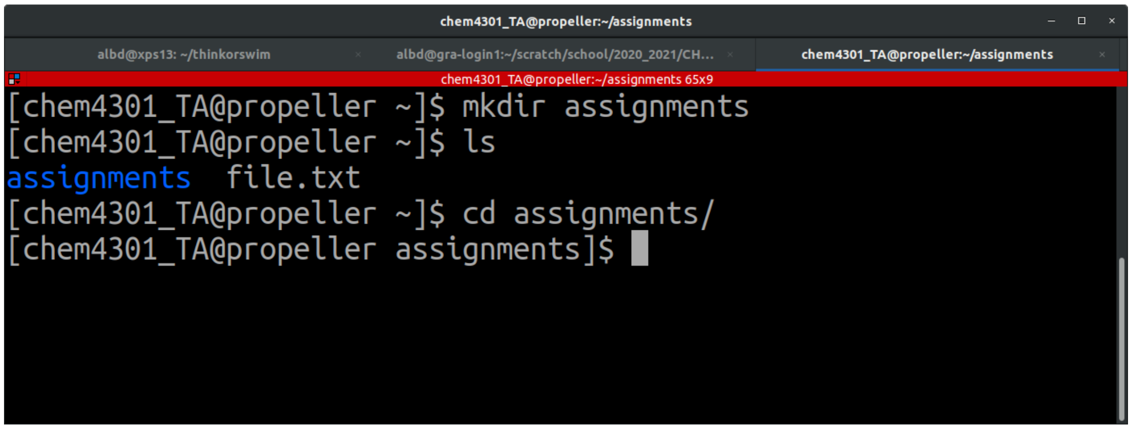


```
[chem4301_TA@propeller ~]$ mkdir assignments
[chem4301_TA@propeller ~]$ ls
assignments  file.txt
[chem4301_TA@propeller ~]$
```

Figure 12: The `mkdir` command.

2.3 The `cd` command

The `mkdir` command leads nicely into the next command, `cd`, which allows the user to change directory. The `cd` command will allow you to change your present working directory and has the syntax `cd directory`. For example, if one wishes to access the `assignments` directory just created above, one would execute `cd assignments/` as in Fig. 13.



```
[chem4301_TA@propeller ~]$ mkdir assignments
[chem4301_TA@propeller ~]$ ls
assignments  file.txt
[chem4301_TA@propeller ~]$ cd assignments/
[chem4301_TA@propeller assignments]$
```

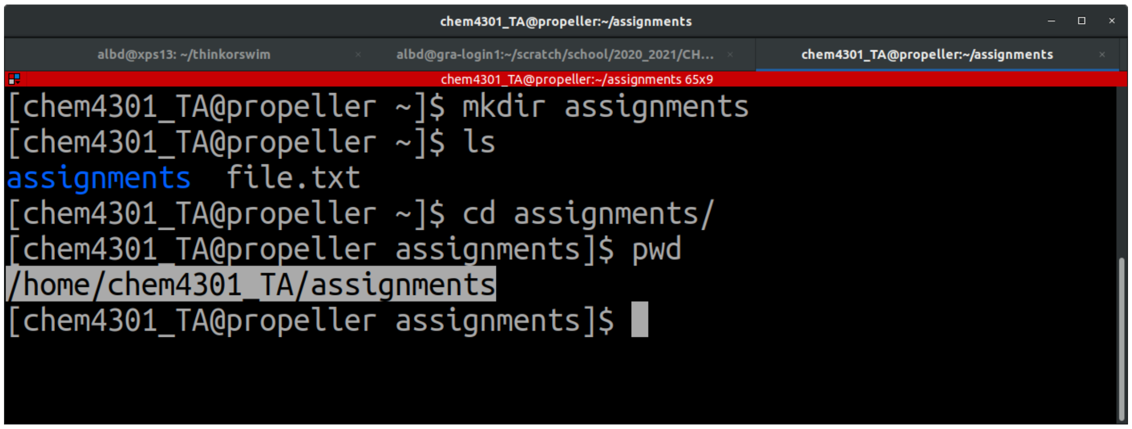
Figure 13: The `cd` command.

Of note here, bash is case sensitive and, therefore, so is `cd` and all other commands in this tutorial. It is important that you type file or directory names exactly as they are shown.

The other side to the `cd` command is when needing to go up a directory. In Fig.13, if you wanted to go back up to the parent directory of assignments, you would execute the command `cd ..` The `cd` command can also have the names of several directories chained so that you can move through more than one at a time. Finally, use simply `cd` to return to your home directory from any point.

2.4 The `pwd` command

An important command that helps navigate your directory tree is the `pwd` command, which stands for print working directory. It prints out the full path of your present working directory and the syntax is simply pwd. An example can be seen in Fig. 14, where the highlighted text is the location of the present working directory.

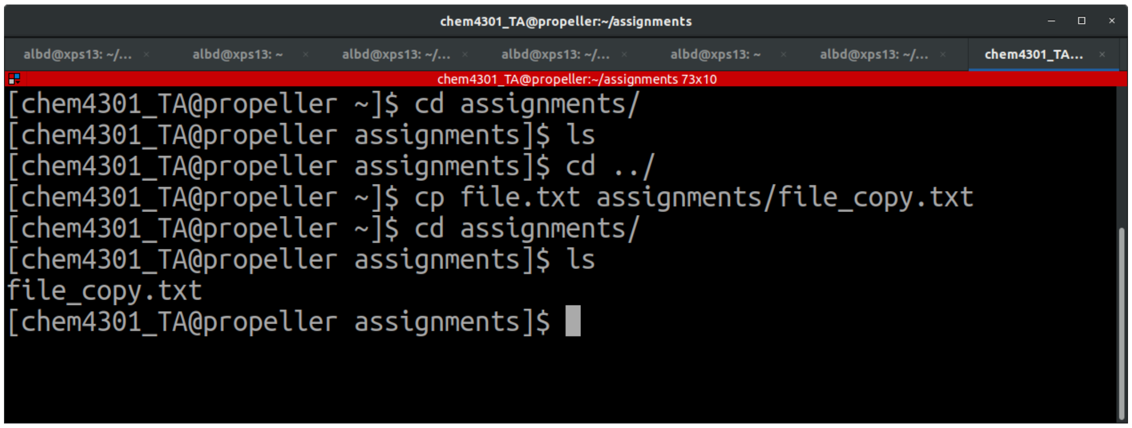

chem4301_TA@propeller:~/assignments
albd@xps13: ~/thinkorswim
albd@gra-login1:~/scratch/school/2020_2021/CH...
chem4301_TA@propeller:~/assignments
chem4301_TA@propeller:~/assignments 65x9

```
[chem4301_TA@propeller ~]$ mkdir assignments
[chem4301_TA@propeller ~]$ ls
assignments  file.txt
[chem4301_TA@propeller ~]$ cd assignments/
[chem4301_TA@propeller assignments]$ pwd
/home/chem4301_TA/assignments
[chem4301_TA@propeller assignments]$ 
```

Figure 14: The pwd command.

2.5 The cp command

The next command is the copy command or cp. This command has a similar syntax as the scp command, and has the form cp [option(s)] current_name new_name. This command will copy a specific file or directory and can be used either to make a copy with a new name, or to place a copy of the file or directory in a new location. An example of copying a file can be seen in Fig. 15, where file.txt was copied from the home directory to the assignments directory, with the new name of file_copy.txt.



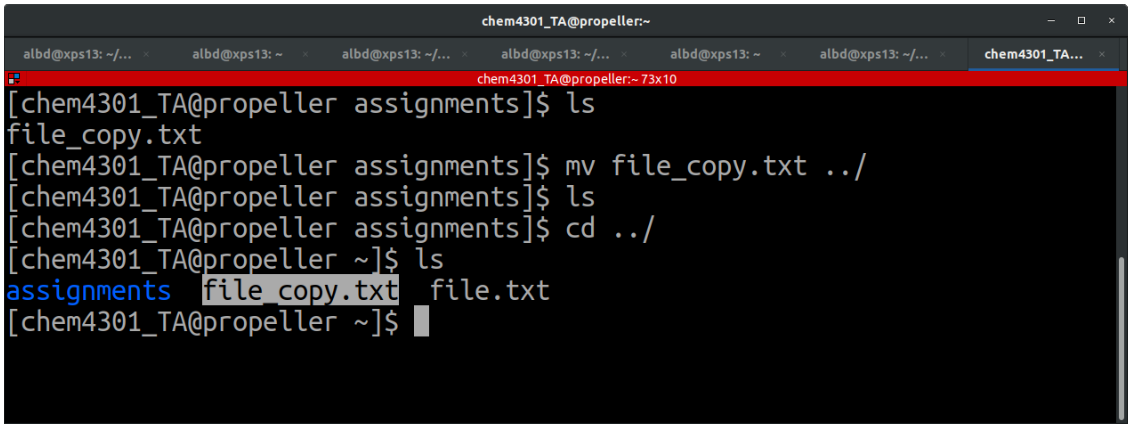
```
[chem4301_TA@propeller ~]$ cd assignments/
[chem4301_TA@propeller assignments]$ ls
[chem4301_TA@propeller assignments]$ cd ../
[chem4301_TA@propeller ~]$ cp file.txt assignments/file_copy.txt
[chem4301_TA@propeller ~]$ cd assignments/
[chem4301_TA@propeller assignments]$ ls
file_copy.txt
[chem4301_TA@propeller assignments]$ 
```

Figure 15: The cp command.

In order to copy a whole directory with cp, the -r flag must be used, which indicates recursive copying. This is similar to the use of the -r option with scp. Another useful option flag for the cp command is -i, which copies interactively, prompting you before it overwrites any preexisting files.

2.6 The mv command

In the same vein as the cp command is the move command or mv. This command has the syntax of mv [option(s)] current_name new_name and will move a specific file or directory from its current name and/or directory to a new name and/or directory. It can be used to rename a file, to move it, or both simultaneously. For example, moving the file file_copy.txt from the assignments directory to the home directory can be seen in Fig. 16. One of the few major differences between the `mv` command and the `cp` command is that the `mv` command does not need the `-r` flag in order to move entire directories.

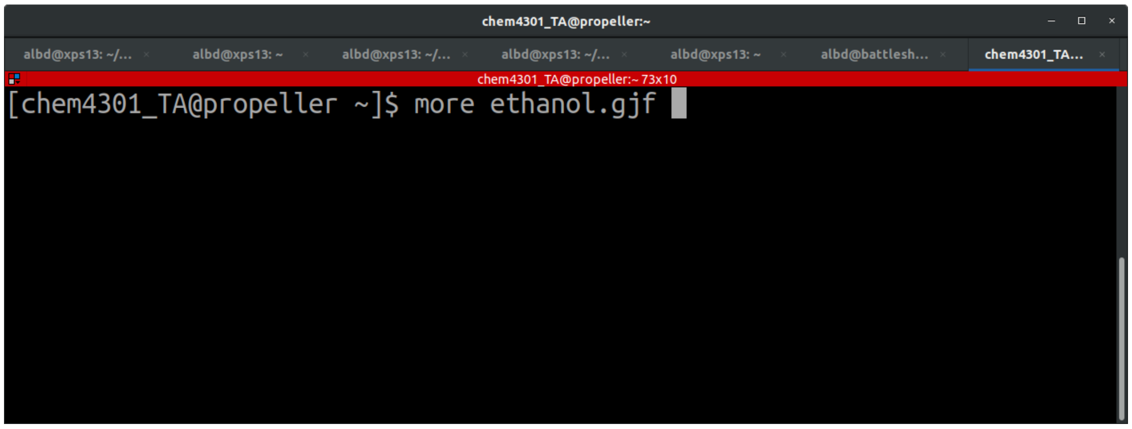


```
[chem4301_TA@propeller assignments]$ ls
file_copy.txt
[chem4301_TA@propeller assignments]$ mv file_copy.txt ../
[chem4301_TA@propeller assignments]$ ls
[chem4301_TA@propeller assignments]$ cd ../
[chem4301_TA@propeller ~]$ ls
assignments  file_copy.txt  file.txt
[chem4301_TA@propeller ~]$ 
```

Figure 16: The `mv` command.

2.7 The `more`, `less` and `cat` commands

These three commands are extremely useful in viewing the contents of the files you will generate in this course, with each command having slightly different effects. We will start with the `more` command, which has the syntax of `more [option(s)] file_name`, as in Fig. 17.



```
[chem4301_TA@propeller ~]$ more ethanol.gjf 
```

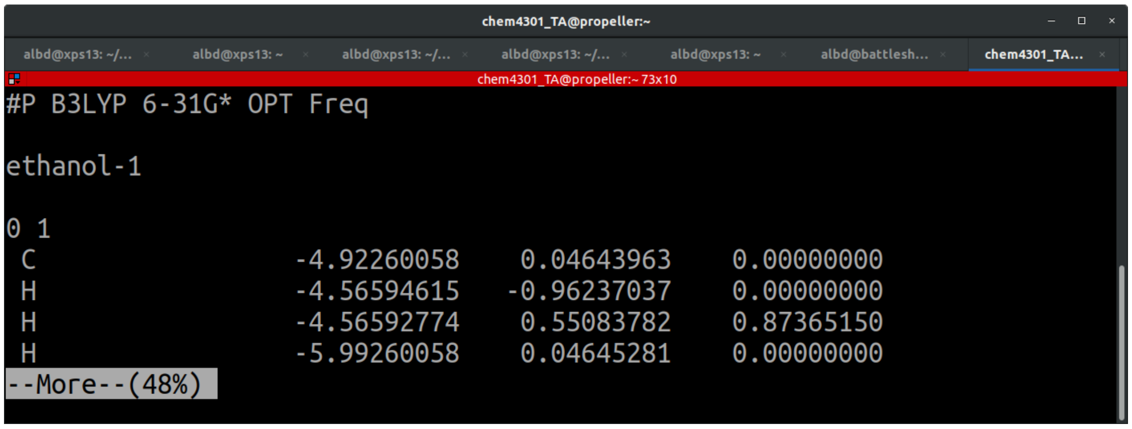
Figure 17: The `more` command.

Executing the `more` command opens the contents of the file for inspection. You can then scroll through the file by using the `spacebar` to move one page forward (or `Enter` to move forward by only one line) and the `b` key to move one page backwards. The result can be seen in Fig. 18.

To search for a string in the file, hit the `/` key and then type the string you want to find, followed by `Enter`. To exit from the `more` command, either continue paging to the bottom of the file, or hit `q` on the keyboard, and you will be returned to the terminal.

The `less` command has the syntax `less [option(s)] file_name`. It provides a similar output to that of `more`, with the main difference being that the `less` command is technically faster, as it does not load the entire file at once. Files can be navigated in the same manner using both `less` and `more`.

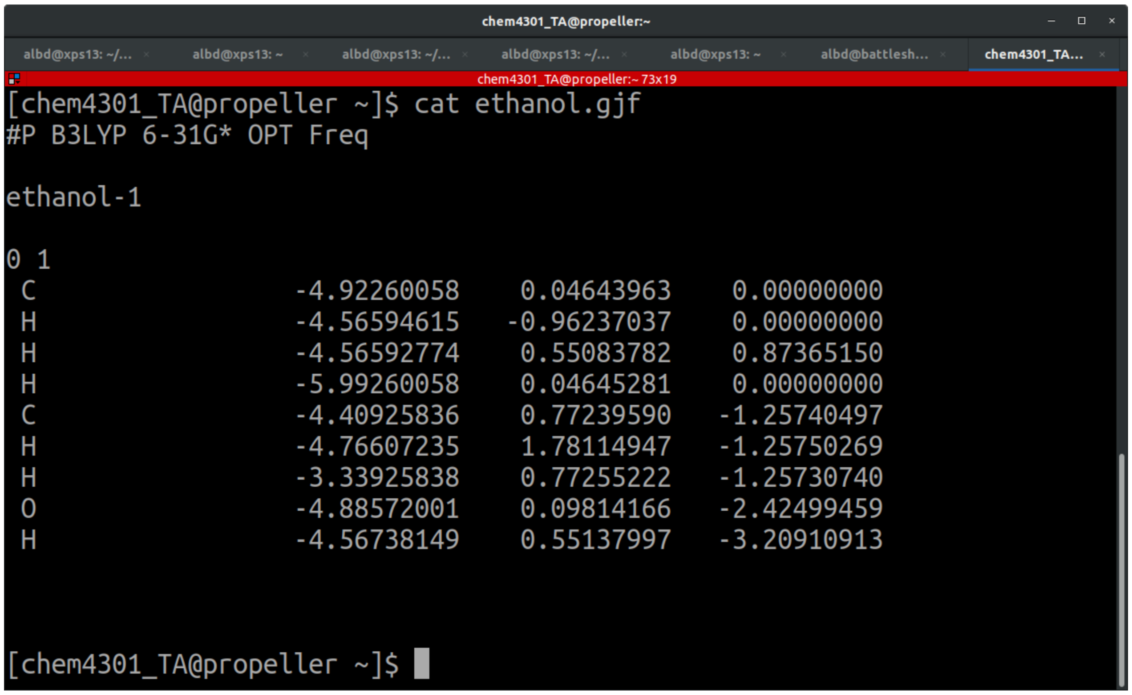
Finally, the `cat` or concatenate command has the syntax `cat [option(s)] file_name(s)`, like `more` or `less`. It has two main functions: concatenating (combining) files and printing the results to the terminal. If the `cat` command is used on a single file, it will simply display that file, as shown in Fig. 19. The `cat`



```
#P B3LYP 6-31G* OPT Freq

ethanol-1

0 1
 C                  -4.92260058    0.04643963    0.00000000
 H                  -4.56594615   -0.96237037    0.00000000
 H                  -4.56592774    0.55083782    0.87365150
 H                  -5.99260058    0.04645281    0.00000000
--More--(48%)
```

Figure 18: The more command result.

command is very useful when combined with redirects and appends in order to create a single larger file from smaller component files.



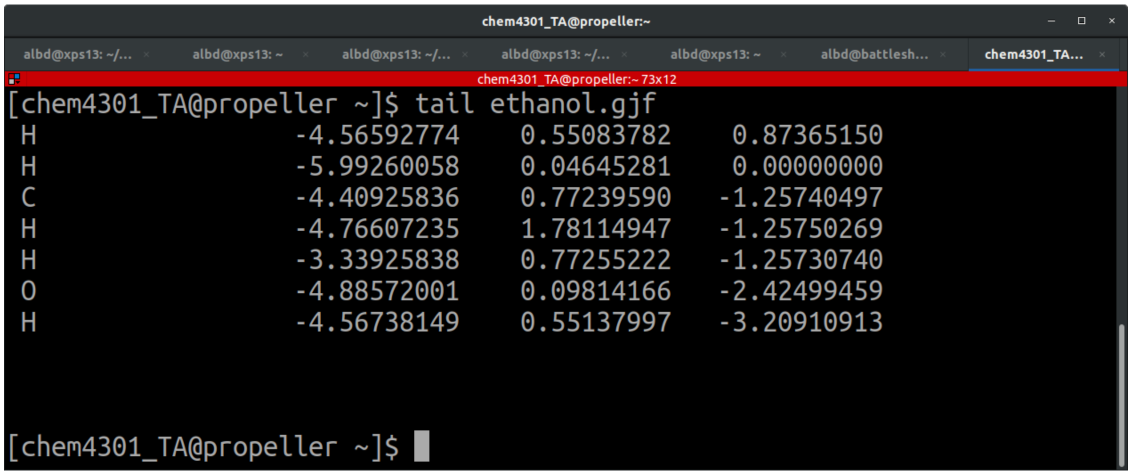
```
[chem4301_TA@propeller ~]$ cat ethanol.gjf
#P B3LYP 6-31G* OPT Freq

ethanol-1

0 1
 C                  -4.92260058    0.04643963    0.00000000
 H                  -4.56594615   -0.96237037    0.00000000
 H                  -4.56592774    0.55083782    0.87365150
 H                  -5.99260058    0.04645281    0.00000000
 C                  -4.40925836    0.77239590   -1.25740497
 H                  -4.76607235    1.78114947   -1.25750269
 H                  -3.33925838    0.77255222   -1.25730740
 O                  -4.88572001    0.09814166   -2.42499459
 H                  -4.56738149    0.55137997   -3.20910913


[chem4301_TA@propeller ~]$ 
```

Figure 19: The `cat` command result.

2.8 The `head` and `tail` commands

Next is the `tail` command, and the similar command of `head`. The `tail` command has the syntax of `tail [option(s)] file_name`, and by default displays the last 10 lines of a file. Similarly, `head` displays the first 10 lines. An example of the `tail` command and its output can be seen in Fig. 20.

One common option used for both `tail` and `head` is the `-n` flag, which allows you to change the default of the last 10 lines of the file. For example, using `tail -n 5 file_name` will display the last 5 lines of the file. If you add a plus sign, `tail -n +5 file_name` will display all lines in the file, except for the first 5.

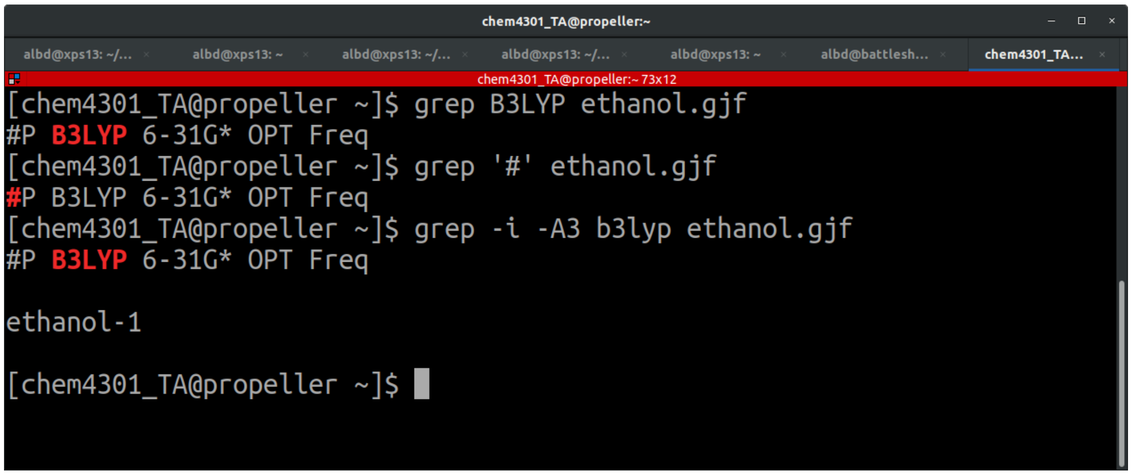


```
[chem4301_TA@propeller ~]$ tail ethanol.gjf
 H                  -4.56592774    0.55083782    0.87365150
 H                  -5.99260058    0.04645281    0.00000000
 C                  -4.40925836    0.77239590   -1.25740497
 H                  -4.76607235    1.78114947   -1.25750269
 H                  -3.33925838    0.77255222   -1.25730740
 O                  -4.88572001    0.09814166   -2.42499459
 H                  -4.56738149    0.55137997   -3.20910913



[chem4301_TA@propeller ~]$ 
```

Figure 20: The `tail` command result.

2.9 The `grep` command

The `grep` or `g/re/p` (globally search for a regular expression and print matching lines) is your search command for text patterns that match those specified by the user. It is one of the most powerful commands in your arsenal for this course. The `grep` command will be used to find specific lines of text within your files and has the syntax, `grep [option(s)] pattern file(s)`. For example, if you were looking for the level of theory in your gaussian input file `ethanol.gjf` and didn't want to open the file with `less` or `more`, `grep` could find this line for you, as can be seen in Fig. 21.



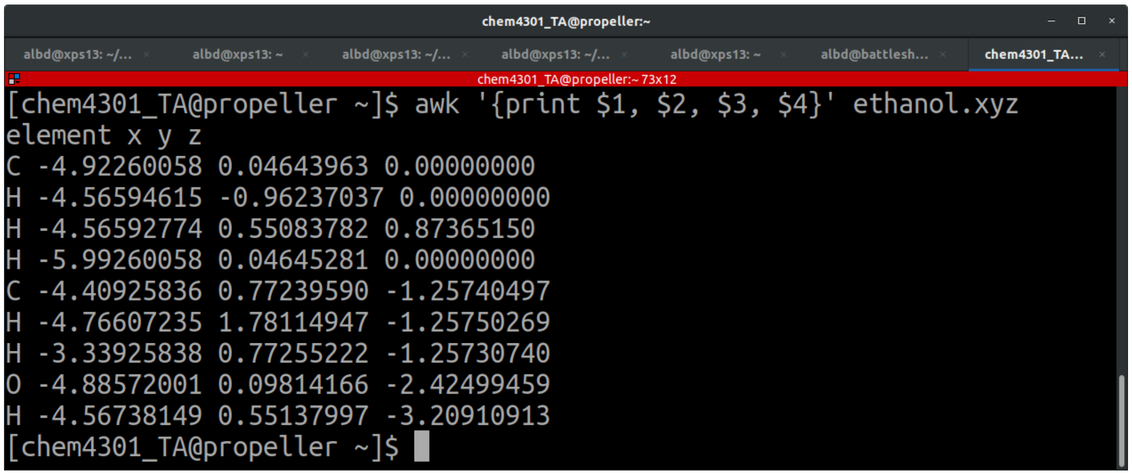
```
[chem4301_TA@propeller ~]$ grep B3LYP ethanol.gjf
#P B3LYP 6-31G* OPT Freq
[chem4301_TA@propeller ~]$ grep '#' ethanol.gjf
#P B3LYP 6-31G* OPT Freq
[chem4301_TA@propeller ~]$ grep -i -A3 b3lyp ethanol.gjf
#P B3LYP 6-31G* OPT Freq

ethanol-1

[chem4301_TA@propeller ~]$ 
```

Figure 21: The `grep` command result.

For certain special characters, or for spaces, the `grep` pattern must be surrounded by delimiters, such as `'bleh is blah'` or `"bleh is blah"`, as can be seen for the second example in Fig. 21. When `grep` is used with its default options, the pattern must match exactly (including spaces) and it is case sensitive. Some useful flags for `grep` are the `-i` flag, which ignores the case sensitivity, as well as the `-A#` flag, which prints the next number of lines after the pattern match; examples of both are shown in Fig. 21. `grep` will be widely used within this course and is an extremely valuable tool in overall competency in `GNU/Linux`.

2.10 The awk command

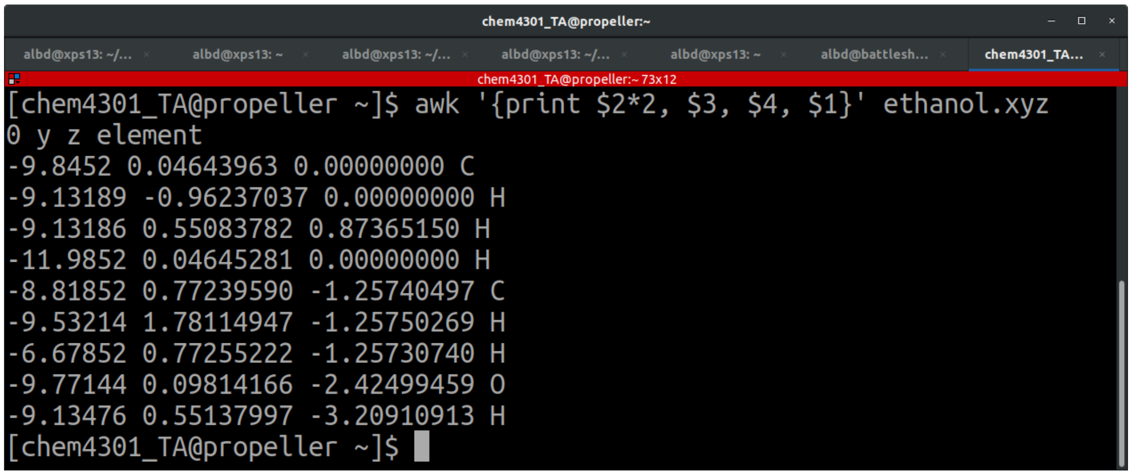
The awk command is another important basic tool within bash. awk is effectively its own programming language and was named after its authors: Aho, Weinberger, and Kernighan. It has the basic syntax of awk [option(s)] [script] file_name, where script can be a simple or more complex awk script. While awk can be used to do a wide range of things, we will primarily be using it for manipulation of data fields, such as output values from Gaussian. A small example of the power of awk is shown in Fig. 22, where the command awk 'print $1, $2, $3, $4' ethanol.xyz prints out columns 1–4 of the file ethanol.xyz.



```
[chem4301_TA@propeller ~]$ awk '{print $1, $2, $3, $4}' ethanol.xyz
element x y z
C -4.92260058 0.04643963 0.00000000
H -4.56594615 -0.96237037 0.00000000
H -4.56592774 0.55083782 0.87365150
H -5.99260058 0.04645281 0.00000000
C -4.40925836 0.77239590 -1.25740497
H -4.76607235 1.78114947 -1.25750269
H -3.33925838 0.77255222 -1.25730740
O -4.88572001 0.09814166 -2.42499459
H -4.56738149 0.55137997 -3.20910913
[chem4301_TA@propeller ~]$
```

Figure 22: The awk command result.

Examples of column manipulation with awk include changing the order of the columns to place the element column last, and multiplying all entries in the x column by a fixed constant, as shown in Fig. 23.



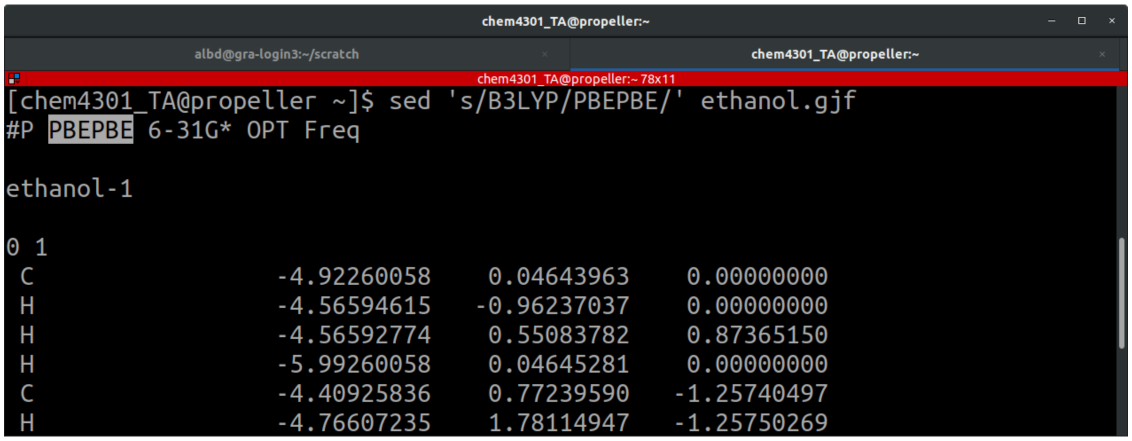
```
[chem4301_TA@propeller ~]$ awk '{print $2*2, $3, $4, $1}' ethanol.xyz
0 y z element
-9.8452 0.04643963 0.00000000 C
-9.13189 -0.96237037 0.00000000 H
-9.13186 0.55083782 0.87365150 H
-11.9852 0.04645281 0.00000000 H
-8.81852 0.77239590 -1.25740497 C
-9.53214 1.78114947 -1.25750269 H
-6.67852 0.77255222 -1.25730740 H
-9.77144 0.09814166 -2.42499459 O
-9.13476 0.55137997 -3.20910913 H
[chem4301_TA@propeller ~]$
```

Figure 23: The awk command examples.

awk can also be used to sum or average the values in a column. These examples only scratch the surface of the utility of awk. A good resource for examples can be found here, although it is by no means exhaustive.

2.11 The sed command

Finally the sed command, or stream editor, can perform a range of functions on files, such as find and replace, insertions, and/or deletions of text. sed has a general syntax of sed [option(s)] [script] file_name. The most common utility of sed is editing a file without opening it in a text editor. An example of the find and replace action of `sed` is shown in Fig. 24, where it is used to find the first string that matches the pattern `B3LYP` in the file `ethanol.gjf` and replace it with `PBEPBE`.



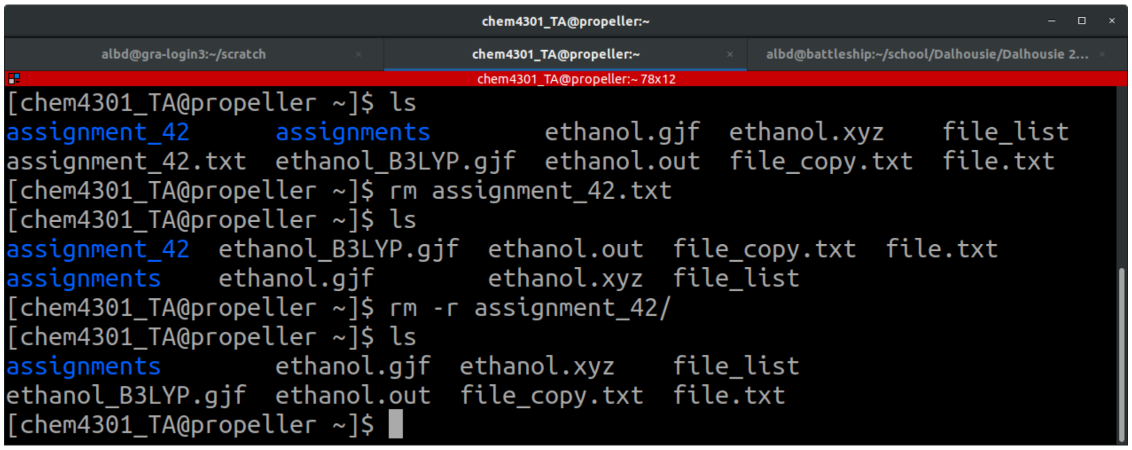
```
[chem4301_TA@propeller ~]$ sed 's/B3LYP/PBEPBE/' ethanol.gjf
#P PBEPBE 6-31G* OPT Freq

ethanol-1

0 1
 C                 -4.92260058    0.04643963    0.00000000
 H                 -4.56594615   -0.96237037    0.00000000
 H                 -4.56592774    0.55083782    0.87365150
 H                 -5.99260058    0.04645281    0.00000000
 C                 -4.40925836    0.77239590   -1.25740497
 H                 -4.76607235    1.78114947   -1.25750269
```

Figure 24: The `sed` command examples.

The syntax for this example is that the `s` specifies that `sed` should substitute the first pattern (`B3LYP`) with the second pattern (`PBEPBE`). This is only done for the first match of that pattern on each line of the file. If you wished to do it for all matches, the above should read `sed 's/B3LYP/PBEPBE/g' ethanol.gjf`, where the `g` at the end requires a global find and replace. Another important flag for the `sed` command is the `-i` flag. Without the `-i` flag, all the `sed` command will do is print to the terminal. The `-i` specifies that the file should be edited in place and the change will be reflected if you were to `cat` the file. Much like `awk`, `sed` can do a wide range of things with its own syntax and language. More resources can be found here.

2.12 The `rm` command

We have saved the `rm` or `remove` command, which is the most dangerous command that you will learn, for last. The syntax is `rm [option(s)] file_name or dir_name`, and this will remove the directory or file in question. The action of `rm` can be seen in Fig. 25 for the file `assignment_42`, and then for the directory `assignment_42/`. The difference for the removal of a directory over a file is the inclusion of `-r` for the directory. In both cases, **BE WARNED** that, once it is removed, it is gone and cannot be recovered (there is no recycle bin). Additionally, `rm -r` will delete all files and sub-directories contained within the parent directory you chose to delete. Be careful when you run `rm` that you are deleting the file you really want gone.

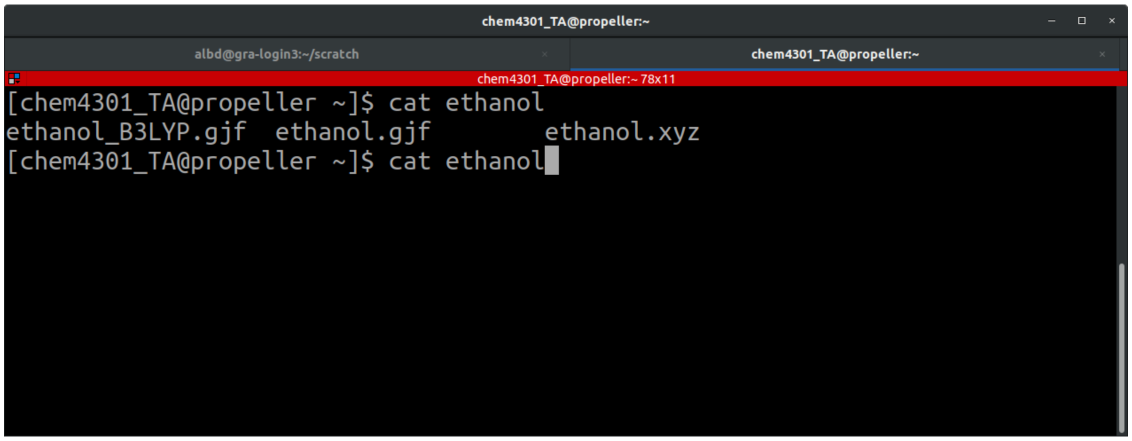

chem4301_TA@propeller:~
albd@gra-login3:~/scratch
chem4301_TA@propeller:~
albd@battleship:~/school/Dalhousie/Dalhousie 2...
chem4301_TA@propeller:~ 78x12

```
[chem4301_TA@propeller ~]$ ls
assignment_42       assignments        ethanol.gjf  ethanol.xyz    file_list
assignment_42.txt  ethanol_B3LYP.gjf  ethanol.out  file_copy.txt  file.txt
[chem4301_TA@propeller ~]$ rm assignment_42.txt
[chem4301_TA@propeller ~]$ ls
assignment_42  ethanol_B3LYP.gjf  ethanol.out  file_copy.txt  file.txt
assignments    ethanol.gjf        ethanol.xyz  file_list
[chem4301_TA@propeller ~]$ rm -r assignment_42/
[chem4301_TA@propeller ~]$ ls
assignments        ethanol.gjf  ethanol.xyz    file_list
ethanol_B3LYP.gjf  ethanol.out  file_copy.txt  file.txt
[chem4301_TA@propeller ~]$ 
```

Figure 25: An example of the `rm` command.

2.13 Auto-complete

The Tab key can be used to auto-complete file, directory, or command names to save time and the effort of typing them in full. For example, if you wish to cat the file from earlier to see if your sed command worked, you could type only cat e and then hit the Tab key. Assuming that there are no other directories or files that begin with e, the bash prompt would auto-complete up to the first divergence, and a second hit of the tab key would list the possible files, as can be seen in Fig. 26.

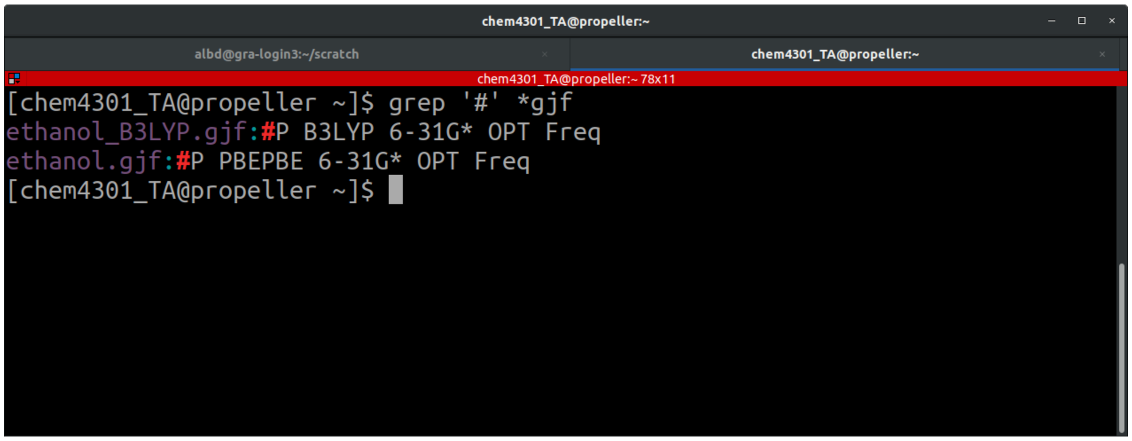


```
[chem4301_TA@propeller ~]$ cat ethanol
ethanol_B3LYP.gjf  ethanol.gjf        ethanol.xyz
[chem4301_TA@propeller ~]$ cat ethanol
```

Figure 26: An example of auto-complete.

2.14 Wildcards

An important feature in bash is the use of wildcards, which are represented by different special characters. The most common standard wildcards are * to represent any number of characters in a string and ? to represent any single character, although there are many others. Wildcards can be extremely useful when chained with other commands, such as ls, cp, and grep (although they are widely used across all commands). For example, if you wanted to grep out the route cards to see the levels of theory used in all Gaussian input files (*.gjf as in Fig. 26), this could be done using grep '#' *gjf, as shown in Fig. 27.



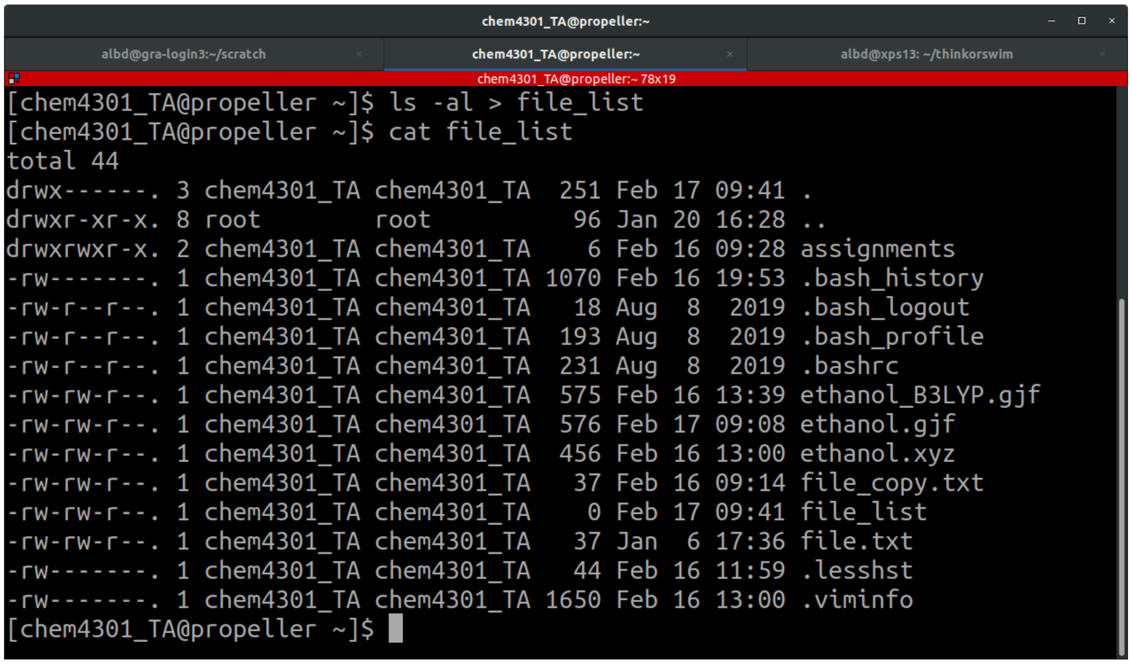
```
[chem4301_TA@propeller ~]$ grep '#' *gjf
ethanol_B3LYP.gjf:#P B3LYP 6-31G* OPT Freq
ethanol.gjf:#P PBEPBE 6-31G* OPT Freq
[chem4301_TA@propeller ~]$
```

Figure 27: An example of a wildcard.

This command can be extremely useful, as it displays both the Gaussian route cards and the corresponding files in which they were found. The use of a wildcard avoids having to specify each file separately. While appearing trivial for one or two files, as you begin to get into 10s-1000s and more files, use of wildcards can become a lifesaver.

2.15 File redirection

Redirection is a feature within Linux that, when executing a command, allows you to change the standard input/output devices. The standard input device is the keyboard, known as `stdin`, and the standard output is the screen (terminal), known as `stdout`. The symbol used for output redirection is the > symbol. It allows us to run commands such as `ls -al > file_list`, as in Fig. 28. Rather than print the output of `ls -la` to the screen, this command redirects it to the file `file_list`.


chem4301_TA@propeller:~
albd@gra-login3:~/scratch
chem4301_TA@propeller:~
albd@xps13: ~/thinkorswim
chem4301_TA@propeller:~ 78x19

```
[chem4301_TA@propeller ~]$ ls -al > file_list
[chem4301_TA@propeller ~]$ cat file_list
total 44
drwx------. 3 chem4301_TA chem4301_TA  251 Feb 17 09:41 .
drwxr-xr-x. 8 root        root          96 Jan 20 16:28 ..
drwxrwxr-x. 2 chem4301_TA chem4301_TA    6 Feb 16 09:28 assignments
-rw-------. 1 chem4301_TA chem4301_TA 1070 Feb 16 19:53 .bash_history
-rw-r--r--. 1 chem4301_TA chem4301_TA   18 Aug  8  2019 .bash_logout
-rw-r--r--. 1 chem4301_TA chem4301_TA  193 Aug  8  2019 .bash_profile
-rw-r--r--. 1 chem4301_TA chem4301_TA  231 Aug  8  2019 .bashrc
-rw-rw-r--. 1 chem4301_TA chem4301_TA  575 Feb 16 13:39 ethanol_B3LYP.gjf
-rw-rw-r--. 1 chem4301_TA chem4301_TA  576 Feb 17 09:08 ethanol.gjf
-rw-rw-r--. 1 chem4301_TA chem4301_TA  456 Feb 16 13:00 ethanol.xyz
-rw-rw-r--. 1 chem4301_TA chem4301_TA   37 Feb 16 09:14 file_copy.txt
-rw-rw-r--. 1 chem4301_TA chem4301_TA    0 Feb 17 09:41 file_list
-rw-rw-r--. 1 chem4301_TA chem4301_TA   37 Jan  6 17:36 file.txt
-rw-------. 1 chem4301_TA chem4301_TA   44 Feb 16 11:59 .lesshst
-rw-------. 1 chem4301_TA chem4301_TA 1650 Feb 16 13:00 .viminfo
[chem4301_TA@propeller ~]$
```

Figure 28: An example of an output redirection.

Redirection can be extremely useful in conjunction with commands such as `grep`, where you might search for the computed energy in a number of Gaussian outputs and want the results listed nicely in a single data file, but more on that later. The other useful output redirection symbol is the append symbol or >>. Rather then overwrite the file (if it already exists), use of >> will just append the output of the command to the bottom of the output file.

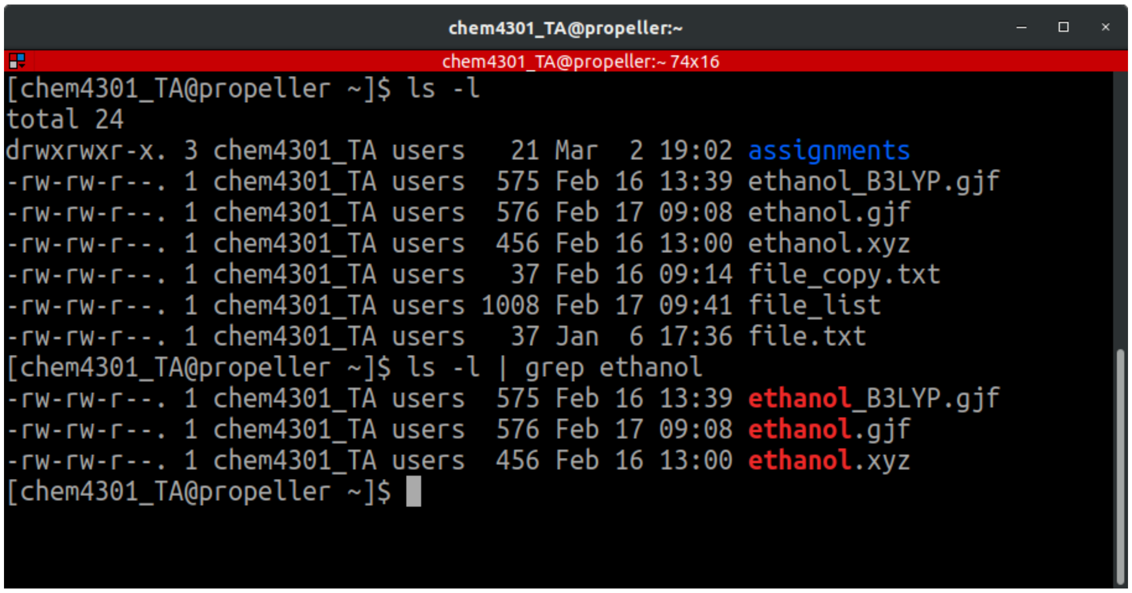
The flip side to this is the input redirection symbol, or <. This will become useful to indicate the input file to be read by Gaussian (or other programs). For example, to execute a Gaussian calculation using our `ethanol.gjf` input file, run `g09 < ethanol.gjf > ethanol.out`.

2.16 Piping

Piping is required for more advanced chaining of commands. A pipe, represented by `|`, allows the output of one command to become the input of another. An example of a pipe can be seen in Fig. 29, where the output of the `ls` command is piped into the `grep` command. Here, `ls -l | grep ethanol` outputs only the listed files whose names contain the string `ethanol`.

2.17 Example scripts

This will stand as your final warning that the terminal is (for the most part) case sensitive so, if you find that scripts or commands are not doing what you expect, the first recommendation will be to check your case, and also your spelling. Another warning is that, if you execute a command, be prepared for it to do exactly what you ask – with great power comes great responsibility, and the ability to delete all of your work. **Consider yourself warned**.

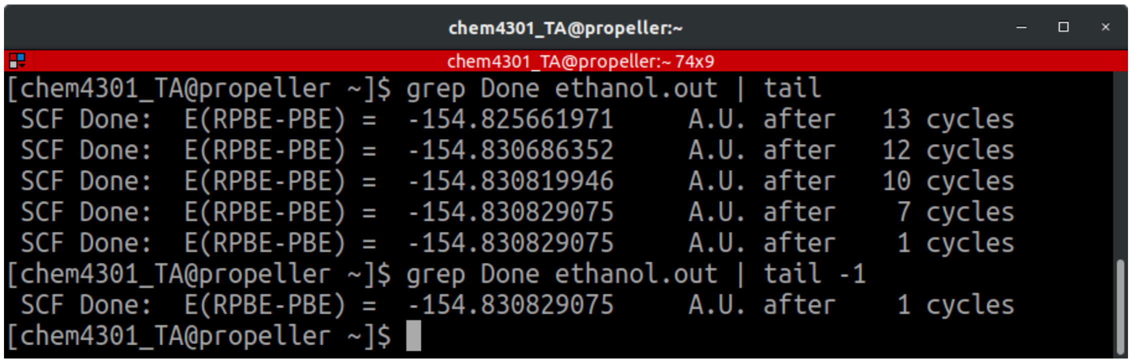


```
[chem4301_TA@propeller ~]$ ls -l
total 24
drwxrwxr-x. 3 chem4301_TA users   21 Mar  2 19:02 assignments
-rw-rw-r--. 1 chem4301_TA users  575 Feb 16 13:39 ethanol_B3LYP.gjf
-rw-rw-r--. 1 chem4301_TA users  576 Feb 17 09:08 ethanol.gjf
-rw-rw-r--. 1 chem4301_TA users  456 Feb 16 13:00 ethanol.xyz
-rw-rw-r--. 1 chem4301_TA users   37 Feb 16 09:14 file_copy.txt
-rw-rw-r--. 1 chem4301_TA users 1008 Feb 17 09:41 file_list
-rw-rw-r--. 1 chem4301_TA users   37 Jan  6 17:36 file.txt
[chem4301_TA@propeller ~]$ ls -l | grep ethanol
-rw-rw-r--. 1 chem4301_TA users  575 Feb 16 13:39 ethanol_B3LYP.gjf
-rw-rw-r--. 1 chem4301_TA users  576 Feb 17 09:08 ethanol.gjf
-rw-rw-r--. 1 chem4301_TA users  456 Feb 16 13:00 ethanol.xyz
[chem4301_TA@propeller ~]$
```

Figure 29: An example of piping.

We conclude this section with a few example scripts that will become extremely useful in this course for the workup of your data. These scripts come in the form of the simple bash commands seen above, now chained together.

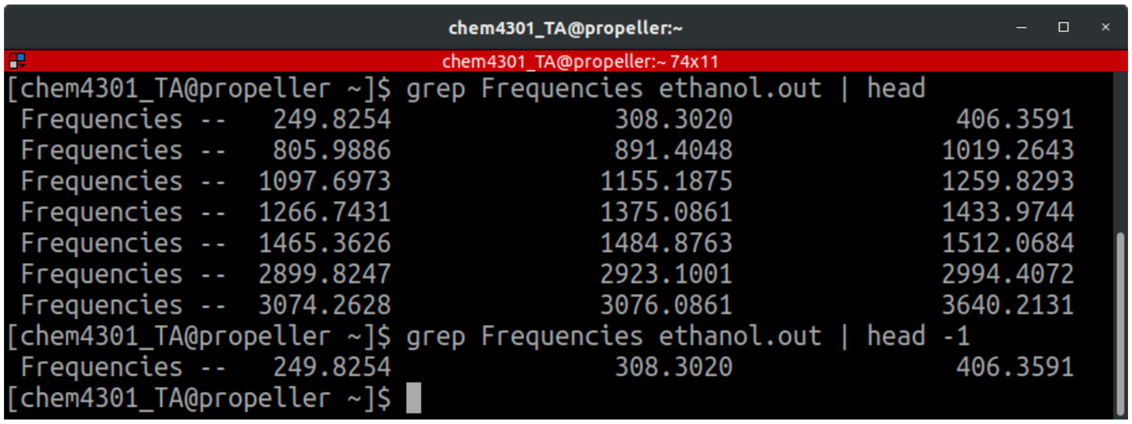
The first example concerns extracting the final electronic energies from Gaussian output files. If you wanted to grep out the final energy from just one file, you could run the command `grep Done ethanol.out | tail -1`, which would have the output seen in Fig. 30. This command outputs the very last occurrence of a line containing the string `Done` from the Gaussian output; assuming all things have gone correctly, this would be your final energy for that job. A more advanced solution to extract the final energies from multiple Gaussian output files simultaneously uses a `for` loop. Although loops are not covered in this tutorial, the command for this would be: `for i in *out ; do grep -H Done $i | tail -1 ; done`.



```
[chem4301_TA@propeller ~]$ grep Done ethanol.out | tail
 SCF Done:  E(RPBE-PBE) =  -154.825661971     A.U. after   13 cycles
 SCF Done:  E(RPBE-PBE) =  -154.830686352     A.U. after   12 cycles
 SCF Done:  E(RPBE-PBE) =  -154.830819946     A.U. after   10 cycles
 SCF Done:  E(RPBE-PBE) =  -154.830829075     A.U. after    7 cycles
 SCF Done:  E(RPBE-PBE) =  -154.830829075     A.U. after    1 cycles
[chem4301_TA@propeller ~]$ grep Done ethanol.out | tail -1
 SCF Done:  E(RPBE-PBE) =  -154.830829075     A.U. after    1 cycles
[chem4301_TA@propeller ~]$
```

Figure 30: An example of finding the output energy for a Gaussian job.

Another important component to the output of many Gaussian jobs is the Frequencies (assuming you have asked to code to calculate them). You will want to verify that there are no negative (or imaginary) modes to ensure that the geometry has optimised to a stable minimum and not a transition state or higher-order saddle point. Since the frequencies are reported in increasing order, this can be achieved by looking at the first occurrence of 'Frequencies' in your Gaussian output. The appropriate command is `grep Frequencies ethanol.out | head -1` and the output can be seen in Fig. 31. A `for` loop can again be employed to iterate over many files with `for i in *out ; do grep -H Frequencies $i | head -1 ; done`.



```
[chem4301_TA@propeller ~]$ grep Frequencies ethanol.out | head
 Frequencies --   249.8254               308.3020               406.3591
 Frequencies --   805.9886               891.4048              1019.2643
 Frequencies --  1097.6973              1155.1875              1259.8293
 Frequencies --  1266.7431              1375.0861              1433.9744
 Frequencies --  1465.3626              1484.8763              1512.0684
 Frequencies --  2899.8247              2923.1001              2994.4072
 Frequencies --  3074.2628              3076.0861              3640.2131
[chem4301_TA@propeller ~]$ grep Frequencies ethanol.out | head -1
 Frequencies --   249.8254               308.3020               406.3591
[chem4301_TA@propeller ~]$ 
```

Figure 31: An example of finding the Frequencies for a Gaussian job.

Finally, there are other quantities that you will wish to extract from your Gaussian outputs, which can be found through modifications of the above commands. One example is the thermal correction to Enthalpy, which can be found using `grep Enthalpy ethanol.out`. Learning the appropriate search strings for Gaussian output files is critical to fast and efficient processing of the generated data.

3 Text Editors

Text editors are the bread and butter of any command line interface. Even before the console wars, rabid fanboys have existed for their specific text editor. A few of note that will be discussed here are `vi` (actually `VIM`, or `Vi IMproved`, the clone of `vi` that we will be using), `nano`, and `Notepad++` (for windows users). Other common ones exist, such as `emacs`, although they require an inordinate amount of complex hand motions and gestures, or extra and dislocatable fingers, so they will not be discussed here. Feel free to use whichever text editor you wish (Word is not a text editor). Just be warned that, if you use one which is not discussed in this document, help may come in the form of "I don't know, google it" (Here be dragons).

3.1 `vi`

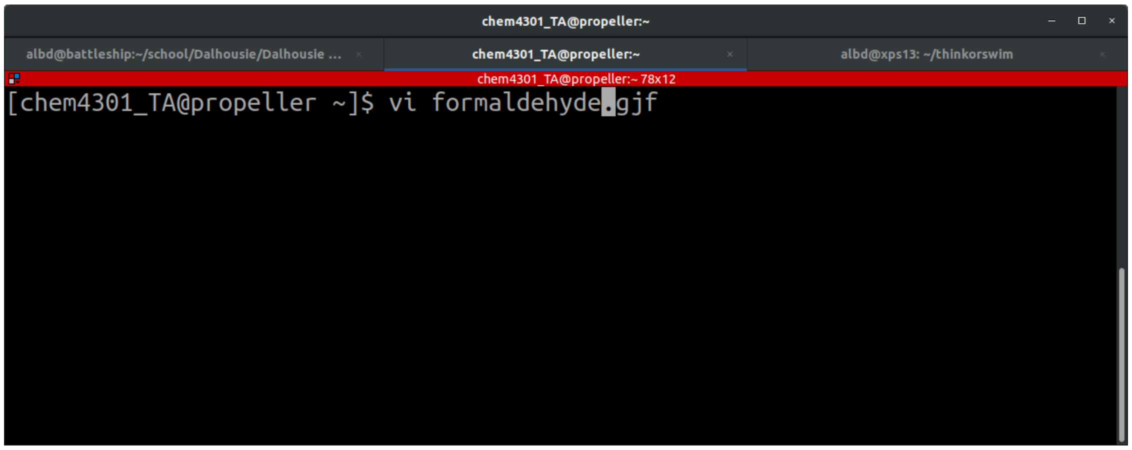
`VIM`, referred to as simply `vi` going forward, is the text editor with which the authors have the most experience and, therefore, it will be the preferred choice for editing files. `vi` is an extremely configurable and efficient text editor. It is the default editor on most UNIX and UNIX-derived systems (OSX and many GNU/Linux systems) and can also be downloaded for Windows and many other operating systems (even the Amiga!).

To open a new or existing file with `vi`, type `vi file_name`. For example, if you wanted to write a Gaussian input file for the compound formaldehyde, you would first open a new file by typing `vi formaldehyde.gjf` and then pressing `Return`, as in Fig. 32.

`vi` has three fundamental modes: normal mode, visual mode and insert mode. Most of your time using `vi` will be spent in either normal mode or insert mode. Insert mode is the easiest of the modes to understand and can be accessed by pressing `i` after opening a file. It allows you to start inserting text at the location of the cursor. Once you enter insert mode, you will see `-- INSERT --` at the bottom of the screen, as in Fig. 33. In the case of our file `formaldehyde.gjf`, if we wanted to write the first line of the input, we would first press `i` and then type the route card `#p B3LYP 6-31G* OPT Freq`. This is also shown in Fig. 33.

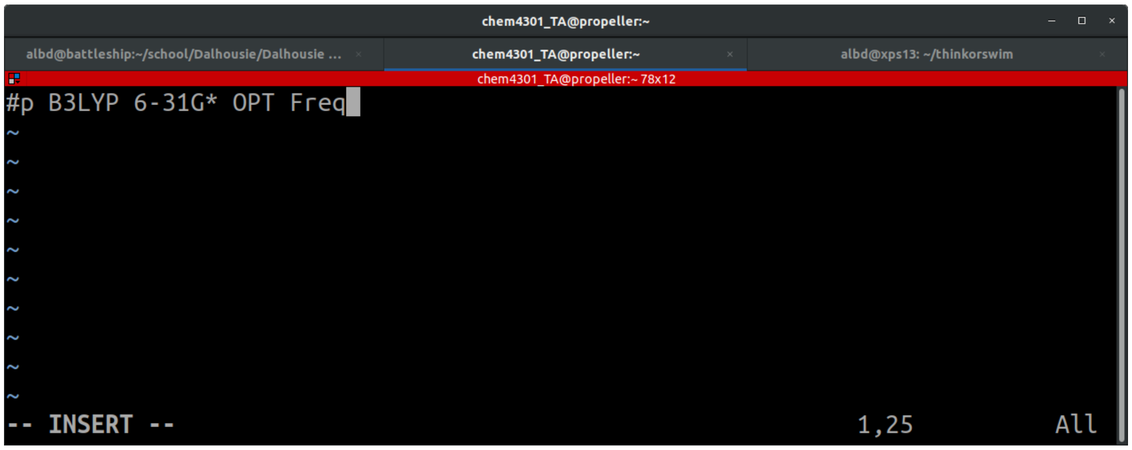
To exit insert mode, hit `escape`. This will return you to command mode, which is where the real power of `vi` lies. In command mode, you can navigate across the previously written text, run commands to search, replace, delete, and a whole host of other things, through `vi`'s own pseudo programming language. Navigation can be achieved either by using the letters `h,j,k,l` for `left, down, up, right`, respectively, or by using the arrow keys.

The next most important commands for `vi` are to save and exit. In command mode, type `:w` and hit `Enter` to save your file, as in Fig. 34. You will then see the name of your file appear at the bottom left of


chem4301_TA@propeller:~
albd@battleship:~/school/Dalhousie/Dalhousie ...
chem4301_TA@propeller:~
albd@xps13: ~/thinkorswim
chem4301_TA@propeller:~ 78x12

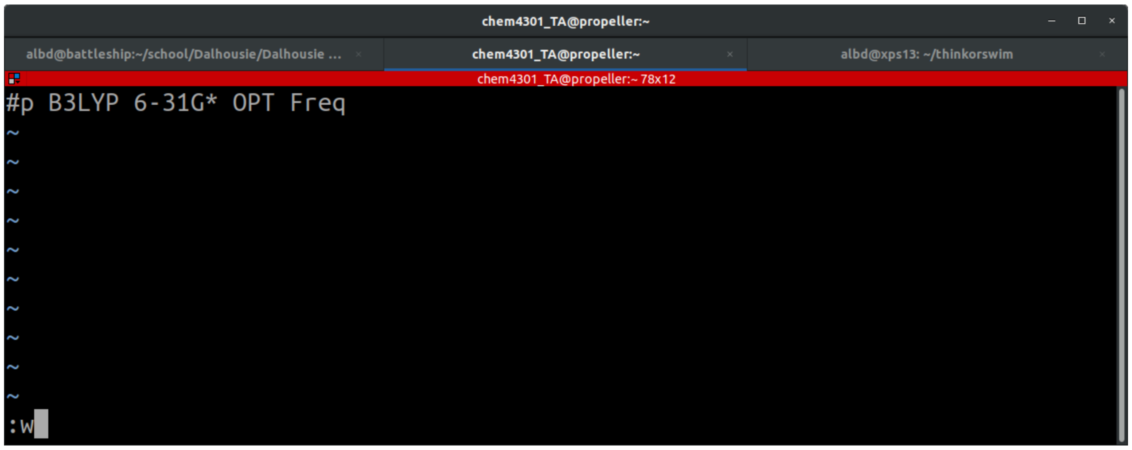
```
[chem4301_TA@propeller ~]$ vi formaldehyde.gjf
```

Figure 32: Creating and opening a file with `vi`.


chem4301_TA@propeller:~
albd@battleship:~/school/Dalhousie/Dalhousie ...
chem4301_TA@propeller:~
albd@xps13: ~/thinkorswim
chem4301_TA@propeller:~ 78x12

```
#p B3LYP 6-31G* OPT Freq
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                              1,25          All
```

Figure 33: `vi` insert mode.


chem4301_TA@propeller:~
albd@battleship:~/school/Dalhousie/Dalhousie ...
chem4301_TA@propeller:~
albd@xps13: ~/thinkorswim
chem4301_TA@propeller:~ 78x12

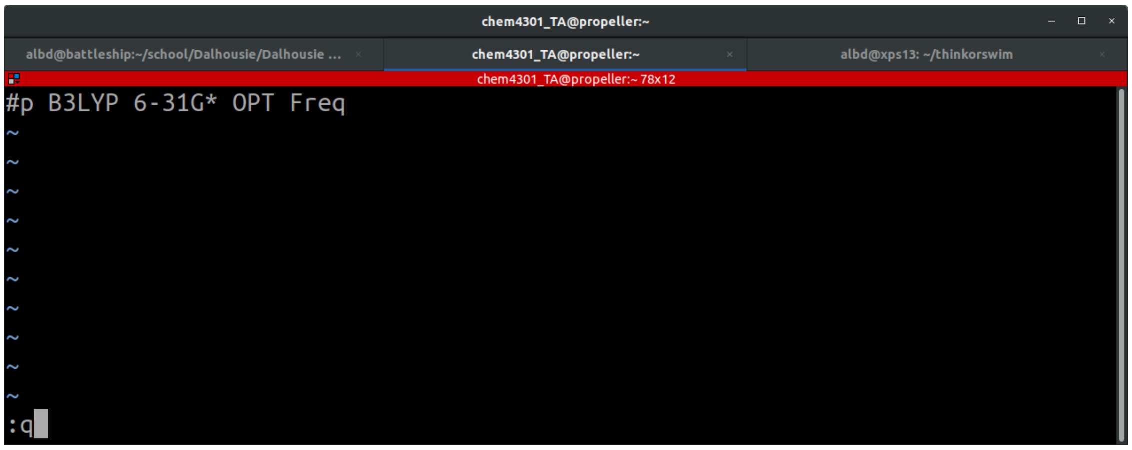
```
#p B3LYP 6-31G* OPT Freq
~
~
~
~
~
~
~
~
~
~
:w
```

Figure 34: `vi` save file.

the `vi` window, along with the numbers of lines and characters in the file and confirmation of the write.

Once the file has been written or saved, one can exit or keep working. There are a few different ways to exit `vi`; all require being in command mode. The first method is to type `:q`, followed by `Enter`, as in Fig. 35.

Upon quitting, you will be returned to the command line of the terminal. You can also chain together the save and quit commands using `:wq`, which is an amalgamation of the two above commands. An alternative command to simultaneously save and quit is `ZZ`. Finally, the combination of `:q!` quits and abandons all changes to your file.


chem4301_TA@propeller:~
albd@battleship:~/school/Dalhousie/Dalhousie ...
chem4301_TA@propeller:~
albd@xps13: ~/thinkorswim
chem4301_TA@propeller:~ 78x12

```
#p B3LYP 6-31G* OPT Freq
~
~
~
~
~
~
~
~
~
~
:q
```

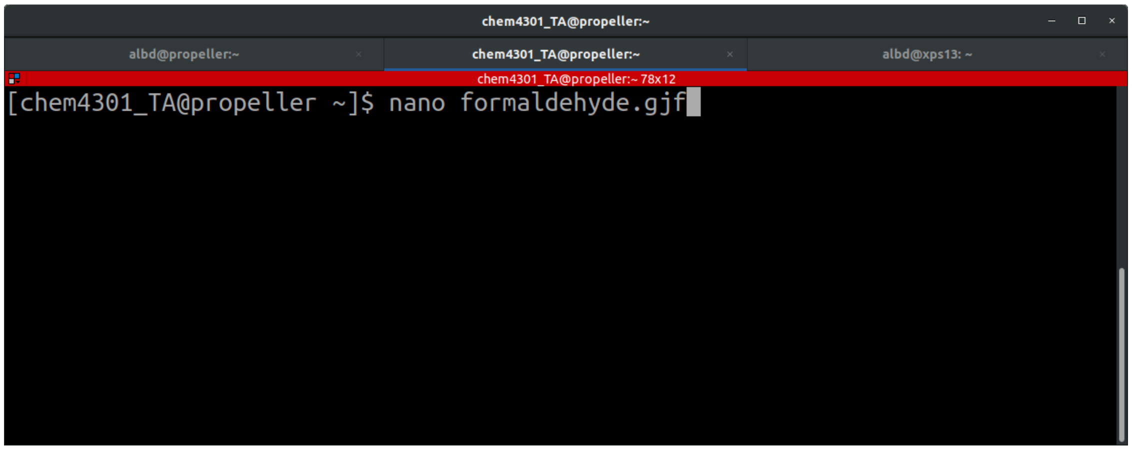
Figure 35: `vi` quit or exit.

Knowledge of far more than the above commands is required before you can unlock the power of `vi`. In an effort to not reinvent the wheel, we recommend that you run through the brief vimtutor tutorial, which can be accessed on the web, or in the terminal by the `vimtutor` command. `vimtutor` goes through key things like copying and pasting text, undo, redo, visual modes, searching, and much more (all surprisingly quickly). Another helpful resource is a `vi` cheat sheet and we recommend you keep one at hand, either on screen or in print. An example can be found here, although many more can be found through google.

As final note, similar to the bash command line, `vi` **does not hold your hand** and will delete, overwrite, or quit without saving if you ask. You, the user, are considered to be the expert, so be careful.

3.2 nano

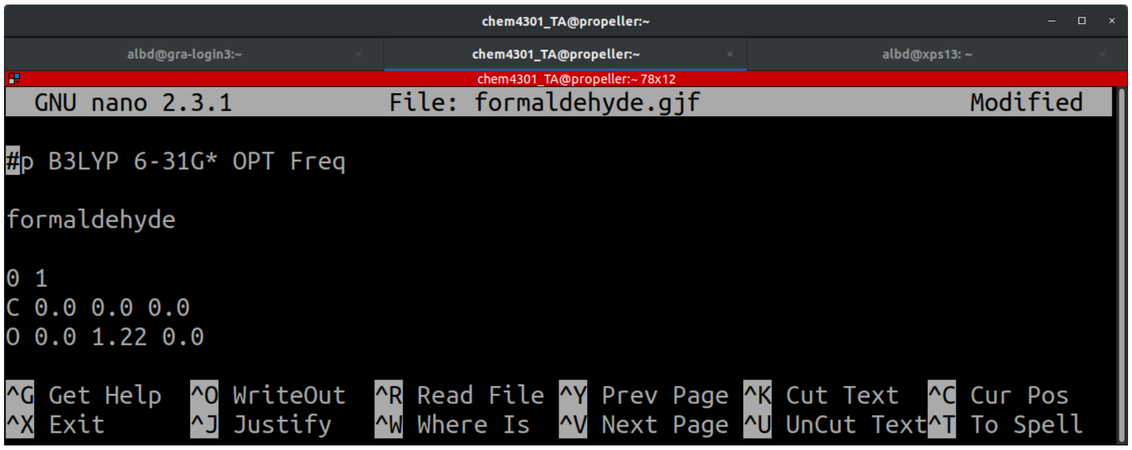
Another common text editor is `GNU nano`. While `nano` is not always installed by default with every Linux distribution, it is far simpler than `vi` or `emacs` as it displays the basic commands to you at all times. To open either a new or existing file with `nano`, use the command `nano file_name`, as shown for our file `formaldehyde.gjf` in Fig. 36.



```
[chem4301_TA@propeller ~]$ nano formaldehyde.gjf
```

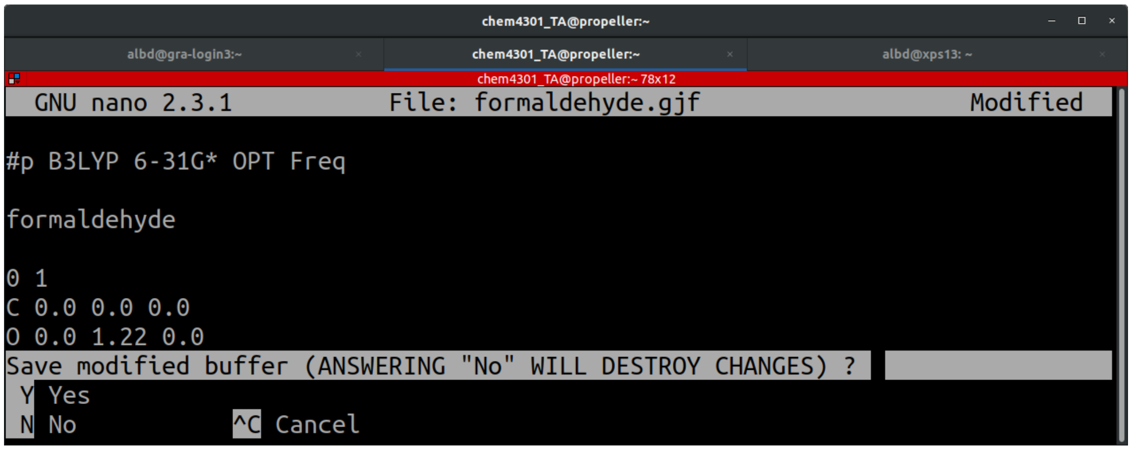
Figure 36: `nano` opening files.

Unlike vi, nano does not have separate command and insert modes. You can start writing immediately after opening the program, with navigation being done by the arrow keys. After you have finished editing the file, you can save it and exit nano using the ^X or ctrl-x command seen at the bottom of Fig. 37. You will then be prompted to either choose Y to save the changes or N to discard them, as shown in Fig. 38. After this you will be asked to confirm the file name, and then the program will exit.



```
  GNU nano 2.3.1            File: formaldehyde.gjf                    Modified

#p B3LYP 6-31G* OPT Freq

formaldehyde

0 1
C 0.0 0.0 0.0
O 0.0 1.22 0.0

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

Figure 37: nano writing to files.



```
  GNU nano 2.3.1            File: formaldehyde.gjf                    Modified

#p B3LYP 6-31G* OPT Freq

formaldehyde

0 1
C 0.0 0.0 0.0
O 0.0 1.22 0.0
Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?
 Y Yes
 N No           ^C Cancel
```

Figure 38: nano save and exiting.

nano is a very simple, yet efficient, text editor. Generally is a fair bit easier to use than vi for beginners, although it lacks the more robust and useful features of vi. It is recommended that you use vi over nano once you become comfortable with Linux, although this is not required.

3.3 Notepad++

Notepad++ is a free and open-source program distributed on the Windows platform and can be downloaded here. It gives you a fair bit more freedom than the basic Notepad program (should you choose to not download and install vi), and avoids the issues that are present with word processors (looking at you Microsoft word). Notepad++ operates in the same manner as many other windows-based programs with options of save, open, copy, paste, etc. Notepad++ is recommended for Windows users as you **cannot** use a program like Microsoft Word to generate your input files or read your output files in this course because it is a word processor and not a text editor. However, other text editors such as vi are recommended as, should you need to edit anything on the server itself, Notepad++ will not be available to you.

4 Installing Avogadro

Avogadro is free software that will help you to build and visualise molecules, which will be required for the course. Upon opening the program, the use and navigation of Avogadro will be the same for all OSes.

4.1 Windows

The Avogadro program executable can be downloaded from here.

4.2 OSX

For OSX, you can download the source files from the Avogadro website and follow the instructions here.

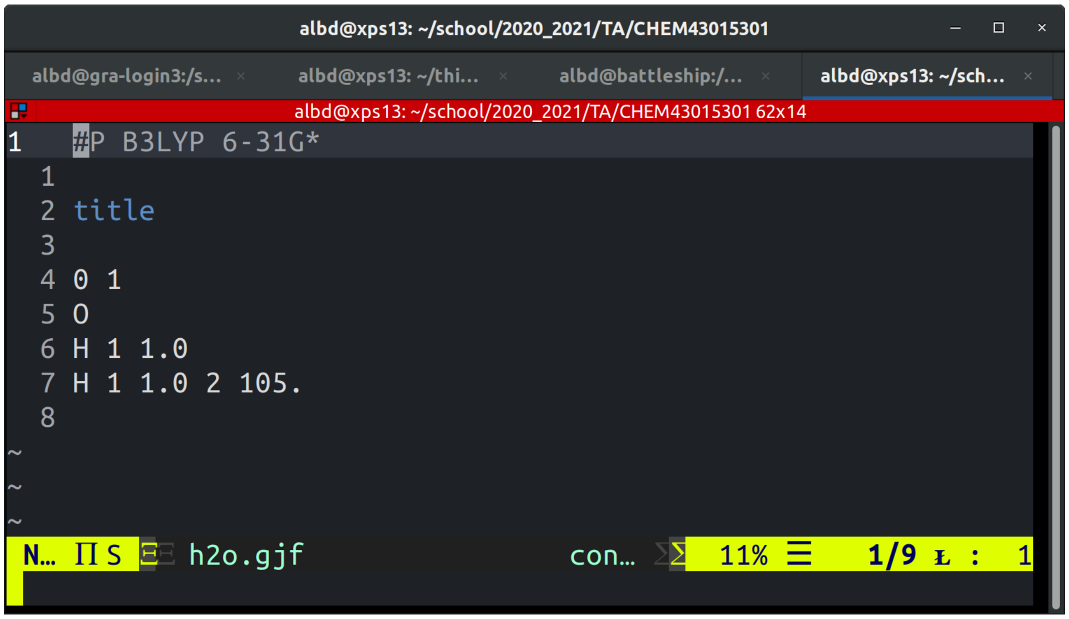
Alternatively, you can open the terminal and first install brew via `/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"`. You should then be able to run the commands here to install Avogadro.

4.3 Ubuntu/debian based versions of linux

To install, run the command: `sudo apt install avogadro`. To open avogadro, use `avogadro` or `avogadro file_name`.

4.4 Avogadro Use

Upon installation of Avogadro, it can be accessed through the normal route of clicking on the executable. With Linux, you can also run the program from the terminal by typing `avogadro` or `avogadro file_name`, where `file_name` is either a Gaussian output file (`.out`), or a file containing the geometry of a molecule in Cartesian coordinates (`.xyz`). Unfortunately, Avogadro does not open standard Gaussian input files, so we must convert to a `.xyz` file. Starting with the example of a single water molecule, as in Fig. 39, we must convert our Gaussian input file and z-matrix into a format that can be read by Avogadro.

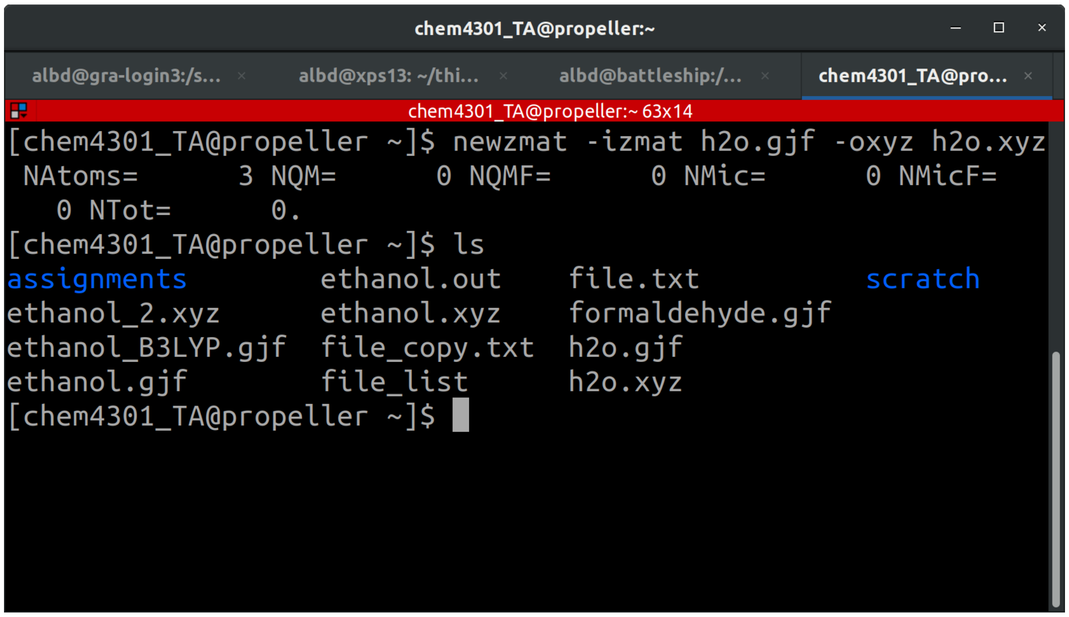

albd@xps13: ~/school/2020_2021/TA/CHEM43015301
albd@gra-login3:/s...
albd@xps13: ~/thi...
albd@battleship:/...
albd@xps13: ~/sch...
albd@xps13: ~/school/2020_2021/TA/CHEM43015301 62x14
N... S h2o.gjf con... 11% 1/9 : 1

```
#P B3LYP 6-31G*

title

0 1
O
H 1 1.0
H 1 1.0 2 105.

```

Figure 39: Example of water Gaussian input file.

We first transfer our `h2o.gjf` file to propeller using the `scp` command learned earlier. Once the file has been transferred, we must run the `newzmat` utility packaged with Gaussian, which has the ability to convert any Gaussian input to a `.xyz` file. Newzmat can be run using the command `newzmat -izmat h2o.gjf`

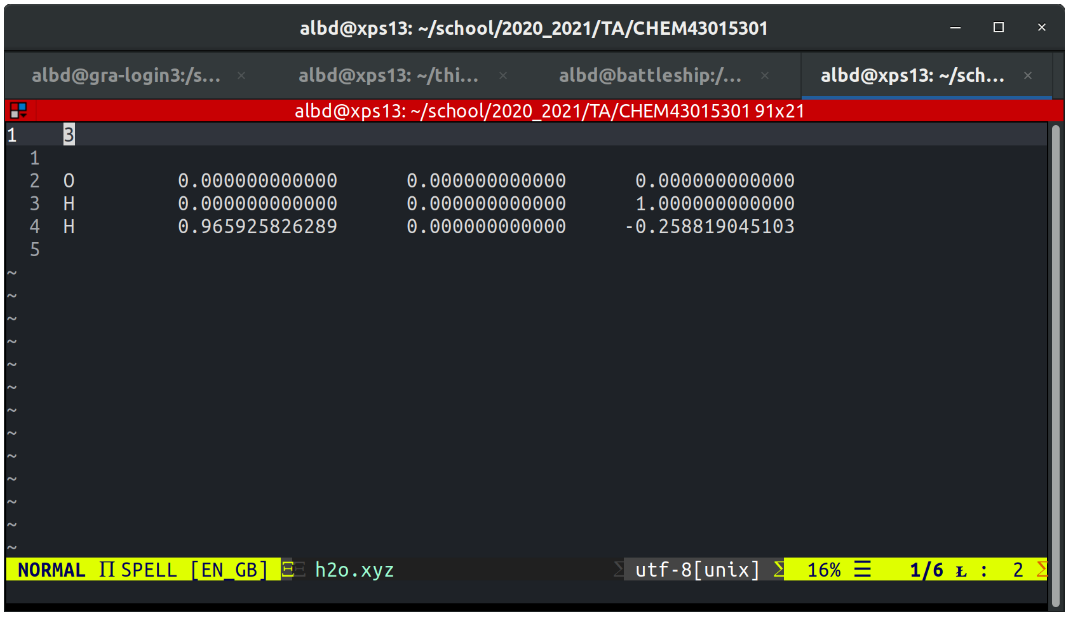
`-oxyz h2o.xyz`, where `-iXXXX` is the input file type and `-oXXX` is the output file type. This command and its output can be seen in Fig. 40.



```
[chem4301_TA@propeller ~]$ newzmat -izmat h2o.gjf -oxyz h2o.xyz
 NAtoms=      3 NQM=       0 NQMF=       0 NMic=       0 NMicF=
   0 NTot=      0.
[chem4301_TA@propeller ~]$ ls
assignments        ethanol.out    file.txt          scratch
ethanol_2.xyz      ethanol.xyz    formaldehyde.gjf
ethanol_B3LYP.gjf  file_copy.txt  h2o.gjf
ethanol.gjf        file_list      h2o.xyz
[chem4301_TA@propeller ~]$ 
```

Figure 40: Example of running newzmat to convert a Gaussian input file to xyz.

The next step is to edit the `.xyz` to insert the total number of atoms within the molecule on the very first line of the file, followed by a blank line, as shown in Fig. 41.



```
3

O         0.000000000000      0.000000000000      0.000000000000
H         0.000000000000      0.000000000000      1.000000000000
H         0.965925826289      0.000000000000     -0.258819045103

```

Figure 41: Example of editing the xyz file.

The `.xyz` file should then be retrieved from the server via `scp`. Avogadro can read the `.xyz` file and display the molecule on screen, as in Fig. 42. The position of manipulation tools to move, rotate, add more molecules, etc. is highlighted by the central red box. Much like any other program, once you are happy with your input, you can save and close it using the File menu in the top left of the window.

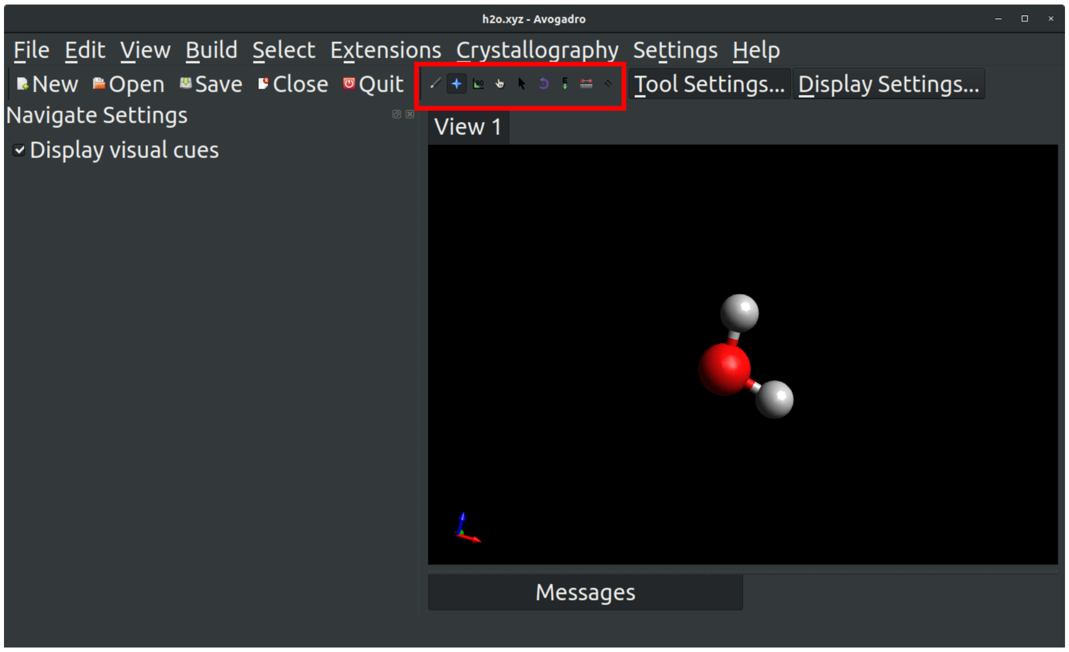

h2o.xyz - Avogadro
File Edit View Build Select Extensions Crystallography Settings Help
New Open Save Close Quit
Tool Settings... Display Settings...
Navigate Settings
Display visual cues
View 1
Messages

Figure 42: Example of visualising water molecule with Avogadro.

Once you have verified that your input structure is reasonable, you can run the original Gaussian input file. If you made any changes to the structure in Avogadro, you should save the new geometry and copy the new xyz coordinates back into your Gaussian input, replacing the original z-matrix. For the first computational assignment in course, it is important that you stick with using a z-matrix, but Cartesian coordinates can be used in the second assignment.

Avogadro does have a basic ability to generate Gaussian input files from the Extensions tab, where you can select the type of calculation, the level of theory, basis set, and much more. However, for production calculations, it is recommended that you write your route card and specify charge and multiplicity by hand rather than using this simplistic tool.

You should be now equipped with the basic tools to complete the computational assignments in the CHEM4301/5301 course and, more generally, for beginning to use Gaussian, one of the first and most ubiquitous computational chemistry packages for molecular calculations. Good luck and have fun!