Beginner's Guide to Doing Science on the OU Supercomputer

Blaine Mooers, PhD
OUHSC

Andrew Fagg, PhD
OU-Norman

Oklahoma Data Science Workshop, Virtual meeting
10 November 2022

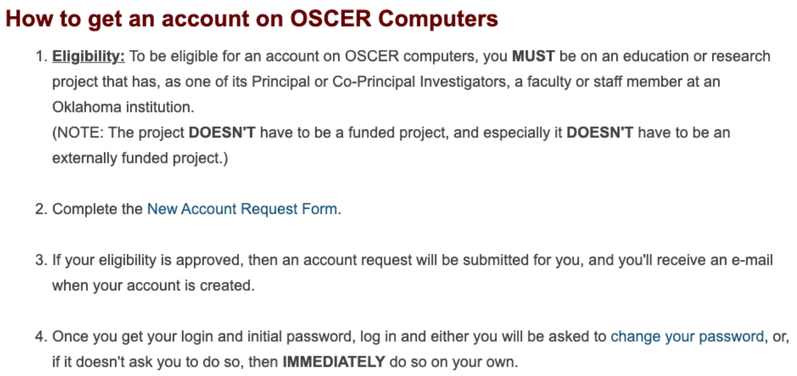
Why use the supercomputer?

- **You need more computing power (10-1000 times more CPUs)**.
- Embarrassingly parallel or truly parallel compute jobs.
- Some Nivida GPUs, more coming.
- Queuing system frees you to do other things.
- Over 660 software packages already installed.
- 100 Petabyte tape archive system (OURRStore).
- 10s TB of hard drive space available (ORRDisk).
- OSCER staff are responsive and helpful.

Why not use Colab instead?

- Limited access to free CPUs and GPUs.
- No domain science software available.
- Spend 5-20 minutes installing domain science software at start of each session, after you have the installation script.
- Need a Google account.
- Logged out due to inactivity.
- Must pay for access to more CPUs or GPUs.
- Must pay for more Google Drive space.
- File I/O on Google Drive is slow.
- Little local help.

Get Account



How to get an account on OSCER Computers

1. **Eligibility:** To be eligible for an account on OSCER computers, you **MUST** be on an education or research project that has, as one of its Principal or Co-Principal Investigators, a faculty or staff member at an Oklahoma institution.
   (NOTE: The project **DOESN'T** have to be a funded project, and especially it **DOESN'T** have to be an externally funded project.)

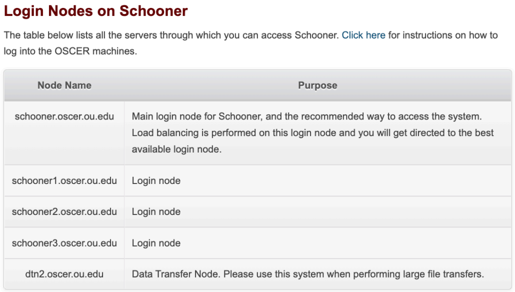
2. Complete the New Account Request Form.

3. If your eligibility is approved, then an account request will be submitted for you, and you'll receive an e-mail when your account is created.

4. Once you get your login and initial password, log in and either you will be asked to change your password, or, if it doesn't ask you to do so, then **IMMEDIATELY** do so on your own.

`https://www.ou.edu/oscer/support/accounts/new_account`

Login Nodes



Login Nodes on Schooner

The table below lists all the servers through which you can access Schooner. Click here for instructions on how to log into the OSCER machines.

Node Name	Purpose
schooner.oscer.ou.edu	Main login node for Schooner, and the recommended way to access the system. Load balancing is performed on this login node and you will get directed to the best available login node.
schooner1.oscer.ou.edu	Login node
schooner2.oscer.ou.edu	Login node
schooner3.oscer.ou.edu	Login node
dtn2.oscer.ou.edu	Data Transfer Node. Please use this system when performing large file transfers.

Enter the following in your terminal.

```
touch .bashAliases
alias schooner='ssh <username>@schooner.oscer.ou.edu'
source ~/.bashAliases # add to .bashrc
schooner
```

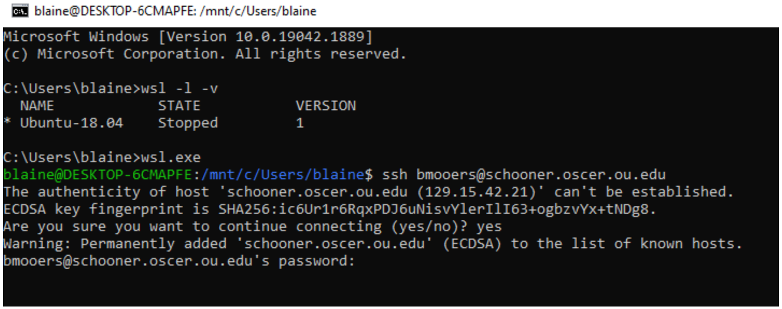
In the future, type ‘schooner’ in the terminal to login.

Four ways to connect to schooner from Windows

- Windows Subsystem for Linux (WSL) (fast and robust; I recommend this option)
- The PuTTy application for Windows (fragile, cannot resize the GUI)
- Use tramp inside Emacs in WSL (not as fast but robust)
- Emacs installed as a native Windows application. (I failed to connect while running Windows10 in a Parallels virtual environment on a Mac. Should work on a PC.)

Windows Subsystem for Linux (wsl)

Contrary to the OSCER website, I had to use schoooer2 rather than schooner to connect.

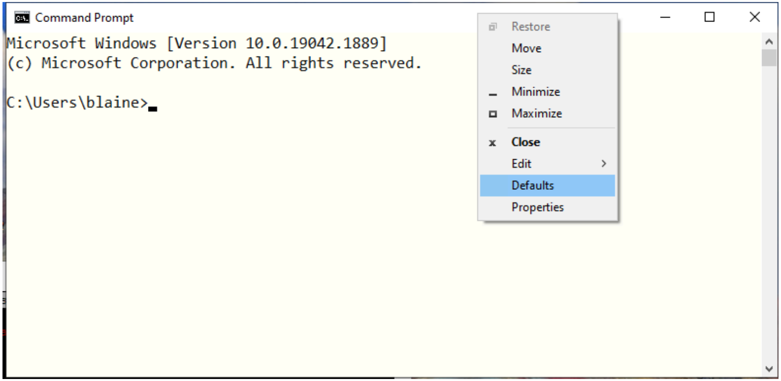


```
blaine@DESKTOP-6CMAPFE: /mnt/c/Users/blaine

Microsoft Windows [Version 10.0.19042.1889]
(c) Microsoft Corporation. All rights reserved.

C:\Users\blaine>wsl -l -v
  NAME            STATE           VERSION
* Ubuntu-18.04    Stopped         1

C:\Users\blaine>wsl.exe
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$ ssh bmooers@schooner.oscer.ou.edu
The authenticity of host 'schooner.oscer.ou.edu (129.15.42.21)' can't be established.
ECDSA key fingerprint is SHA256:ic6Ur1r6RqxPDJ6uNisvYlerIlI63+ogbzvYx+tNDg8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'schooner.oscer.ou.edu' (ECDSA) to the list of known hosts.
bmooers@schooner.oscer.ou.edu's password:
```

Notes: No centered dots appear when entering your password.

```
wsl --install # installs Ubuntu by default
wsl -l -v # check which version is installed
wsl # start up Ubuntu terminal
ssh <username>@schooner2.oscer.ou.edu
```

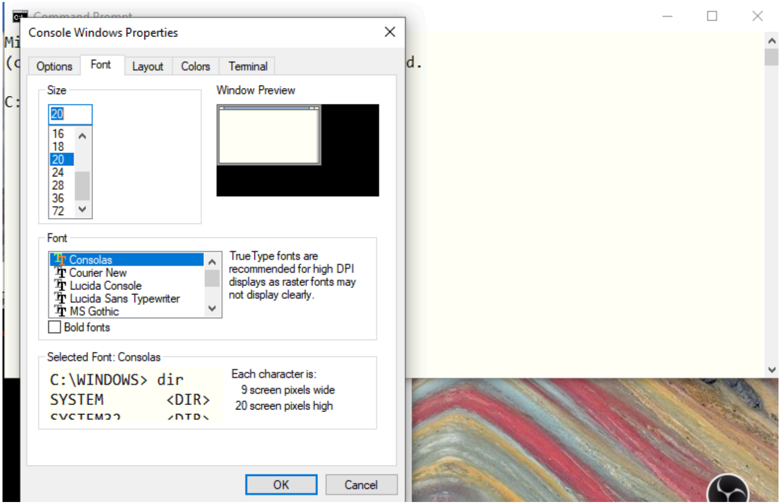
WSL customize (1/2)



```
Microsoft Windows [Version 10.0.19042.1889]
(c) Microsoft Corporation. All rights reserved.

C:\Users\blaine>_
```

Select “Defaults” to edit appearance of terminal window. The appearance of the terminal on your computer will be its appearance when logged into schooner, so customize the terminal’s appearance on your computer to alter your experience on schooner.

WSL customize (2/2)



Console Windows Properties

Options | Font | Layout | Colors | Terminal

Size: 20

16, 18, 20, 24, 28, 36, 72

Window Preview

Font

Consolas
Courier New
Lucida Console
Lucida Sans Typewriter
MS Gothic

TrueType fonts are recommended for high DPI displays as raster fonts may not display clearly.

Bold fonts

Selected Font: Consolas

```
C:\WINDOWS> dir
SYSTEM          <DIR>
```

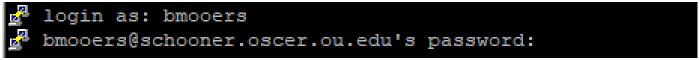
Each character is:
9 screen pixels wide
20 screen pixels high

OK | Cancel

Close terminal and reopen to get change applied.

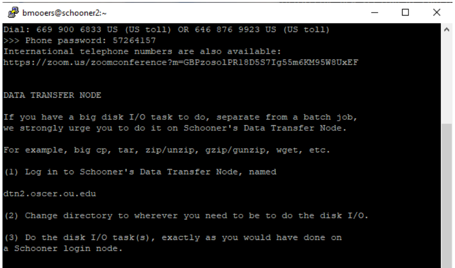
PuTTy for Windows

See
`https://ou.edu/oscer/support/machine_access#windows`
PuTTy is an alternative to using wsl on Windows. Note: allow ten minutes for the configuration.



```
login as: bmooers
bmooers@schooner.oscer.ou.edu's password:
```

Note: no centered dots appear when entering your password



```
Dial: 669 900 6833 US (US toll) OR 646 876 9923 US (US toll)
>>> Phone password: 57264157
International telephone numbers are also available:
https://zoom.us/zoomconference?m=GBPzosolPR18D5S7Ig55m6KM95W8UxEF


DATA TRANSFER NODE

If you have a big disk I/O task to do, separate from a batch job,
we strongly urge you to do it on Schooner's Data Transfer Node.

For example, big cp, tar, zip/unzip, gzip/gunzip, wget, etc.

(1) Log in to Schooner's Data Transfer Node, named

dtn2.oscer.ou.edu

(2) Change directory to wherever you need to be to do the disk I/O.

(3) Do the disk I/O task(s), exactly as you would have done on
a Schooner login node.
```

Access schooner from Emacs

The tramp package for Emacs by Michael Albinus enables ssh'ing into schooner from the minibuffer inside Emacs. Tramp is now built into Emacs.

```
C-x C-f
/ssh:bmooers@schooner2.oscer.ou.edu:~/.bashrc
or
/ssh:bmooers@schooner2.oscer.ou.edu
M-x shell # gives bash-like shell
C-g # abort last Emacs command
Fn F10 File/Close # close current buffer (file)
C-x C-c # exit out of session on schooer.
```

Replace 'ss'h with 'scp' to copy files onto schooner or from schooner.

`https://willschenk.com/articles/2020/tramp_tricks/`

Installing Emacs in WSL version 1

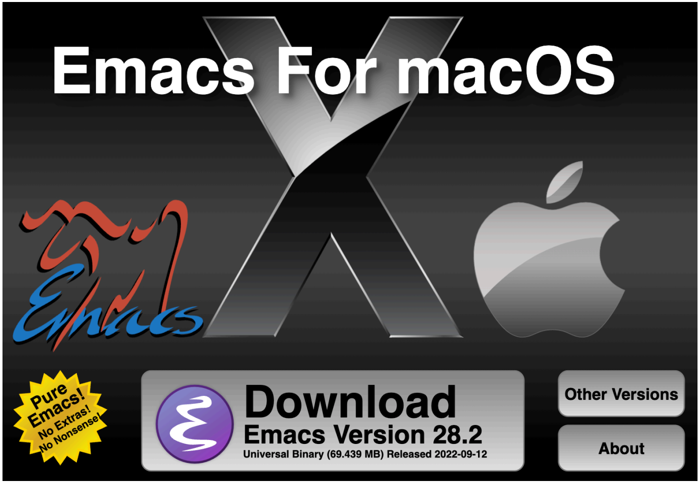
```
sudo apt update && sudo apt upgrade
sudo apt-get install emacs
# install emacs25 in Ubuntu 18
```

To learn how to install more current versions of Emacs on Ubuntu, see `https://learnubuntu.com/install-emacs/`

Installing Emacs executable on Windows 10

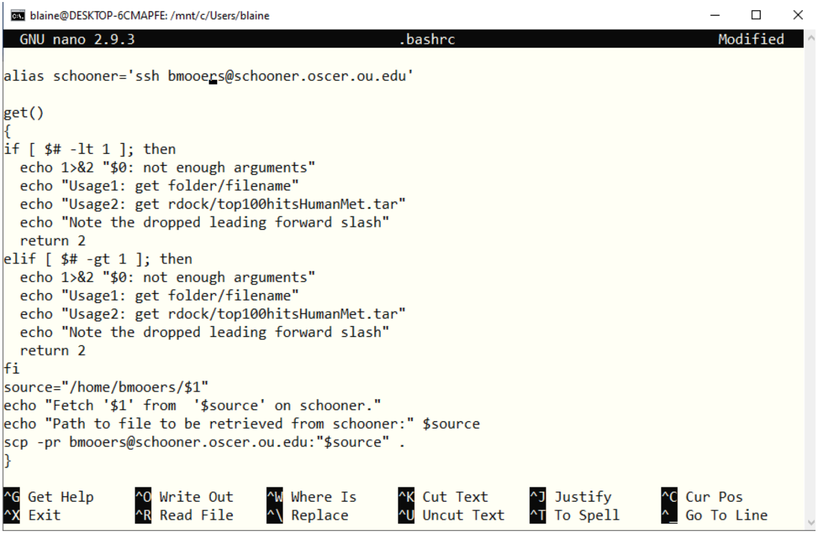
- Download emacs-28.2-installer.exe (latest stable version) from https://www.gnu.org/software/emacs/download.html. (You will have to click on a link that takes you to a nearby site).
- The installer requires a 64-bit version of Windows.
- Run the installer and pin the icon for fast access.

Installing Emacs on MacOS


Emacs For macOS
Pure Emacs! No Extras! No Nonsense!
Download
Emacs Version 28.2
Universal Binary (69.439 MB) Released 2022-09-12
Other Versions
About

`https://emacsformacos.com/`
Other versions –> nightly build == version 29.0.5

Editing .bashrc with nano in WSL



```
blaine@DESKTOP-6CMAPFE: /mnt/c/Users/blaine
  GNU nano 2.9.3                         .bashrc                          Modified

alias schooner='ssh bmooers@schooner.oscer.ou.edu'

get()
{
if [ $# -lt 1 ]; then
  echo 1>&2 "$0: not enough arguments"
  echo "Usage1: get folder/filename"
  echo "Usage2: get rdock/top100hitsHumanMet.tar"
  echo "Note the dropped leading forward slash"
  return 2
elif [ $# -gt 1 ]; then
  echo 1>&2 "$0: not enough arguments"
  echo "Usage1: get folder/filename"
  echo "Usage2: get rdock/top100hitsHumanMet.tar"
  echo "Note the dropped leading forward slash"
  return 2
fi
source="/home/bmooers/$1"
echo "Fetch '$1' from  '$source' on schooner."
echo "Path to file to be retrieved from schooner:" $source
scp -pr bmooers@schooner.oscer.ou.edu:"$source" .
}

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^_ Go To Line
```

Using the schooner alias in WSL

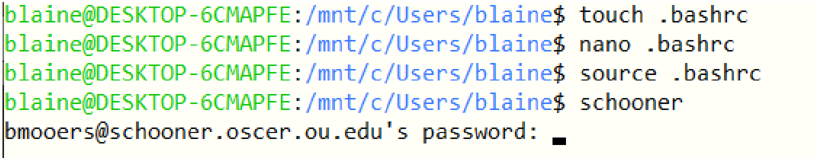
```
alias schooner='ssh bmooers@schooner.oscer.ou.edu'
```

Notes:

No spaces around equal sign in the alias definition.

Nest single and double quotes.

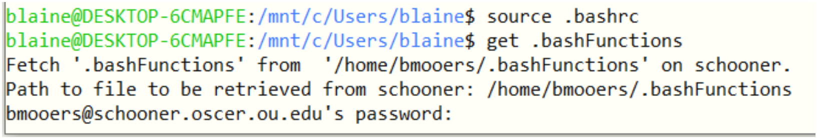
Store alias outside of .bashrc (e.g., in .myaliases)



```
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$ touch .bashrc
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$ nano .bashrc
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$ source .bashrc
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$ schooner
bmooers@schooner.oscer.ou.edu's password:
```

Running the get() function in WSL

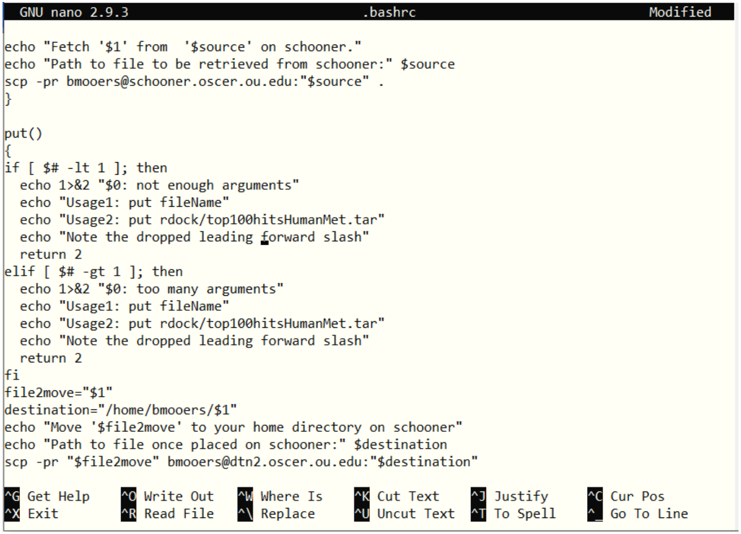
Bash aliases do not take arguments. Instead, use bash functions. I define these inside of a file called bashFunctions. This file is sourced from .bashrc by adding the line "source .bashFunctions" to .bashrc and moving .bashFunctions to your home directory. This file can be found on GitHub: (`https://github.com/MooersLab/bashFunctions4oscer`)



```
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$ source .bashrc
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$ get .bashFunctions
Fetch '.bashFunctions' from  '/home/bmooers/.bashFunctions' on schooner.
Path to file to be retrieved from schooner: /home/bmooers/.bashFunctions
bmooers@schooner.oscer.ou.edu's password:
```

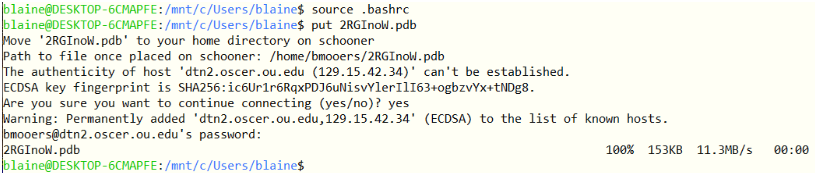
Use Data Transfer Node 2 (dtn2)

Use the Data Transfer Node 2 (dtn2) to move large files onto and off schooner



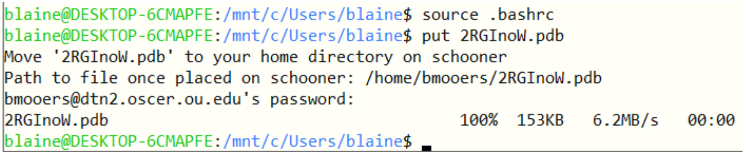
```
GNU nano 2.9.3                         .bashrc                         Modified

echo "Fetch '$1' from  '$source' on schooner."
echo "Path to file to be retrieved from schooner:" $source
scp -pr bmooers@schooner.oscer.ou.edu:"$source" .
}

put()
{
if [ $# -lt 1 ]; then
  echo 1>&2 "$0: not enough arguments"
  echo "Usage1: put fileName"
  echo "Usage2: put rdock/top100hitsHumanMet.tar"
  echo "Note the dropped leading forward slash"
  return 2
elif [ $# -gt 1 ]; then
  echo 1>&2 "$0: too many arguments"
  echo "Usage1: put fileName"
  echo "Usage2: put rdock/top100hitsHumanMet.tar"
  echo "Note the dropped leading forward slash"
  return 2
fi
file2move="$1"
destination="/home/bmooers/$1"
echo "Move '$file2move' to your home directory on schooner"
echo "Path to file once placed on schooner:" $destination
scp -pr "$file2move" bmooers@dtn2.oscer.ou.edu:"$destination"

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^_ Go To Line
```

Using put() function in WSL



```
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$ source .bashrc
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$ put 2RGInoW.pdb
Move '2RGInoW.pdb' to your home directory on schooner
Path to file once placed on schooner: /home/bmooers/2RGInoW.pdb
The authenticity of host 'dtn2.oscer.ou.edu (129.15.42.34)' can't be established.
ECDSA key fingerprint is SHA256:ic6Ur1r6RqxPDJ6uNisvYlerIlI63+ogbzvYx+tNDg8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'dtn2.oscer.ou.edu,129.15.42.34' (ECDSA) to the list of known hosts.
bmooers@dtn2.oscer.ou.edu's password:
2RGInoW.pdb                                                   100%  153KB  11.3MB/s   00:00
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$
```

No authentication prompt on second operation:



```
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$ source .bashrc
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$ put 2RGInoW.pdb
Move '2RGInoW.pdb' to your home directory on schooner
Path to file once placed on schooner: /home/bmooers/2RGInoW.pdb
bmooers@dtn2.oscer.ou.edu's password:
2RGInoW.pdb                                 100%  153KB   6.2MB/s   00:00
blaine@DESKTOP-6CMAPFE:/mnt/c/Users/blaine$
```

tar files before using scp

10,000 file per folder limit. If output is large in volume, write to **/scratch/<username>**.

```
# tar is available on schooner and in WSL
#
# Make tar archive
tar cvf hotResults.tar ./hotResults
# Untar
tar xvf hotResults.tar
```

Limits to working on schooner

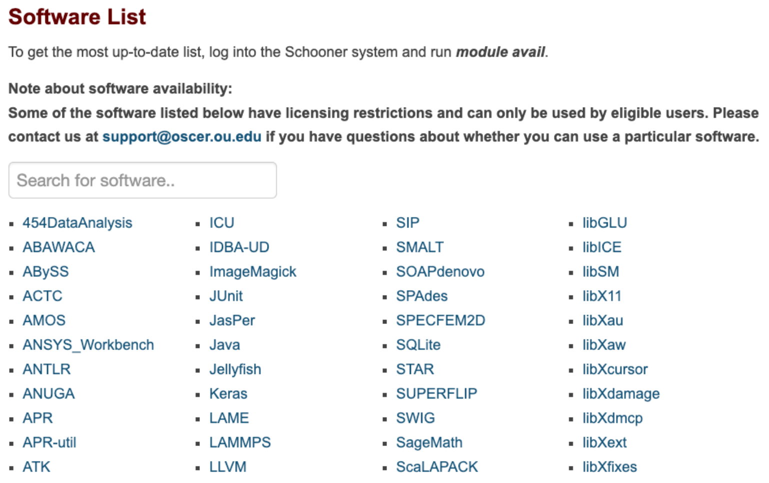
- 20 GB of space in **/home/<username>**.
- 48 hours max of run time per job (a problem for long-running MD simulations).
- Two weeks of lifetime on **/scratch/<username>**.
- Generally, no GUI based programs.
- No GUI based text editors. (Use nano, vim (:wq!), or emacs (C-x C-c, C==control)).
- Limits on web server usage.
- Software packages often outdated.

Software

- Installed already on schooner by OSCER staff
- Install yourself with conda
- Install yourself by other means

Installed Software

Over 660 software packages are already installed by OSCER staff.



Software List

To get the most up-to-date list, log into the Schooner system and run ***module avail***.

Note about software availability:

Some of the software listed below have licensing restrictions and can only be used by eligible users. Please contact us at support@oscer.ou.edu if you have questions about whether you can use a particular software.

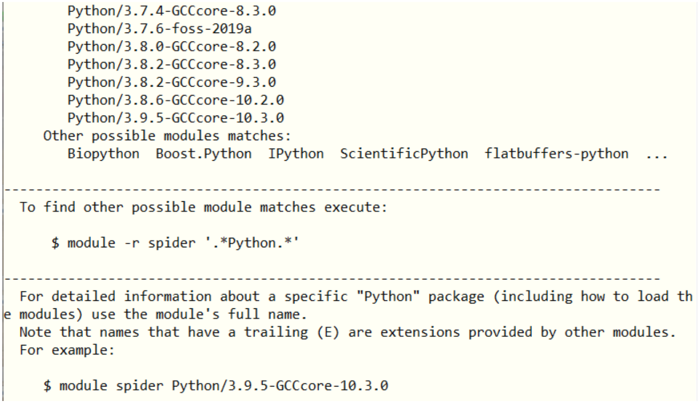
Search for software..

454DataAnalysis	ICU	SIP	libGLU
ABAWACA	IDBA-UD	SMALT	libICE
ABySS	ImageMagick	SOAPdenovo	libSM
ACTC	JUnit	SPAdes	libX11
AMOS	JasPer	SPECFEM2D	libXau
ANSYS_Workbench	Java	SQLite	libXaw
ANTLR	Jellyfish	STAR	libXcursor
ANUGA	Keras	SUPERFLIP	libXdamage
APR	LAME	SWIG	libXdmcp
APR-util	LAMMPS	SageMath	libXext
ATK	LLVM	ScaLAPACK	libXfixes

Searching for Python

The spider option is used to search for software.

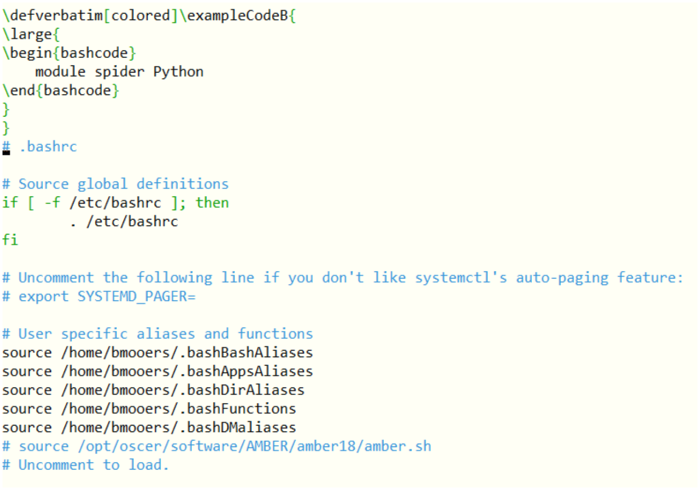
```
module spider Python
```



```
        Python/3.7.4-GCCcore-8.3.0
        Python/3.7.6-foss-2019a
        Python/3.8.0-GCCcore-8.2.0
        Python/3.8.2-GCCcore-8.3.0
        Python/3.8.2-GCCcore-9.3.0
        Python/3.8.6-GCCcore-10.2.0
        Python/3.9.5-GCCcore-10.3.0
     Other possible modules matches:
        Biopython  Boost.Python  IPython  ScientificPython  flatbuffers-python  ...

----------------------------------------------------------------------------------
  To find other possible module matches execute:

      $ module -r spider '.*Python.*'

----------------------------------------------------------------------------------
  For detailed information about a specific "Python" package (including how to load th
e modules) use the module's full name.
  Note that names that have a trailing (E) are extensions provided by other modules.
  For example:

     $ module spider Python/3.9.5-GCCcore-10.3.0
```

.bashrc file in home directory(1/2)

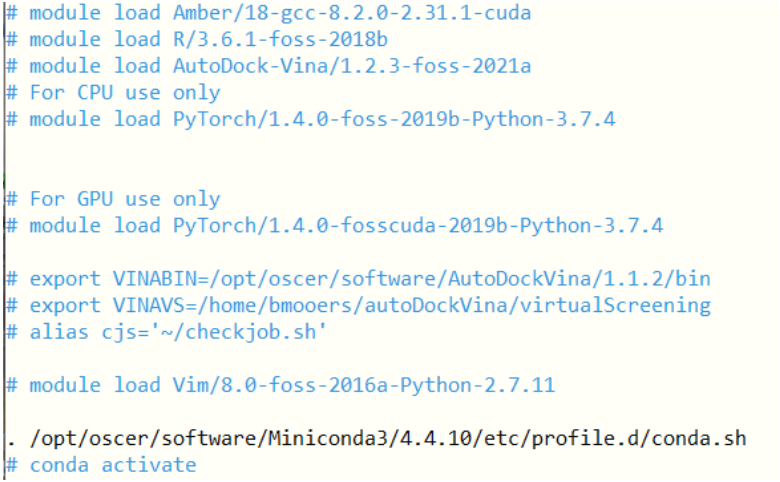
The pound sign in the left margin comments out a line of code.



```
\defverbatim[colored]\exampleCodeB{
\large{
\begin{bashcode}
    module spider Python
\end{bashcode}
}
}
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions
source /home/bmooers/.bashBashAliases
source /home/bmooers/.bashAppsAliases
source /home/bmooers/.bashDirAliases
source /home/bmooers/.bashFunctions
source /home/bmooers/.bashDMaliases
# source /opt/oscer/software/AMBER/amber18/amber.sh
# Uncomment to load.
```

.bashrc (2/2)

Uncomment a ‘module load’ command to load the corresponding software and make it available to your current session on schooner.



```
# module load Amber/18-gcc-8.2.0-2.31.1-cuda
# module load R/3.6.1-foss-2018b
# module load AutoDock-Vina/1.2.3-foss-2021a
# For CPU use only
# module load PyTorch/1.4.0-foss-2019b-Python-3.7.4


# For GPU use only
# module load PyTorch/1.4.0-fosscuda-2019b-Python-3.7.4

# export VINABIN=/opt/oscer/software/AutoDockVina/1.1.2/bin
# export VINAVS=/home/bmooers/autoDockVina/virtualScreening
# alias cjs='~/checkjob.sh'

# module load Vim/8.0-foss-2016a-Python-2.7.11

. /opt/oscer/software/Miniconda3/4.4.10/etc/profile.d/conda.sh
# conda activate
```

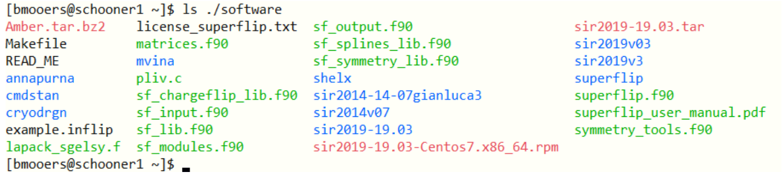
Pre-installed Software

Using simple bash commons to inspect the installed software that are available to all.

```bash
ls /opt/oscer/software
cd /opt/oscer/software
cntdir # returns 669 folders
ls *[d,D]dock* # returns six
module spider Dock # returns five
pwd # present working directory
cd # returns to your home directory
mkdir dumbDemos # No whitespaces in folder names!
rmdir dumbDemos
rm -rf junkFiles # works on folders and files
ls -a # list the hidden files too
ls -d # list the folders only
ls --help
```

Self-installed Software

You can ask the OSCER staff to install software for you, but due to a staff shortage, you may have to wait 2-4 weeks for the software to be installed. You can always install it yourself. It is best to compile Fortan and C programs from source so that they are optimized to run on schooner. I store such software in a directory called ‘software’.



```
[bmooers@schooner1 ~]$ ls ./software
Amber.tar.bz2     license_superflip.txt  sf_output.f90                      sir2019-19.03.tar
Makefile          matrices.f90           sf_splines_lib.f90                 sir2019v03
READ_ME           mvina                  sf_symmetry_lib.f90                sir2019v3
annapurna         pliv.c                 shelx                              superflip
cmdstan           sf_chargeflip_lib.f90  sir2014-14-07gianluca3             superflip.f90
cryodrgn          sf_input.f90           sir2014v07                         superflip_user_manual.pdf
example.inflip    sf_lib.f90             sir2019-19.03                      symmetry_tools.f90
lapack_sgelsy.f   sf_modules.f90         sir2019-19.03-Centos7.x86_64.rpm
[bmooers@schooner1 ~]$ 
```

Use conda to install Emacs Ver. 28

Must have sourced Miniconda (see bottom of .bashrc(2/2) slide above).

```
# The order of commands matters.
# Create a new enviroment called emacs28.
conda create -n emacs28
conda activate emacs28
conda install -c conda-forge emacs
# Now start Emacs.
emacs
# Exit Emacs with Cntrl-x Cntrl-c.
#
# Next, close the emacs28 environent.
conda deactivate
# Enter to activate the emacs28 env
conda activate emacs28
```

Make an alias to emacs28

This alias saves time.

```
alias e28='conda activate emacs28 && emacs'
e28
conda deactivate # exit the emacs28 env
conda env --help
conda env list
ls /home/bmooers/.conda/envs/
```

Alias to the Vim/8.0 module

To ease the loading of the Vim/8.0 module without always loading it on login, use an alias.

```
alias v8='module load Vim/\
8.0-foss-2016a-Python-2.7.11 && vim'
v8
```

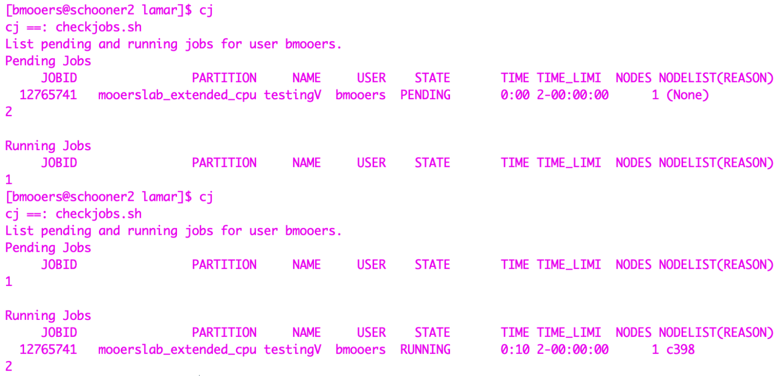
The backslash means line continuation.

Aliases can be defined for other modules that you frequenlty load.

Alias cj

The “cj” alias runs the checkjob.sh script that lists the pending and running jobs in a human friendly format.

```
alias cj='/home/bmooers/bashScripts/checkjobs.sh'
```



```
[bmooers@schooner2 lamar]$ cj
cj ==: checkjobs.sh
List pending and running jobs for user bmooers.
Pending Jobs
     JOBID                PARTITION     NAME     USER    STATE       TIME TIME_LIMI  NODES NODELIST(REASON)
  12765741   mooerslab_extended_cpu testingV  bmooers  PENDING       0:00 2-00:00:00      1 (None)
2

Running Jobs
     JOBID                PARTITION     NAME     USER    STATE       TIME TIME_LIMI  NODES NODELIST(REASON)
1
[bmooers@schooner2 lamar]$ cj
cj ==: checkjobs.sh
List pending and running jobs for user bmooers.
Pending Jobs
     JOBID                PARTITION     NAME     USER    STATE       TIME TIME_LIMI  NODES NODELIST(REASON)
1

Running Jobs
     JOBID                PARTITION     NAME     USER    STATE       TIME TIME_LIMI  NODES NODELIST(REASON)
  12765741   mooerslab_extended_cpu testingV  bmooers  RUNNING       0:10 2-00:00:00      1 c398
2
```

Content of bash script checkjobs.sh

```bash
#!/usr/bin/bash
printf  "Pending Jobs\n" && \
squeue -u bmooers -t PENDING -o "%.10i %.24P \
%.8j %.8u %.8T %.10M %.9l %.6D %R" | sed -n 'p;$=' &&
printf "\nRunning Jobs\n" && \
squeue -u bmooers -t RUNNING -o "%.10i %.24P \
%.8j %.8u %.8T %.10M %.9l %.6D %R" | sed -n 'p;$='
```

Alien world for biologists (1 of 2)

- ssh to login with a password.
- Land of plain text files (No WS Word!)
- No Desktop or folder icons.
- No Web browser or e-mail.
- No GUIs (some exceptions)
- No clickable executables.
- Need to understand the folder hierarchy.
- Need to use **pwd and cd** to navigate between folders.
- Need to use shell scripts.
- Available software is hidden.

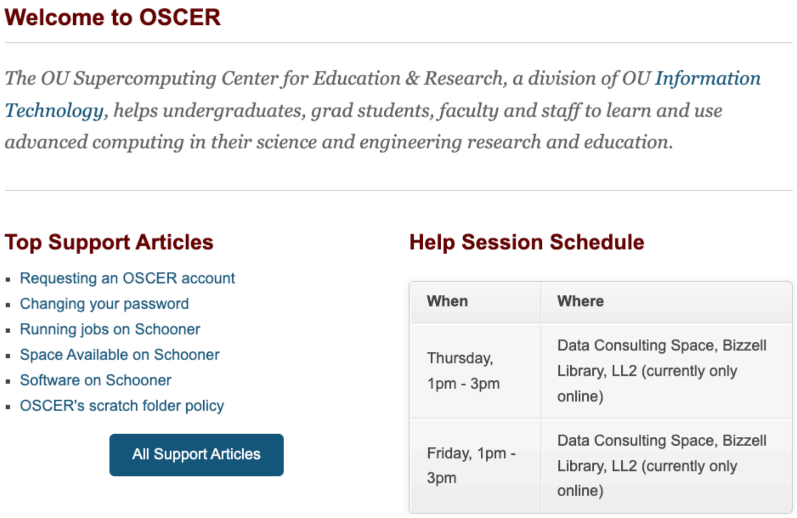
Alien world for biologists (2 of 2)

- Limited interactive computing.
- Linux
- ssh to login with a password.
- Need to understand the folder hierarchy.
- Need to use ls, pwd and cd to move around.
- Have to use the queue system.
- Need to use shell scripts.
- Not obvious where the installed software is hidden.

User-friendly features of schooner

- You can login remotely via ethernet.
- Your mouse works.
- You have 20 GB of space to install software.
- You can install software yourself.
- The scratch disk space is almost unlimited.
- The easy-to-use nano editor is always ready.
- You can run other text editors (Vim, Emacs).
- You can import configuration files for text editors.
- You can use your favorite aliases and bash functions.

Getting Help



Welcome to OSCER

The OU Supercomputing Center for Education & Research, a division of OU Information Technology, helps undergraduates, grad students, faculty and staff to learn and use advanced computing in their science and engineering research and education.

Top Support Articles

- Requesting an OSCER account
- Changing your password
- Running jobs on Schooner
- Space Available on Schooner
- Software on Schooner
- OSCER's scratch folder policy

Help Session Schedule

When	Where
Thursday, 1pm - 3pm	Data Consulting Space, Bizzell Library, LL2 (currently only online)
Friday, 1pm - 3pm	Data Consulting Space, Bizzell Library, LL2 (currently only online)

Useful scripts

- Bash functions for Schooner at OSCER `https://github.com/MooersLab/bashFunctions4oscer`
- Emacs configuration file for schooner supercomputer `https://github.com/MooersLab/emacs4oscer`
- Vim configuration file for schooner supercomputer `https://github.com/MooersLab/vimrc4oscer`

Acknowledgements

Funding:

- NIH: R01 CA242845, R01 AI088011
- NIH: P30 GM145423 (PI Ann West) Oklahoma COBRE in Structural Biology,
- NIH: P30 CA225520 (PI Robert Mannel) Stephenson Cancer Center
- OCAST: HR20-002
- PHF: Team Science Grant with Ian Dunn