Unit 1: Basics of UNIX

September 3, 2015

By UNIX, I mean any UNIX-like operating system, including Linux and Mac OS X. On the Mac you can access a UNIX terminal window with the Terminal application (under /Applications/Utilities from the Finder). Most modern scientific computing is done on UNIX-based machines, often by remotely logging in to a UNIX-based server.

If you're using your own Mac, you should install the Xcode developer tools. As an alternative to using the Mac Terminal, you can use the BCE Linux virtual machine. If you're using a Windows machine, you can use BCE or you could also explore *cygwin*.

Any tutorials mentioned below are available at
http://statistics.berkeley.edu/computing/training/tutorials.

1 BCE

We'll make use of a virtual machine running Linux, called the Berkeley Common Environment (BCE). You'll need to install virtual machine client software called VirtualBox on your computer and then import and start the BCE virtual machine. Please follow the instructions at bce.berkeley.edu under the `Install` tab. Some additional information in the form of screencasts can be found at http://bce.berkeley.edu/screencasts.html (note that the screencast on the BCE Linux command line is the same material in the UNIX basics tutorial mentioned next).

2 UNIX basics

Please see the material in our tutorial on the basics of UNIX to get up to speed on working in a UNIX-style command line environment. The tutorial is presented in the context of BCE, so you should be able to follow and replicate what is done exactly. There are some questions at the end of the tutorial you can use to self-test your understanding.

To connect to a remote machine, you need to use SSH. SSH is available as `ssh` from the BCE command line as well as the Mac Terminal. There are also SSH clients for Windows. For more information on SSH client programs and on using SSH, please see this webpage. Here's an example of connecting to one of the SCF machines from BCE (this assumes you have an SCF account, which you may, and that your user name is paciorek, which it is not).

```
> ssh paciorek@radagast.berkeley.edu
```

To copy files between machines, we can use *scp*, which has similar options to *cp*. The first command here copies a local file to a remote machine. The second copies from a remote machine to the machine you are on. You can use relative paths to refer to locations on the local machine. On the remote machine all paths need to be absolute or to be relative to your home directory on that machine. Again, this assumes you have an SCF account.

```
> scp file.txt paciorek@radagast.berkeley.edu:~/research/.
> scp paciorek@radagast.berkeley.edu:/data/file.txt
~/research/renamed.txt
```

There are also client programs for file transfer; see this webpage.

3 Version control

We'll be using Git extensively to do version control. Git stores the files for a project in a *repository*. Here are some basic Git commands you can use to access the class materials from the command line. There are also graphical interfaces to Git that you can explore. That said, I recommend Github Desktop, available for Mac and Windows.

1. To clone (i.e., copy) a repository (in this case from Github) [*berkeley-stat243* is the project and *stat243-fall-2015* is the repository]:
   ```
   >git clone https://github.com/berkeley-stat243/stat243-fall-2015
   ```

2. To update a repository to reflect changes made in a remote copy of the repository:
   ```
   >cd /to/any/directory/within/the/local/repository
   > git pull
   ```

We'll see a lot more about Git in section, where you'll learn to set up a repository, make changes to repositories and work with local and remote versons of the repository. The section material will follow our tutorial on the basics of Git, Github, and version control.

4 Editors

For statistical computing, we need an editor, not a word processor, because we're going to be operating on files of code and data files, for which word processing formatting gets in the way.

4.1 Some useful editors

- traditional UNIX: *emacs*, *vim*
- Windows: *WinEdt*
- Mac: *Aquamacs Emacs, TextMate, TextEdit*
- some newer editors: *Atom*, *Sublime Text*
- Be careful in Windows - file suffixes are often hidden
- RStudio provides a built-in editor for R code files

4.2 Basic emacs

- *emacs* has special modes for different types of files: R code files, C code files, Latex files – it's worth your time to figure out how to set this up on your machine for the kinds of files you often work on
  - For working with R, ESS (emacs speaks statistics) mode is helpful. This is built into Aquamacs emacs. Alternatively, the Windows and Mac versions of R, as well as RStudio (available for all platforms) provide a GUI with a built-in editor.
- To open emacs in the terminal window rather than as a new window, which is handy when it's too slow (or impossible) to pass (i.e., tunnel) the graphical emacs window through ssh:

```
> emacs -nw file.txt
```

Table 1: Helpful *emacs* control sequences

Sequence	Result
C-x,C-c	Close the file
C-x,C-s	Save the file
C-x,C-w	Save with a new name
C-s	Search
ESC	Get out of command buffer at bottom of screen
C-a	Go to beginning of line
C-e	Go to end of line
C-k	Delete the rest of the line from cursor forward
C-space, then move to end of block	Highlight a block of text
C-w	Remove the highlighted block, putting it in the kill buffer
C-y *(*after using **C-k** or **C-w***)*	Paste from kill buffer ('y' is for 'yank')