Bash: A brief introduction

Medical Biophysics Tech Talks

Slides adapted from the Software Carpentry Workshop Lecture Series

September 2017

Background

At a high level, computers do four things

- run programs
- store data
- communicate with each other, and
- interact with us

The Command-Line Interface

Going back even further, the only way to interact with early computers was to rewire them. But in between, from the 1950s to the 1980s, most people used line printers. These devices only allowed input and output of the letters, numbers, and punctuation found on a standard keyboard, so programming languages and software interfaces had to be designed around that constraint.

This kind of interface is called a command-line interface, or CLI, to distinguish it from a graphical user interface, or GUI, which most people now use. The heart of a CLI is a read-evaluate-print loop, or REPL: when the user types a command and then presses the Enter (or Return) key, the computer reads it, executes it, and prints its output. The user then types another command, and so on until the user logs off.

Nelle's Pipeline: Starting Point

Nelle Nemo, a marine biologist, has just returned from a six-month survey of the North Pacific Gyre, where she has been sampling gelatinous marine life in the Great Pacific Garbage Patch. She has 1520 samples in all and now needs to:

- Run each sample through an assay machine that will measure the relative abundance of 300 different proteins. The machine's output for a single sample is a file with one line for each protein.

Nelle's Pipeline: Starting Point

Nelle Nemo, a marine biologist, has just returned from a six-month survey of the North Pacific Gyre, where she has been sampling gelatinous marine life in the Great Pacific Garbage Patch. She has 1520 samples in all and now needs to:

- Run each sample through an assay machine that will measure the relative abundance of 300 different proteins. The machine's output for a single sample is a file with one line for each protein.
- Calculate statistics for each of the proteins separately using a program her supervisor wrote called `goostats`.

Nelle's Pipeline: Starting Point

Nelle Nemo, a marine biologist, has just returned from a six-month survey of the North Pacific Gyre, where she has been sampling gelatinous marine life in the Great Pacific Garbage Patch. She has 1520 samples in all and now needs to:

- Run each sample through an assay machine that will measure the relative abundance of 300 different proteins. The machine's output for a single sample is a file with one line for each protein.
- Calculate statistics for each of the proteins separately using a program her supervisor wrote called `goostats`.
- Compare the statistics for each protein with corresponding statistics for each other protein using a program one of the other graduate students wrote called `goodiff`.

Nelle's Pipeline: Starting Point

Nelle Nemo, a marine biologist, has just returned from a six-month survey of the North Pacific Gyre, where she has been sampling gelatinous marine life in the Great Pacific Garbage Patch. She has 1520 samples in all and now needs to:

- Run each sample through an assay machine that will measure the relative abundance of 300 different proteins. The machine's output for a single sample is a file with one line for each protein.
- Calculate statistics for each of the proteins separately using a program her supervisor wrote called `goostats`.
- Compare the statistics for each protein with corresponding statistics for each other protein using a program one of the other graduate students wrote called `goodiff`.
- Write up results. Her supervisor would really like her to do this by the end of the month so that her paper can appear in an upcoming special issue of Aquatic Goo Letters.

Key Points

- A shell is a program whose primary purpose is to read commands and run other programs.
- The shell's main advantages are its high action-to-keystroke ratio, its support for automating repetitive tasks, and its capacity to access networked machines.
- The shell's main disadvantages are its primarily textual nature and how cryptic its commands and operation can be.

Navigating Files and Directories

The part of the operating system responsible for managing files and directories is called the **file system**. It organizes our data into files, which hold information, and directories (also called "folders"), which hold files or other directories.

Several commands are frequently used to create, inspect, rename, and delete files and directories. To start exploring them, let's open a shell window:

Absolute vs Relative Paths

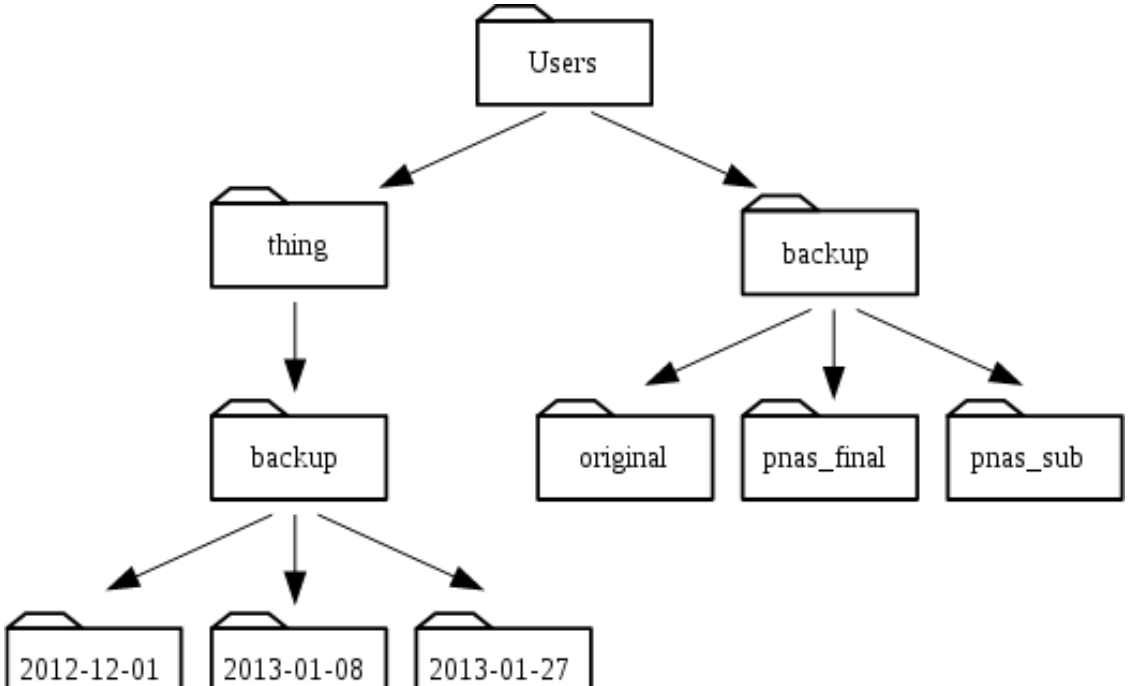
Starting from `/Users/amanda/data/`, which of the following commands could Amanda use to navigate to her home directory, which is `/Users/amanda`?

1. `cd .`
2. `cd /`
3. `cd /home/amanda`
4. `cd ../..`
5. `cd ~`
6. `cd home`
7. `cd ~/data/..`
8. `cd`
9. `cd ..`

Relative Path Resolution

Using the filesystem diagram below, if pwd displays `/Users/thing`, what will `ls -F ../backup` display?

1. `../backup: No such file or directory`
2. `2012-12-01 2013-01-08 2013-01-27`
3. `2012-12-01/ 2013-01-08/ 2013-01-27/`
4. `original/ pnas_final/ pnas_sub/`



`ls` Reading Comprehension

Assuming a directory structure as in the previous Figure (File System for Challenge Questions), if `pwd` displays `/Users/backup`, and `-r` tells `ls` to display things in reverse order, what command will display:

`pnas_sub/ pnas_final/ original/`

1. `ls pwd`
2. `ls -r -F`
3. `ls -r -F /Users/backup`
4. Either 2 or 3 above, but not 1.

Exploring More `ls` Arguments

What does the command ls do when used with the `-l` and `-h` arguments?
Some of its output is about properties that we do not cover in this lesson (such as file permissions and ownership), but the rest should be useful nevertheless.

Listing Recursively and By Time

The command `ls -R` lists the contents of directories recursively, i.e., lists their sub-directories, sub-sub-directories, and so on in alphabetical order at each level. The command `ls -t` lists things by time of last change, with most recently changed files or directories first. In what order does `ls -R -t` display things? Hint: `ls -l` uses a long listing format to view timestamps.

Key Points

1. The file system is responsible for managing information on the disk.
2. Information is stored in files, which are stored in directories (folders).
3. Directories can also store other directories, which forms a directory tree.
4. `cd path` changes the current working directory.
5. `ls path` prints a listing of a specific file or directory; `ls` on its own lists the current working directory.
6. `pwd` prints the user's current working directory.
7. `whoami` shows the user's current identity.
8. `/` on its own is the root directory of the whole file system.

Key Points

9. A relative path specifies a location starting from the current location.
10. An absolute path specifies a location from the root of the file system.
11. `..` means 'the directory above the current one'; `.` on its own means 'the current directory'.
12. Most files' names are something.extension. The extension isn't required, and doesn't guarantee anything, but is normally used to indicate the type of data in the file.
13. Most commands take options (flags) which begin with a `-`.

Working With Files and Directories

Objectives

1. Create a directory hierarchy that matches a given diagram.
2. Create files in that hierarchy using an editor or by copying and renaming existing files.
3. Display the contents of a directory using the command line.
4. Delete specified files and/or directories.

Renaming Files

Suppose that you created a `.txt` file in your current directory to contain a list of the statistical tests you will need to do to analyze your data, and named it: `statstics.txt`
After creating and saving this file you realize you misspelled the filename! You want to correct the mistake, which of the following commands could you use to do so?

1. `cp statstics.txt statistics.txt`
2. `mv statstics.txt statistics.txt`
3. `mv statstics.txt .`
4. `cp statstics.txt .`

Moving and Copying

Suppose I ran the following commands

```
$ pwd
/Users/jamie/data
$ ls
proteins.dat
$ mkdir recombine
$ mv proteins.dat recombine
$ cp recombine/proteins.dat ../proteins-saved.dat
```

What would the output be of running `ls`?

Organizing Directories and Files

Jamie is working on a project and she sees that her files aren't very well organized:

```
$ ls -F
analyzed/ fructose.dat raw/ sucrose.dat
```

The fructose.dat and sucrose.dat files contain output from her data analysis. What command(s) covered in this lesson does she need to run so that the commands below will produce the output shown?

```
$ ls -F
analyzed/ raw/
$ ls analyzed
fructose.dat sucrose.dat
```

Copy With Multiple Filenames

For this exercise, you can test the commands in the `data-shell/data` directory.

In the example below, what does cp do when given several filenames and a directory name?

texttt$ mkdir backup

$ cp amino-acids.txt animals.txt backup/

In the example below, what does cp do when given three or more file names?

texttt$ ls -F amino-acids.txt animals.txt backup/ elements/ morse.txt pdb/ planets.txt salmon.txt sunspot.txt

$ cp amino-acids.txt animals.txt morse.txt