CSUS Astronomy Computer Basics: Version 1 (7 June, 2000)

Randy L. Phelps
Department of Physics and Astronomy
California State University, Sacramento

Introduction

Most of you will be familiar with computers which use the Windows operating system. In the world of science, very few computers use Windows, but rather Unix is the operating system of choice. Most high-end scientific computers in astronomy are built by Sun Microsystems, and they are quite expensive, but they are also very reliable. A Unix-based operating system, called Linux, has recently become quite popular since it can be used on everyday PC computers that cost much less than Unix-based workstations. Additionally, peripherals for PCs tend to be fairly inexpensive, making a Linux-based PC a good compromise for scientific research, particularly at non-research institutions like CSU, Sacramento, where resources for scientific research are rather limited.

For students, the use of a Linux-based machine poses some complications, as those systems are not supported by the University, and hence their administration is the responsibility of the faculty member who uses them. This generally results in a less-than-ideal setup, in terms of installed software, networking, and overall robustness of the system. Additionally, the commands within Linux are generally different enough that students should expect to spend a fair bit of time becoming familiar with "another way" to use computers. This guide is intended to help you through the basics of Linux, and associated programs/software you will be using while undertaking your research project.

I. Logging On

Unlike for Windows, Linux requires the user to have an *account* on the machine, which means you will need a *username* and a *password*. The account will have been setup by the *system administrator*, who will tell you what your username and initial password will be (the password can be changed by you once you logon).

When you first look at the monitor, you will notice that the computer has a name! In my case, I name the machines for beaches. Currently, the computers you will be using are called:

- santa-monica
- santa-barbara

Note that Linux cares about upper-case and lower-case letters.

Ask the system administrator which computers have your account(s).
On "your" computer:

1. Type your *username* at the *login prompt*
2. Type your *password* at the *password prompt*

Recall that Linux cares about upper-case and lower-case letters.

At this point, it is highly desirable to change your password to one that only you will know (be aware that the system manager can override your password, so in theory, there is someone else who can access your files).

To change your password, type: *passwd*

Note the shortened spelling! At this point you will be prompted for your old password and then your new password (twice for verification). Please use a combination of numbers, letters and characters to make your password unique.

You now should be logged in, but in a text only screen. This is analogous to the DOS mode of a PC operating system that you may be familiar with. On a normal PC, the Windows operating system works "on top of" DOS, allowing you to work in a point-and-click mode, that uses windows (hence the name!) to make computing easier. Linux, and in particular the version put out by the company called "Redhat", has a similar feature, called Xwindows.

To invoke the windowing features within Linux, type *startx*

You should now have a window on your screen.

To create yet more windows, beyond the single one you now see:

1. Click on the left mouse button
2. Click on the "New shell" option on the menu

A new shell, or window should now have appeared.

II. Directories/Files

You are now logged on to your computer, but where are you? Each computer has a different *directory structure*, but when you logon you will always be in your so-called *home directory*. For me, this directory is called /home/phelps, as it is the home directory for the person with username "phelps". It is within this directory that all of your administrative items can be kept, whatever they may be (e.g., notes, etc.).

To see what is in your directory, type *ls*, which stands for "list contents of directory"

To create a file within your directory, you may use any number of editing programs, but for consistency, we will all use the program called *emacs*. Emacs is not necessarily the easiest editor to use, but it is amongst the best, and it has features we will want to use. For more on *emacs*, see the later sections.

To create a simple file within a directory:

1. Type *emacs file.txt &*

(The name of the file will be called "file.txt". Note the "&" symbol allows you to continue typing in the window from which *emacs* was invoked. Try this exercise as well without the "&" symbol and try to type in the window from which *emacs* was invoked.

2. Within the screen that has appeared, type *This is a test*
3. Under the heading "Files", click on "Exit emacs"
4. When asked "Save file /home/username/file.txt?", click "yes"
5. Type *ls* to see if your file is there

Your file "file.txt" should be there, but are you sure the file contains the text you typed?

To see the contents of a file, type: *more file.txt* ("*more*" displays the contents of the file.)

Say you want to change the filename from "file.txt" to "File.txt" for some reason. Recall that Linux does care about case, so this *is* a different file name.

To copy a file, type: *cp file.txt File.txt* ("*cp*" copies one file to another)

Now, perhaps, you would like to remove (i.e., delete) the original file, since it contains the same information as the new file.

To remove a file, type: *rm file.txt* ("*rm*" removes a file-Be Careful With This Command!)

The whole process of copying, and then removing a file could have been shortened by simply renaming, or "moving" a file.

To rename a file, type: *mv file.txt File.txt* ("*mv*" moves, or renames, a file)

Your home directory is not, however, the directory where your data analysis will occur. Nor is it necessarily the directory where you want to put all of your administrative-type files. You may know that, on a typical Windows machine, your have a number of *subdirectories*, or a directories within a directory, on your (typically) "C drive". For good organization, it is always best to use subdirectories, which you must create.

To make a directory, type: *mkdir TEST* ("*mkdir*" makes/creates a directory)

Note: Here I have created a subdirectory called "TEST". I try to name all of my subdirectories with all capital letters - it can be a pain, but when you do an ls, the subdirectories stand out, and are listed in alphabetical order, since they all have capital letters. Please try to conform to this "standard".

Since the file we originally created, called, *File.txt*, had as its contents the statement "This is a test", we might like to put this file into the subdirectory *TEST*, since it is indeed a test file, and we would like to know that it need not be kept for eternity. We will, therefore, put the file into the *TEST* subdirectory using the following:

To move a file into a subdirectory, type: *mv File.txt TEST/*

Note: that to move a file you merely need to specify the file to move (File.txt) and the subdirectory (TEST/) into which the file will be moved. The trailing "/" after Test is important to use. Get into the habit of including trailing "/" symbols when specifying directory names!

Another way to accomplish the same task, that is putting a file into a subdirectory, is to copy the file (using *cp*) into the subdirectory, and then removing (*rm*) the original file. While this may not be the preferred method, it does illustrate important subtleties:

To copy a file into a subdirectory, type: *cp File.txt TEST/File.txt* …..(a)
or: *cp File.txt TEST/.* …..(b)
or: *cp File.txt TEST/File_New.txt* …..(c)

Note the subtleties. To copy a file to a destination, you need to have the file name in the destination directory specified (option *a* above). An alternative is to use the dot symbol (.) to use the original file name as the default name in the destination (option *b* above). Using option *c* above allows you to use an alternative name for the file in the destination directory. Remember that the original file will remain in the directory immediately above *TEST* until you remove it, if in fact that is what you want to do. There are reasons why you may not want to delete the original file (if, for example, you want to edit the new file, and retain a copy of the original file in another directory).

So now you have file in a new (sub)directory. How do you see that it is there, and/or see what is in that file? To access the file, or see the contents of the new directory, you must first be in that new directory. In other words, you must change directories.

To change directories, type *cd /home/username/TEST/*, for example

Note: There are a number of subtleties to this command as well. The way the command is written above, *cd /home/username/TEST/*, specifies the complete directory structure for the subdirectory *TEST/*, which resides in the directory */home/username/*. This is the most straightforward way to get to the specified subdirectory, *TEST/*. If you were already in the directory */home/username/*, you could simply type *cd TEST/*. The use of the shortcut depends on where you are located in the directory structure. As time goes on, additional shortcuts will be told to you, but for now, stick with this basic means to navigate directories.

III. Software

A number of different software packages, some more robust than others, will be used to reduce and analyze the data you will be working with. Along with specific programs for photometry, which will be discussed later, the major software packages you should become familiar with include:

- saoimage - a image viewer, written by the Smithsonian Astrophysical Observatory
- ximtool - an image viewer with more capabilities than saoimage
- xv - an internal UNIX/Linux package useful for converting image formats