# A Short Introduction to Unix

Andrew Newman
School of Earth and Atmospheric Sciences
Georgia Tech

Fall 2006


*Adapted from: www.med.nyu.edu/rcr/rcr/course/PPT/unix_lec1.ppt*

# Computers are powerful tools for geophysical research

- Ever increasing reliance on increased computing power for geophysical studies
- As geophysical data sets grow larger and problems become more complex, computing power has also grown (not independently).
- Because of its efficiency, the most powerful computers run Unix
  - Linux, is a Unix-based operating system that is rapidly replacing traditional UNIX because of lower cost, more rapid evolution, and ease of use

# Unix Advantages

- It is very popular, so it is easy to find information and get help
  - Books available at local bookstore/internet
  - Numerous websites
  - Online forums and e-mail lists
  - Many Comp. Sci. students know Unix
- Unix can run on virtually any computer

  (Sun, SGI, x86, Macintosh, etc)
- Unix is free or nearly free
  - Linux/open source software movement, numerous versions of the OS and most all tools are free, or have a free version
  - Linux has caused traditional Unix companies to reduce costs dramatically (e.g., Sun, SGI)

# Stable and Efficient

- Unix is very stable:
  - Operating System crashes are very rare
  - Software failures generally do not require reboot
- Unix is very efficient
  - gets maximum crunching power from the processor(s)
  - Smoothly manages  large data sets
  - Runs well on old systems that are slow using Windows
- Most geophysical tools are developed to run on Unix and Linux machines.
- Matlab programs are an exception because it runs on either Unix or Windows.

# Unix has some Drawbacks

- Command line interface: think DOS
  - + Command line operations use less resources and are easier to automate and remotely manage
  - - Steeper learning curve than windows or Mac
  - + This is changing, on Linux systems, most user tools have GUI interfaces that *may* make it easier
- Many versions of Unix with subtle differences
  - - This is particularly the case between Unix and Linux
  - - Linux is less standardized but evolving much more rapidly (more user-friendly and more powerful): trade-off between standardization and adaptability

# General Unix Tips

- **UNIX is case sensitive!!**
  - `myfile.txt` and `MyFile.txt` do **not** mean the same thing
  - Default directory listings do no separate files from directories like Windows. All structures are listed alphabetically starting with numbers
- Every program is independent
  - the core operating system (known as the ***kernel***) manages each program as a distinct process with its own little chunk of dedicated memory.
  - If one program runs into trouble, it dies, but does not affect the affect the kernel or the other programs running on the computer.
  - Programs and files are less linked than in windows.

# The Unix Shell

- This is the traditional way to communicate with a Unix computer, through a command program known as a ***shell.*** Similar to Windows ***cmd*** or dos prompt.
- The shell interprets the commands that you type on the keyboard.
- Numerous types of shells to work from (e.g. sh, bash, csh, tcsh)—I will try to only use bash
- You can use shell scripts to write simple/complex programs to automate many tasks

# Unix Commands

- Unix commands are short and cryptic like **`vi`** or **`rm`**.
  - Will take some learning.
- Most commands have modifiers which are generally single letters preceded by a hyphen:

  **`ls -l`** or **`mv -R`**

  - Capital and small letters may be different functions

  - A command often requires an argument, which can be a filename or input variable

    **`cat -n filename.txt`**

# Wildcards

- You can substitute the * as a wildcard symbol for any number of characters in any filename.
- If you type just * after a command, it stands for all files in the current directory:

  `ls *` will list all files

- You can mix the * with other characters to form a search pattern:

  `ls a*.txt` will list all files that start with "a" and end in ".txt"

- The "**?**" wildcard stands for any single character:

  `ls draft?.doc` will list *draft1.doc*, *draft2.doc*, *draftb.doc*, etc., but not *draft.doc*

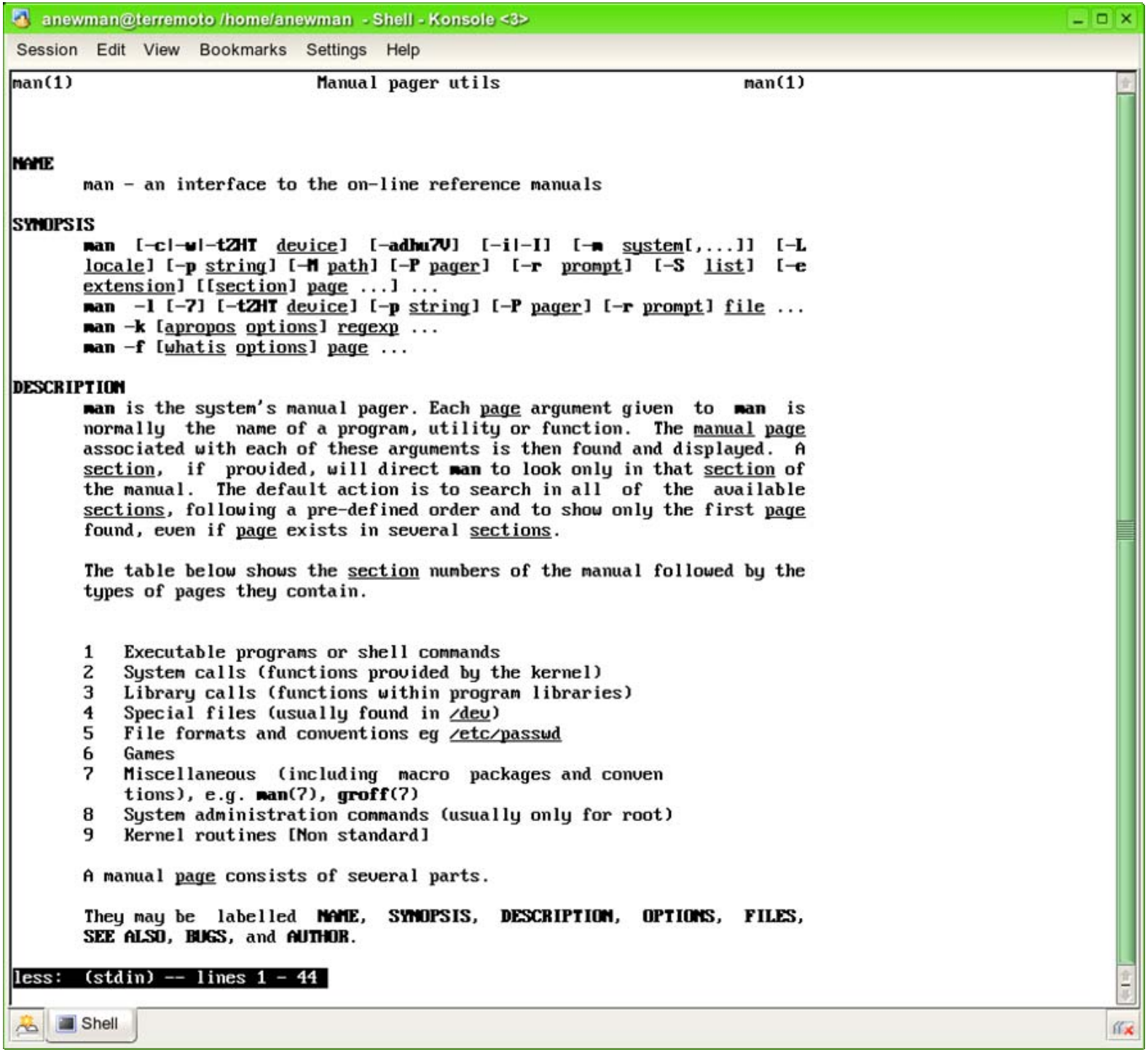
# Typing Mistakes

- Unix is remarkably unforgiving of typing mistakes
  - You can do a lot with just a few keystrokes, but it can be hard or impossible to undo
- If you have not yet hit ‘return’
  - The ‘delete’ key removes the characters that you just typed
  - Generally you can use the arrow keys to move back and forth on the command line to correct a mistake without deleting the rest

# Getting Help in Unix

- Unix is somewhat user-friendly
- There is a simple and standardized help system which consists of a set of "manual" pages for most Unix commands.
- The ***man*** pages tell you which options a particular command can take, and how each option modifies the behavior of the command.
- Type ***man*** and the name of a command to read the manual page for that command.

```
man ls
```


anewman@terremoto /home/anewman - Shell - Konsole <3>
Session Edit View Bookmarks Settings Help
man(1) Manual pager utils man(1)
NAME
man - an interface to the on-line reference manuals
SYNOPSIS
man [-c|-w|-tZHT device] [-adhu7V] [-i|-I] [-m system[,...]] [-L locale] [-p string] [-M path] [-P pager] [-r prompt] [-S list] [-e extension] [[section] page ...] ...
man -l [-7] [-tZHT device] [-p string] [-P pager] [-r prompt] file ...
man -k [apropos options] regexp ...
man -f [whatis options] page ...
DESCRIPTION
man is the system's manual pager. Each page argument given to man is normally the name of a program, utility or function. The manual page associated with each of these arguments is then found and displayed. A section, if provided, will direct man to look only in that section of the manual. The default action is to search in all of the available sections, following a pre-defined order and to show only the first page found, even if page exists in several sections.
The table below shows the section numbers of the manual followed by the types of pages they contain.
1 Executable programs or shell commands
2 System calls (functions provided by the kernel)
3 Library calls (functions within program libraries)
4 Special files (usually found in /dev)
5 File formats and conventions eg /etc/passwd
6 Games
7 Miscellaneous (including macro packages and conven
tions), e.g. man(7), groff(7)
8 System administration commands (usually only for root)
9 Kernel routines [Non standard]
A manual page consists of several parts.
They may be labelled NAME, SYNOPSIS, DESCRIPTION, OPTIONS, FILES, SEE ALSO, BUGS, and AUTHOR.
less: (stdin) -- lines 1 - 44
Shell

# More Help (?)

- The man pages give information about specific commands
- So what if you don't know what command you need?
- **`apropos`** will give you a list of commands that contain a given keyword in their man page header:

```
apropos ftp
```

  - The **`man`** command with the **`-k`** modifier gives a similar result to **`apropos`**
- Look at getting a Unix book
  - O'Reilly books are usually good (e.g. "*Unix in a nutshell*")
- Google search for solutions

# Unix Help on the Web

Here is a list of a few online Unix tutorials:

- Unix for Beginners
  http://www.ee.surrey.ac.uk/Teaching/Unix/
- Getting Started in Unix
  http://www2.ucsc.edu/cats/sc/help/unix/start.shtml
- Unix Guru Universe
  http://www.ugu.com/sui/ugu/show?help.beginners
- Getting Started with the Unix OS
  http://www.leeds.ac.uk/iss/documentation/beg/beg8/beg8.html

# Logging in to the Terremoto (Andy's Linux Box)

From an Ethernet connection within EAS (not the same as Wireless connection)

1. start 'Xwin32' (should be installed on Windows). As a default, this runs an X-server on your machine, allowing remote Unix windows to be displayed
2. Bring up Xconfig within Xwin32
   - Click 'Add…': select StarNetSSH, click next
     - Session name: **terremoto**
     - Hostname: **terremoto.eas.gatech.edu**
     - Login: your login name
     - Username:*I will give this to you (e.g.* ***anewman****)*
     - Command: **/usr/bin/xterm**
     - Leave the rest as default, though you may want to remove the 'show message option once configured correctly'
   - Now start session ***username*@terremoto**
     - You will be prompted for your password (we will set this up together)
   - You should now be connected
3. Test your connection.

# Filename Extensions

- Most UNIX filenames start with a lower case letter and end with a dot followed by one, two, or three letters: **`myfile.txt`**
  - This is a common convention and *is not required*.
  - Is possible to have additional dots in the filename.
- The part of the name following the last dot is called the “extension.”
- The extension is often used to designate the type of file (like windows)

# Some Common Extensions

- By convention:
  - files that end in **`.txt`** are text files
  - files that end in **`.c`** or **`.f`** are source code for "C" or Fortran language
  - files that end in **`.html`** are HTML files for the Web
  - Compressed files have the **`.zip`** or **`.gz`** extension
- Unix does not require these extensions (unlike Windows), but it is a sensible idea and one that you should follow

# Working with Directories

- Directories are a means of organizing your files on a Unix computer.
  - They are equivalent to folders in Windows and Macintosh computers
- Directories contain files, executable programs, and sub-directories
- Understanding how to use directories is crucial to manipulating your files on Unix.

# Your Home Directory

- When you login to a Unix system, you always start in your Home directory.
- Create sub-directories (using **`mkdir`**) to store specific projects or groups of information, just as you would place folders in a filing cabinet.
- Do **not** accumulate thousands of files with cryptic names in your Home directory (unlike me!)

# File & Directory Commands

- This is a minimal list of Unix commands that you must know for file management:

**`ls` (list)**
**`mkdir` (make directory)**
**`cd` (change directory)**
**`rmdir` (remove directory)**
**`cp` (copy)**
**`pwd` (present working directory)**
**`mv` (move, rename)**
**`more`, `less` (view by page)**
**`rm` (remove)**
**`cat` (view entire file on screen)**

- All of these commands can be modified with many options. Learn to use Unix '**`man`**' pages for more information.

# Navigation

- **pwd** (**present working directory**) shows the name and location of the directory where you are currently working:

```
> pwd
/home/anewman
```

  - This is a “pathname,” the slashes indicate sub-directories
  - The initial slash is the “root” of the whole filesytem

- **ls** (**list**) gives you a list of the files in the current directory:

```
> ls
Desktop  NIC00_240.jana  Nicoya  bin  photos
```

  - Use the **ls -l** (**long**) option to get more information about each file

```
> ls -l
total 20
drwx------  2 lvisitor users    4096 Jun 27 16:12 Desktop
drwxrwxr-x  5 lvisitor antelope 4096 Jul 18 13:32 NIC00_240.jana
drwxr-xr-x  3 lvisitor users    4096 Aug 10 07:55 Nicoya
drwxr-xr-x  2 lvisitor users    4096 Aug  8 10:24 bin
drwxr-xr-x  2 lvisitor users    4096 Jul  7 15:14 photos
```

# Sub-directories

- **`cd`** (**change directory**) moves you to another directory

```
>cd bin
> pwd
/home/anewman/bin
```

- **`mkdir`** (**make directory**) creates a new sub-directory inside of the current directory

```
> ls
assembler  phrap      space
> mkdir subdir
> ls
assembler  phrap      space      subdir
```

- **`rmdir`** (**remove directory**) deletes a sub-directory, but the sub-directory must be empty

```
> rmdir subdir
> ls
assembler  phrap      space
```

# Shortcuts

- There are some important shortcuts in Unix for specifying directories
  - `.` (dot) means "the current directory"
  - `..` means "the parent directory" - the directory one level above the current directory, so `cd ..` will move you up one level
  - `~` (tilde) means your Home directory, so `cd ~` will move you back to your Home.
    - Just typing a plain `cd` will also bring you back to your home directory

# Unix File Protections

- File protection (also known as permissions) enables the user to set up a file so that only specific people can read (`r`), write/delete (`w`), and execute (`x`) it.
- Write and delete privilege are the same on a Unix system since write privilege allows someone to overwrite a file with a different one.

# File Owners and Groups

- Unix file permissions are defined according to ownership. The person who creates a file is its owner.
  - You are the owner of files in your Home directory and all its sub-directories
- In addition, there is a concept known as a Group.
  - Members of a group have privileges to see each other's files.
  - This can be done for people working on a joint project or to otherwise share data/programs

# View File Permissions

- Use the **`ls -l`** command to see the permissions for all files in a directory:

```
> ls -l
 total 20
drwx------  2 lvisitor users    4096 Jun 27 16:12 Desktop
drwxrwxr-x  5 lvisitor antelope 4096 Jul 18 13:32 NIC00_240.jana
drwxr-xr-x  3 lvisitor users    4096 Aug 10 07:55 Nicoya
```

  - The username of the owner is shown in the third column. (The owner of the files listed above is lvisitor)
  - The owner belongs to the group "users" or "antelope"

- The access rights for these files is shown in the first column. This column consists of 10 characters known as the attributes of the file: **`r, w, x`**, and **`-`**

  **`d`** indicates directory

  **`---`**

  **`r`** indicates read permission

  **`w`** indicates write (and delete) permission

  **`x`** indicates execute (run) permission

  **`-`** indicates no permission for that operation

```
> ls -l
drwxr-x---  2 browns02 users   8192  Aug 28 18:26    Opioid
-rw-r-----  1 browns02 users   6205  May 30  2000    af124329.gb_in2
-rw-r-----  1 browns02 users   131944 May 31 2000    af151074.fasta
```

- The first character in the attribute string indicates if a file is a directory (**d**) or a regular file (**-**).
- The next 3 characters (**rwx**) give the file permissions for the owner of the file.
- The middle 3 characters give the permissions for other members of the owner's group.
- The last 3 characters give the permissions for everyone else (others)
- The default protections assigned to new files on our system is: **-rw-r-----** (owner=read and write, group =read, others=nothing)

# Changing File Permissions

- Only the owner of a file can change its protections
- Use **`chmod`** (**change mode**) to change/directory file permissions.
  [Beware, this is a confusing command.]

  – Can decide for whom you will change the access permissions:
    - » **the file owner (`u`)**
    - » **the members of your group (`g`)**

# Commands for Files

- Files are used to store information, for example, data or the results of some analysis.

- Many files are simple text and are easy to work with

- **`cat`** dumps the entire contents of a file onto the screen.
  - For a long file this can be annoying

# *More or less*

- commands **`more`** or **`less`** view file contents one screen at a time:

**> more ~/.bash_history**

```
vim readfinsmpout
awk '{printf "%f8.2 %f8.2 \n ", $1,$2}' SIMULPS_clean.xyll | more
awk '{printf "2%f \n ", $1,$2}' SIMULPS_clean.xyll | more
awk '{printf "%f2 \n ", $1,$2}' SIMULPS_clean.xyll | more
awk '{printf "%f2 \n ", $1}' SIMULPS_clean.xyll | more
awk '{printf "%8.2f \n ", $1}' SIMULPS_clean.xyll | more
awk '{printf "2%8.2f \n", $1}' SIMULPS_clean.xyll | more
awk '{printf "%8.2f \n", $1, $2}' SIMULPS_clean.xyll | more
awk '{printf "%8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f
%8.2f %8.2f %8.2f \n"}' SIMULPS_clean.xyll
awk '{printf "%8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f
%8.2f %8.2f %8.2f \n",$0}' SIMULPS_clean.xyll
awk '{printf "%8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f
%8.2f %8.2f %8.2f \n",$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13,$14}' SIMULPS_c
lean.xyll
awk '{printf "%8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f %8.2f
%8.2f %8.2f %8.2f \n",$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13,$14}' SIMULPS_c
lean.xyll > j1
--More--(15%)
```

  - Hit the spacebar to page down through the file
  - On bottom of screen **`more`** shows how much has been displayed
- More sophisticated options for viewing and editing text files are available in a text editor…. a whole other subject/class?

# Copy & Move

- **`cp`** lets you **copy** a file from any directory to any other directory, or create a copy of a file with a new name

```
> cp filename.ext  newfilename.ext
> cp filename.ext   subdir/
> cp /u/jdoe01/filename.ext ./subdir/newfilename.ext
```

- **`mv`** allows you to **move** or rename files
  - Filename and directory syntax for ***mv*** is similar to ***cp***.

```
> mv filename.ext   subdir/newfilename.ext
```

  - **NOTE:** ***mv*** moves a file or directory, thus old file no longer exists

# Delete

- Use the command ***rm*** (**remove)** to delete files
- There is no way to undo this command!!!
  - Most have an alias set up that will ask before it deletes
  - Answer '**Y**', '**y**', anything that starts with '**y**' to delete

```
> ls
Documents  bin  public_html
> touch j1  # creates an empty file named j1
> ls
Documents  bin  j1  public_html
> rm j1
rm: remove test.seq? y
> rm public_html
rm: cannot remove directory `public_html': Is a directory
> rmdir public_html   # will only remove an empty directory
> rm -r public_html   # will recursively remove directory
                      # and everything in it…
                      # be careful with this command!
```