

PATRICIA LEDESMA LIÉBANA

Basic Linux and Unix commands, editing and transferring files

The most recent version of this document can be found in the “Training & Publications” page in the Research Computing web site:

www.kellogg.northwestern.edu/researchcomputing/training.htm

1. Accessing skew4 or WRDS

Current Kellogg faculty and doctoral students can obtain skew4, WRDS, Social Science Computing Cluster (SSCC) and Quest accounts upon request by filling online forms.

For skew, point your web browser to <http://skew4.kellogg.northwestern.edu>. For WRDS, point your browser to <http://wrds.wharton.upenn.edu> and click on “account request”. For an account in the SSCC, click on the link to "Account Application" on the left menu in <http://sscc.northwestern.edu>.

To access either server, you need to connect using some kind of terminal emulator. There are two available through NU site licenses:

1. A text-based interface, Secure Shell SSH
2. A graphical user interface, X-Win32

SSH is the fastest and most reliable. To access WRDS, we recommend SSH over X-Win32: transferring graphics takes time – if you connect with X-Win32, you will notice that the screen refreshes slowly.

This handout is focused on basic commands to "survive" in a Linux or UNIX setting. In addition to skew4 and the SSCC, Northwestern has a high performance computing cluster, "QUEST," with more than 3,000 compute nodes. This cluster went into operation in January 2010.

The SSCC is a cluster of servers, and hence, learning to use the batch job manager software (PBS Pro) is an important additional skill. We have an online web page that walks you through the necessary steps for various types of jobs:

<http://www.kellogg.northwestern.edu/researchcomputing/sec/manuals/pbspro>

For information about QUEST, point your web browser to:

<http://www.it.northwestern.edu/research/adv-research/quest.html>

A note to Mac users: Since the Mac OS is a UNIX-based operating system, SSH is already installed in a Mac: Look for a terminal under Utilities and, to login to skew4 or WRDS, type:

```
ssh login@servername
```

For example: ssh netid@skew4.kellogg.northwestern.edu

Good freeware complements to install in your Macintosh are CyberDuck (<http://cyberduck.ch/>, a graphical user interface for sFTP and FTP) and TextWrangler (<http://www.barebones.com/products/textwrangler/>, a text editor).

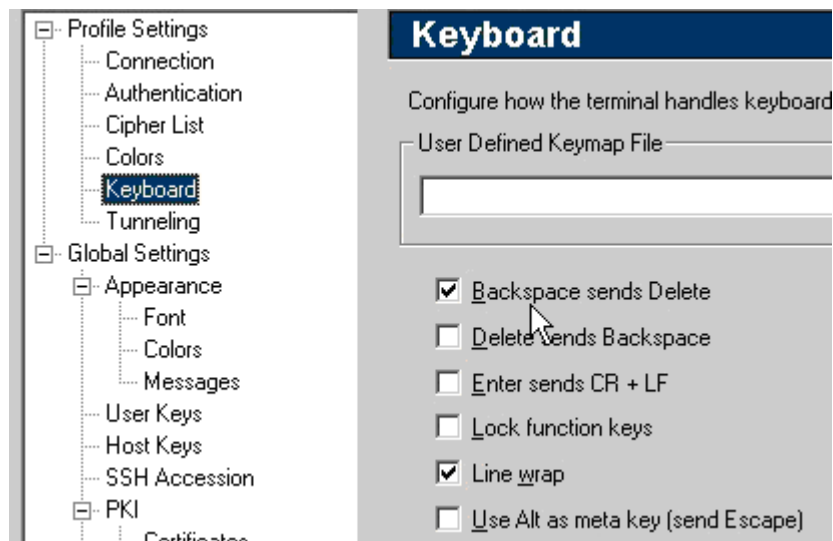
2. Connecting with Secure Shell “SSH”

- This session will be conducted using SSH Secure Shell, which is a connection protocol that encrypts all traffic between the client (the PC where you start SSH) and the host (skew4 or any other UNIX or Linux host). SSH allows you to work in text mode only, i.e., you will not see graphics or Windows-like displays. Some topics will be demonstrated using another terminal emulator that allows you to see graphics, X-Win32. Both programs have been site-licensed by Northwestern. Refer to the following web page:

<http://www.kellogg.northwestern.edu/researchcomputing/terminal-emulators.htm>

In SSH there are two settings you may want to change:

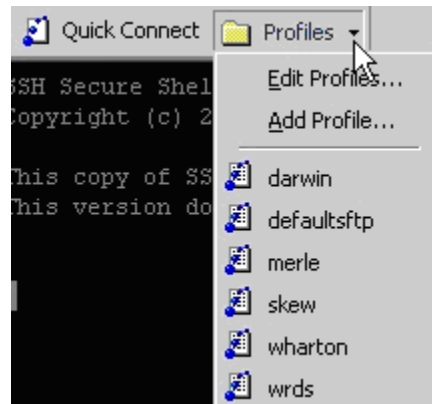
- Backspacing: If you try to backspace in your session and instead you get a series of “^H^H”, select “Settings” from the “Edit” menu. Click on the “+” by Profile Settings and then click on “Keyboard”. The first choice is consistent with a Sun’s keyboard.



- If you use Emacs as an editor, you may set the “Alt” key (normally reserved for MS Windows) as the “meta” key. During this session, we will use “pico” as the editor. “Emacs” and “vi” are two powerful alternatives, but they have a steep learning curve.
- Another convenient customization when you are working with several hosts or several connections at a time is modifying the color. Disabling “Use Global Colors” under the “Colors” setting in the Profile Settings allows you to create different color schemes for each session in the “Global Settings | Appearance” windows.
- Saving your session: Many users ask how to save their setting to avoid typing in the hostname each time. SSH can do this. Click on “Quick connect” and type in the host and your login ID. After you click on the “Connect” button in the dialog window generated by “Quick Connect” you will be asked to type in the password. When the connection is established, SSH will ask you to add this “profile” that includes any color, keyboard, hostname and login name settings (everything you need to save except the password). On the top left corner of the application, a small dialog window will appear where you type in the profile name:



- After typing the profile name, it will appear under the “Profiles” button:



3. Connecting with X-Win32

X-Win32 is software licensed by Northwestern’s Academic Technologies. The licensed is restricted to machines within the Northwestern domain. Thus, it is difficult to get the program to work with VPN.

X-Win allows you to emulate the windows graphical user interface in Linux and UNIX servers.

Once you have logged in, to type in commands, you need a terminal/console window. Right-click on the desktop to see the context menus. The option of a terminal/console window is

typically there. From a terminal window, you can type commands or launch programs such as “xstata” (for the Stata GUI in UNIX) or sftp2, for secure FTP.

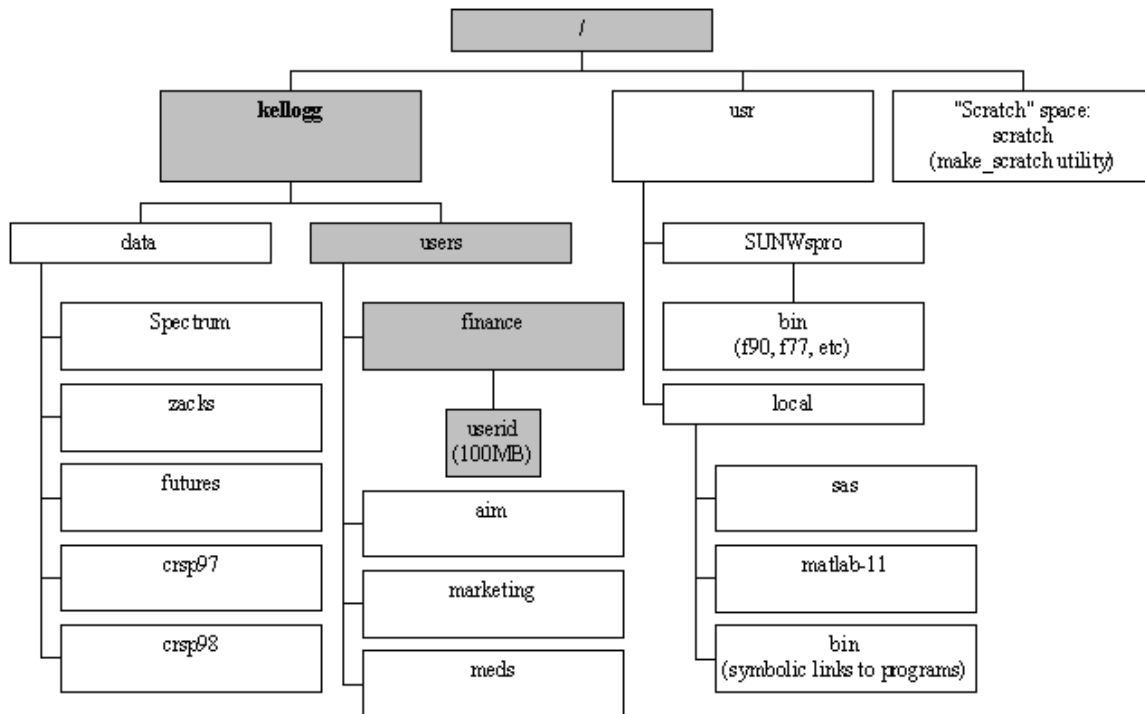
4. Linux/UNIX “personality”

A few Linux/UNIX rules to remember:

- UNIX treats everything (hardware, directories, etc) like a file. Think of a screen or a printer as “write-only” files, keyboards as “read-only” files, etc.
- UNIX commands are case sensitive and they are generally all in lower case.
- File names in UNIX are case sensitive (sample.txt ≠ SAMPLE.TXT ≠ sample.TXT).

The last two items are critical to remember. Commands and statements in most of the applications used in UNIX hosts are not case sensitive, but if you resort to a UNIX command or a file external to the application, you have to remember the exact case.

- In UNIX, files with a name that starts with a period (for example, .cshrc) are “hidden” files. Generally, they are configuration files for an application or shell.
- In UNIX you will not see disks as in a PC (C:\ or D:\). In UNIX there are “file systems” which are neatly joined into a hierarchical tree. Thus, when you switch directories you may be switching between hard drives or even to a CD ROM without realizing it. The top level in the tree is called “root” (the PC equivalent would be C:\ in a hard drive). The following graph shows a simplified version of the skew3 file system.



- Information about skew4: Please check the skew3 web pages for a wealth of information and tips: <http://www.kellogg.northwestern.edu/researchcomputing/skew3.htm>
- Default directory: When you login to skew4, the directory you will see by default (your “home directory”) is:

/kellogg/users/department/netid

where “department” is your departmental affiliation (finance, accounting, marketing, mors, econ, etc) and “netid” is your university network ID. You can return anytime to your home directory by typing simply “cd” at the prompt and pressing enter.

- Shells: When you log into a Linux/UNIX system, you issue commands to a “shell”. A “shell” is the command interpreter; it mediates between the user and the machine itself (DOS, for example, is a command interpreter). There are four shells: the Korn shell (ksh), the Bash or “Bourne-again” shell (bash), the C shell (csh) and the TC shell (tcsh). The default shell in skew4 is the bash, while in WRDS it is the tcsh.

Each shell has configuration files. During the session I will demonstrate what the configuration files are for the tcsh. The configuration files allow you to customize your session. For the tcsh and csh, the configuration file is “.cshrc”; for the ksh and bash shells, the configuration file is “.profile”.

- Most Linux/UNIX commands have options, called “switches”, which modify the command’s behavior.

5. Essential commands

Once you have logged into a server, you can issue commands at the prompt.

| Commands | What they do |
|-----------------------------|-----------------------------------------------------------|
| man | Manual pages in UNIX (e.g. "man pwd") |
| info | Manual pages in Linux (e.g. "info pwd") |
| pwd, whoami | Present working directory, ID of currently logged on user |
| ls | List space, gives a list of files in current directory |
| cd | Change directory |
| more, less, head, tail, cat | Commands to display the contents of an ASCII file |
| mkdir, rmdir | Create (make) a new directory, remove existing directory |
| cp, mv, rm | File operations: Copy, move and remove |
| vi, pico, emacs | Text editors |

- *Wild cards:* you can use wild cards to refer to collections of files. The asterisk ("*") replaces any number of characters (for example, "ls *.txt" would list all the files whose name ends with ".txt" in the current directory), while the question mark ("?") replaces only one character (e.g., "file?.txt" can stand for file1.txt or filea.txt, but not for file12.txt).

The tilde ("~") helps you move quickly to someone's home directory. By itself, it refers to your home directory (for example, "cd ~" would take you back to your home directory). Followed by a netid, it will take you to the specified netid's home directory: "cd ~ple531", for example, will take you to the "ple531" home directory, my directory. By default in Kellogg's system, users cannot browse each other's home directories. I have changed the permissions in mine so that users can browse all of my directories and files except a few protected directories.

- *Redirecting output:* Any command in has a default device for output and a default device for input. For example, when you ask for a file list ("ls"), the output is the screen, while the command takes its input from the current directory (disk). Linux and UNIX allow you to change the input and output devices of commands. For example, instead of showing the list of files on screen, you may want to save that list in a file. You could modify the command as follows: "ls > filelist.txt".
- *Pipes:* You also may use the output of one command as the input for a subsequent one. This can be done using "pipes". The pipe is denoted by the vertical bar. For example:

```
ls | more
```

Sends the result of "ls" (a list of files) to the "more" command. The "more" command shows a file (or in this case, the output of "ls") screen by screen.

- *Repeating commands:* Your session keeps a history of commands given, which you can see with the "history" command. Any of them can be repeated by typing "!#", where "#" is the command number in the command history

Also "!x" will repeat the last command that started with an "x"

"!" will repeat the previous command

- In the tcsh, the up (↑) and down (↓) arrows in the keyboard scroll through the commands you have issued during the session.

Also in the tcsh, if you want to edit a command, CTRL-A will take you to the first character of the command, while CTRL-E will take the cursor to the last character of the command. CTRL-K will delete all characters to the right of the cursor.

- *File and directory permissions:* Linux/UNIX distinguish three types of permissions, read (r), write (w) and execute (x). If a permission is not given, the system denotes it with a dash (“-“). For example, listing the directories in my account shows:

```
drwxrwxr-x+ 5 ple531  nonprofit  1024 Nov  9  2000 nonprofits
drwx----- 2 ple531  kellogg  512 Feb 28  2001 nsmail
drwx----- 2 ple531  kellogg  512 Dec  2  11:34 papers
drwx-----+ 3 ple531  kellogg  3584 May 19  2001 pensions
drwxr-xr-x  2 ple531  kellogg  512 Nov 22  2000 pgm
-rw-----  1 ple531  kellogg 14880 Jul 27  2001 prices.lst
drwxr-xr-x  4 ple531  kellogg  512 Nov 12  11:29 prophet
drwxr-xr-x  2 ple531  kellogg  512 Nov  9  11:10 prophet2
drwxr-xr-x  4 ple531  kellogg  1536 Aug 17  13:57 public
drwxr-xr-x  4 ple531  kellogg  4608 Feb  1  15:44 saspgm
```

Linux/UNIX provide information on three types of users: The owner of the files, members of the same group to which the owner belongs) and “other users”. A user may belong to one or more groups. The following table summarizes the permission levels:

| | | | | | | | | | |
|----------------------------------------------------------------------|---------------------------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| d - l | r - - | w - - | x - - | r - - | w - - | X - - | r - - | w - - | x - - |
| Type of entry: directory (d), file (-) or symbolic link (l) | Permissions for User (owner) | | | Group | | | Other | | |

If you are interested, refer to the “chmod” command. Further explanations may be found in the references mentioned in section 6 of this document and the following web page:

www.kellogg.northwestern.edu/researchcomputing/tutorials/unixhelp/pages/tasks/access_permissions.html (from the RC web page, click on “Servers”, then on “UNIXHelp for Users” and finally on “Controlling access to your files and directories”)

- The “less” command is used to browse the contents of ASCII files. It works in a ways similar to “more” but it offers some convenient features:
 - Press the **spacebar** to move one screen forward
 - Press **enter** to move one line at a time
 - Press > to move to the last screen of the file
 - Press < to move to the first screen of the file

- Press “q” (quit) to return to the prompt at any point

For more options, refer to the “less” manual page (man less).

6. Job monitoring and control

- Within one session, you can monitor the jobs you have submitted in the background with the “jobs” command.
- To get a list of all the processes (loosely put, “commands”) you are running, use the `ps` command:

```
ps -ef | grep netid
```

- To see the processes that are keeping skew3 busy, use the “top” command.
- To send a job to the background, type the command followed by an ampersand. For example:

```
sas myprogram &
```

Effectively, what happens is that the machine works on your SAS file, but it returns your prompt so you can execute other commands or logout of the machine.

- If you submitted a job in the foreground and it is taking a lot longer than you expected, you may pause it and send it to the background. The sequence to do this is:

Hold down the CTRL key and press “Z” (CTRL-Z). This suspends the job.

Send the job to the background by typing “bg”.

- If you submitted a job and need to stop it (or if you are running a job and the machine is no longer responding), for example of you remember you needed to make another change in the program and forgot it:

1. Press CTRL-Z to suspend the job.
2. Then, use the “ps” command to find the process ID number (PID).
3. With the PID in hand, you can terminate the job by “killing” it. There are different signals you can send to a process: “kill pid” will terminate the process specified and all the “children processes” the specified process spawned. Thus, “kill” by itself orders a “clean” termination. If the program does not respond to the simple “kill”, use “kill -9 pid”, which will force the specified process to end. It may not terminate child processes; thus, the unqualified kill is preferable.

7. Editing files

One of the simplest editors installed in most systems is "nano" in Linux and "pico" in UNIX. Start "pico/nano" by typing its name at the command prompt. If you type "pico/nano filename", where "filename" is the name of an existing file, pico will open the file for editing. If "filename" does not exist, pico will create it (if you quit without saving the new file, pico will delete it).

The following graphic shows you how nano looks when you have a file open:

```

GNU nano 1.3.12      File: sasbenchmarks.sas

options fullstimer;
*libname here '.';

/* bench 1: transpose long and wide */

data long;
retain x1-x50 0;
do i = 1 to 7000;
output;
end;
run;
proc transpose data=long out=longt;
var x1-x50;
run;
proc transpose data=longt out=longtt;
var coll-col7000;
run;

^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text    ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell

```

The last few lines show the commands available in nano. The caret (“^”) sign represents the control character. To execute any of the commands shown at the bottom of the nano screen, use the control (Ctrl) key followed by the character shown. For example, to read a file into nano, press “Ctrl-R”.

If “nano” requires a confirmation or further information from you to execute the command, it will prompt you in the bottom of the window. Further “control commands” may appear below that prompt. A few critical commands follow:

| To... | Press... |
|----------------------------|-------------------|
| Save and exit pico | ^x |
| Save without quitting pico | ^o |
| Cut one line | ^k |
| Paste previously cut line | ^u |
| Select a block of lines | ^^ (CTRL-SHIFT-^) |
| Go one page down | ^v |

| | |
|----------------------------------------------|------|
| Go one page up | ^y |
| Go to the top of file | ^w^y |
| Go to bottom of file | ^w^y |
| Insert a file in file currently being edited | ^r |

If you ask for the help pages in nano (^G), you can learn about other commands.

8. Learning more

8.1. UNIX

The RC web pages have some online help available. Refer to:

<http://www.kellogg.northwestern.edu/researchcomputing/tutorials/UNIXhelp/Pages/>

This link and others are available from the RC page on “Hardware”:

<http://www.kellogg.northwestern.edu/researchcomputing/hardware.htm>

There are innumerable books about UNIX. The one I prefer is a volume by Paul W. Abrahams and Bruce A. Larson, “UNIX for the Impatient” (2nd edition). One copy is available for short-term borrowing (up to 7 days) from the Research Computing Library in room 4219.

As you become more sophisticated, there are other references in the RC Library: “Mastering Regular Expressions”. Also, we have reference books on the different shells which can teach you commands and techniques that will make you more efficient in UNIX.

8.2. UNIX Text editors

For those users interested and willing to learn “emacs” or “vi”, there are reference books in the RC Library. Searching the web will provide many useful links. Some of them are in the RC web page on text editors:

<http://www.kellogg.northwestern.edu/researchcomputing/text-editors.htm>

9. Practicing

For this session, we will use the WRDS server. Login in to your WRDS account and try the following commands:

9.1. Basic commands

| Command | What it does |
|-------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>pwd</code> | Will give you location of the current directory. This should be your home directory: <code>/home/nwu/userid</code> |
| <code>whoami</code> | Will return your userid |
| <code>ls</code> | Will return a list of the files in your account. If this is the first time you log in, you should see at least a file called "autoexec.sas" |
| <code>ls -a</code> | Will also list "hidden" files, mostly account configuration files with names that start with a period. For example: <code>.cshrc</code> |
| <code>ls -l</code> | Long listing, including owned, access permissions, date of creation and size. |
| <code>ls -al</code> or <code>ls-la</code> | Combination of the two preceding commands |
| <code>more autoexec.sas</code> | "more" is a command that allows you to browse the contents (no editing) of a text file. If the file is long, notice the <code>"-MORE- (#%) "</code> at the bottom of the screen. Press the space bar to move one screen forward, press enter to move on line forward; press <code>"q"</code> to return to the command prompt. |
| <code>less autoexec.sas</code> | Similar to "more", but also allows backward movement: <code>"b"</code> for one screen back, <code>"d"</code> or spacebar for one screen forward, <code>"q"</code> to quit. |
| <code>man less</code> | View the manual page for the "less" command. Note: In Linux, manual pages are viewed with the "info" command (e.g., "info less"). |

9.2. Creating and removing directories, copying, moving and deleting files

Make sure you are in your home directory by using the "pwd" command. If you are not, type "cd" or "cd ~" to return to your home directory. In your home directory, type the following commands:

| Command | What it does |
|-------------------------------------------|----------------------------------------------------------------------------------------------------------|
| <code>mkdir tmp1</code> | Creates a directory called "tmp1" |
| <code>mkdir tmp2</code> | Creates a directory called "tmp2" |
| <code>cd tmp1</code> | Make "tmp1" the current directory, "change directory" |
| <code>pwd</code> | |
| <code>touch junk1.txt</code> | The "touch" command creates an empty file with the name you specify. |
| <code>touch junk2.txt</code> | |
| <code>touch junk3.log</code> | |
| <code>touch junk4.log</code> | |
| <code>touch basura.txt</code> | |
| <code>ls *.txt</code> | List all files that have ".txt" at the end of their name |
| <code>ls *.log</code> | List all files that have ".log" at the end of their name |
| <code>ls junk*</code> | List all files with a name that starts with "junk" |
| <code>ls junk?.txt</code> | List all files with a name that starts with "junk", followed by any character ("?") and ends with ".txt" |
| <code>rm *.log</code> | Delete all files with name ending in ".log" |
| <code>cp junk1.txt basura2.txt</code> | Copy file "junk1.txt" as "basura2.txt" |
| <code>cp junk1.txt ../morejunk.txt</code> | Copy "junk1.txt" to your home directory as "morejunk.txt" |
| <code>cd ..</code> | Go one directory up. This will place you in your hom directory |
| <code>rmdir tmp2</code> | Delete the "tmp2" directiry |
| <code>rmdir tmp1</code> | Try to remove the "tmp1" directory. This will fail because the directory is not empty! |
| <code>rm -r tmp1</code> | To delete a directory that is not empty, use the "rm" command with the "r" (recursive) flag. |
| <code>rm morejunk.txt</code> | Delete morejunk.txt |

9.3 Moving around the file system

| Command | What it does |
|-----------------------|----------------------------------------------------------------------------------------------------|
| cd / | Go to the "root" (top level in the server's file system) directory |
| ls | |
| cd wrds | Go to the wrds directory |
| ls | |
| cd compustat | Go to the compustat directory |
| ls | |
| cd sasdata | |
| ls | |
| ls *.sas7bdat | There are two types of SAS files: data files (sas7bdat) and index files (sas7bndx) |
| cd ../../crsp/sasdata | Got two directory levels up and then two directory levels down, to "sasdata" in the crsp directory |
| pwd | Check the current directory location |
| ls | Check the contents of the current directory |
| cd or cd ~ | Go back to your home directory |

9.6. Pipes: "concatenating" UNIX commands

| Command | What it does |
|----------------------------|----------------------------------------------|
| cd /wrds/compustat/sasdata | Go back to the compustat/sasdata folder |
| pwd | Verify where you are |
| ls *.sas7bdat more | Display the directory contents page by page. |

9.5 Editing a file

Go back to your home directory and type:

```
pico test.sas
```

The editor window will open. Forget about using your mouse! All commands in pico are listed at the bottom of the screen (see also the list on page 10 of this handout). Type the following lines in the editor:

```
data oranges;
  input variety $ flavor texture looks;
label
  variety = "orange variety"
  flavor = "score for flavor"
  texture = "score for texture"
  looks = "score for looks"
  cards;
navel 9 8 6
temple 7 7 7
valencia 8 9 9
mandarin 5 7 8
;
proc print data=oranges;
```

Press CTRL-O to save the file. Pico will propose the current file name at the bottom of the screen. Press ENTER to accept this.

9.6. Running a simple SAS program

Press CTRL-X to exit pico. If you have modified the file since you last saved, pico will prompt you to save (y/n). If you say yes, it will save the file and exit.

To run the program, type:

```
sas test
```

If this was a program that would take a significant amount of time and you would like to logout while it executes, add an ampersand at the of the command:

```
sas test &
```

SAS assumes that the file is really called "test.sas" (it assumes the "file extension"). It runs the program and returns to command prompt after creating two files: test.lst (if the program produced any "printed" output") and test.log (a log of all the commands and SAS processing messages for your program).

Check test.log to find if there were any errors in your program:

```
less test.log
```

Check the output:

```
less test.lst
```

9.7. Redirecting output

| Command | What it does |
|------------------------------------------------------|---------------------------------------------------------------------------------------------------|
| <code>cd /wrds/compustat/sasdata</code> | |
| <code>ls *.sas7bdat > ~/compustatfiles.txt</code> | Create a text file with a list of all the SAS data files in the current directory |
| <code>cd</code> | Go to your home directory |
| <code>less compustatfiles.txt</code> | Browse the file you just created |
| <code>cd /wrds/compustat/seqdata</code> | Go the the directory that contains the Fortran files for Compustat |
| <code>less ina.names</code> | Browse the contents of a file that has company identifiersin the Compustat annual industrial file |
| <code>head ina.names</code> | Browse the top 10 lines in ina.names |
| <code>tail ina.names</code> | Browse the bottom 10 lines of the same file |
| <code>head.ina.names > ~/inanames.txt</code> | Send the top 10 lines of ina.names to a file in your home directory |
| <code>tail ina.names >> ~/inanames.txt</code> | Append the bottom 10 lines of ina.names to the file you just cerated |
| <code>cd</code> | |
| <code>less inanames.txt</code> | Browse the resulting file |

10. Examples of running Stata and Matlab on skew4 or the SSCC

For many problems, using one of the servers is simply a comfortable way of working. Both the SSCC and skew4 have a rich suite of software that includes popular packages such as Stata and Matlab, among many others. While their graphical user interfaces look just like their Microsoft Windows counterparts, you will realize that "batch" jobs, where you submit a program and run it without opening the graphical user interface, is a faster way of working once you are comfortable programming.

10.1. A Stata example

Create a text file called "statatest.do" using pico, typing in it the following commands:

```
*log using ~/training/statatest.log, text
sysuse auto
describe
list make price mpg foreign in 1/5
save ~/auto
regress price mpg foreign weight
*log close
```

This is a very simple Stata program that uses a data file shipped with all versions of Stata, "auto.dta". Note that in the program, the log commands are remarked, so they do not run. When you run Stata in batch mode, Stata creates a text log file with the same name as your do-file, but with extension ".log". to run this program in batch mode, type the following command at the prompt:

```
stata -b do statatest &
```

The same option works in a Windows PC. By not using the interface, even in the PC, you can shave some time off the program run.

10.2. A Matlab example

Type the following Matlab commands in a file called "test.m":

```
clear all;
clc;
n1=10
n2=10000
x=randn(n1,1);
y=randn(n2,1);
mx=mean(x)
my=mean(y)
stdx=std(x)
stdy=std(y)
save myxymat x y
save myallmat
```

To run it in batch mode in Matlab, type the following commands

```
nohup matlab < test.m > test.txt &
```


Notice that Matlab uses re-direction to read the input file (test.m) and write the output (test.txt). In this case, the output file contains exactly what you would have seen on the screen, had you issued each command in an interactive session.

11. Transferring files between servers and between PCs and servers

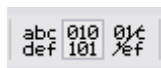
Sooner or later, you will want to transfer a file between computers for further analysis. The Windows SSH application includes a front end for the secure file transfer protocol (sFTP) that allows easy transfer between a PC and a server, while in servers, the ssh and sftp commands allow transfer between servers

SFTP is an agreed upon set of instructions for sending and receiving files through the Internet. Like other Internet protocols (such as telnet, http, or gopher), SFTP has set rules to determine how two machines (regardless of their operating systems) communicate. The protocol can be implemented as a program in any operating system. Thus, you will see a variety of SFTP applications. On campus, for workstations running Microsoft Windows we use Secure FTP as the client, from SSH Communications Security Corp.

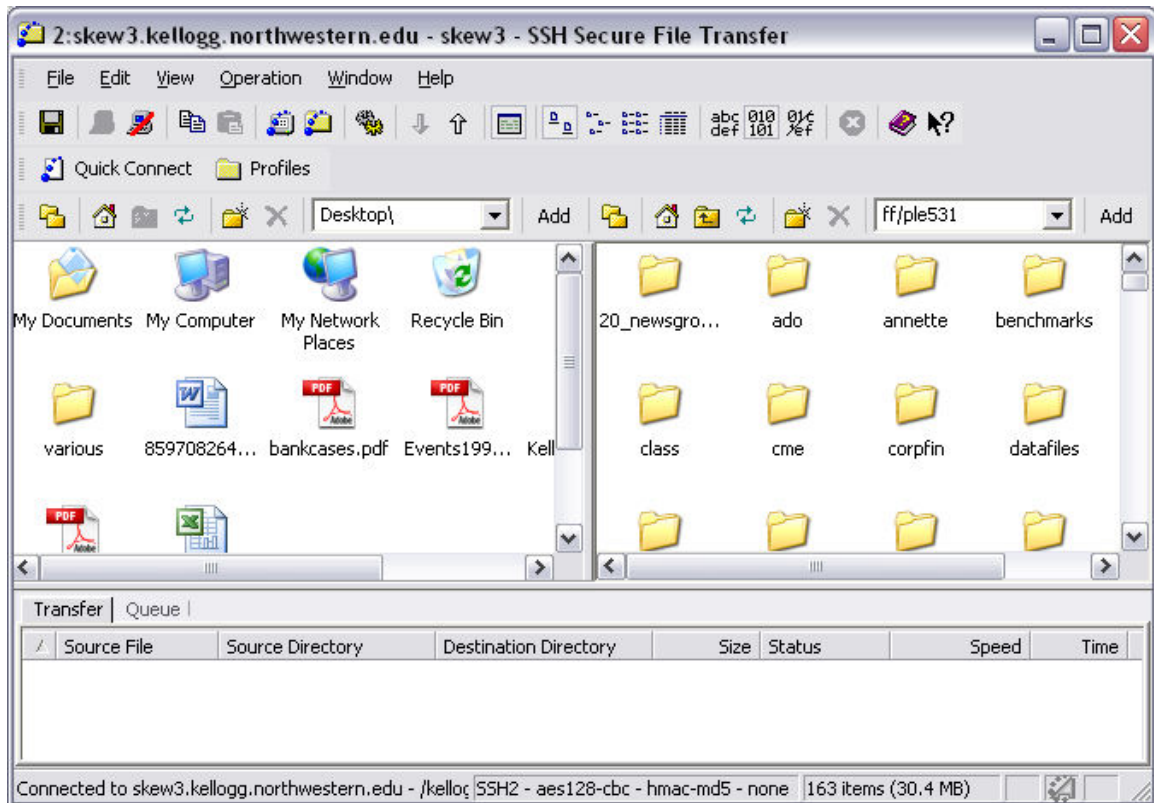
11.1. Between a PC and a server (*skew4*, *SSCC* or *WRDS*)

When you are in SSH, notice an icon of a folder with blue dots across: . Click on it to open the sFTP interface (shown in a screenshot below). The interface consists of two panels. The left one displays the contents of the PC, while the right panel shows the contents of the account where you are logged in (*skew3*, *WRDS* or *SSCC*).

Transferring files in this interface is as easy as dragging and dropping between panels, or selecting "upload" (copying a file from the PC into the server) or "downloading" (copying a file from the server to the PC). You should make sure that you have selected the appropriate file type, text or binary, by clicking on the corresponding icons in the toolbar:



If you upload or download a binary file (such as a SAS or Stata data file) in text format, the file will be corrupted and you will be unable to read it.



11.2. Between servers

- Start a connection by typing your login@host. For example, in skew4 type:

```
sftp wrdslogin@wrds.wharton.upenn.edu
```

Hit enter to connect. Skew4 becomes the client (local) and WRDS is the host.

- *Hosts and clients*: One basic concept to keep in mind is that SFTP links two machines. The machine in which you start SFTP becomes the “client”. The machine you access with SFTP (specifically, you access the machine’s hard drive) becomes the “host”. In order to be able to access a machine with SFTP (i.e., for a machine to be a host), that machine must have been setup in a special way. The machine’s owner or system administrator does this setup. Sometimes the host and client are also called “remote” and “local” machines, respectively.
- *Anonymous FTP*: To access a machine with SFTP you will generally need a login and password. This will grant you access only to certain directories in the machine, the ones your login has the rights to access. Certain machines also allow “anonymous FTP”. The owners of such machines have made available certain directories to the general public. “Anonymous FTP” may require “anonymous” and your e-mail address as the password. The e-mail address is usually required to keep a tally of who has accessed the directories.

- *Type of transfer:* There are ASCII transfers and binary transfers. Any file that is not an ASCII file has to be transferred as binary; otherwise, the file will become unreadable. SFTP client applications may automatically detect the type, but they are not 100% reliable. Be careful.
- *Uploading and downloading:* When you transfer a file from the client machine to the host machine, you are “uploading” the file. When you transfer a file from the host machine to the client machine, you are “downloading” the file.
- You can obtain a list of commands available within SFTP by typing “help” at the prompt. For details on a command, type “help command” (e.g., “help mget”). Note that commands that start with an “l” refer to the local host.

| Type of command | Command | Purpose |
|----------------------------------|--------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| Ending session | quit, exit | Disconnect from host |
| Transferring files | get file | Download file from host to client. Accepts wild cards. For example: “get *.sas7bdat” will retrieve all SAS files whose name starts with a “t”. |
| | put file | Upload file from client to host. Accepts wild cards. |
| Change directory in host | cd directory | Changes to specified directory in host |
| Change directory in client | lcd directory | Changes local directory, i.e., changes to specified directory in the client machine. |
| Make a directory in host | mkdir name | Creates a directory in the host (provided you have write access in the current directory) |
| Remove a directory in host | rm | Deletes a directory in the host, provided you have write access permission to it. |
| Listing files in host | ls | Gives an abbreviated list of files in the host. To see dates of creation, size, etc, type “ls -l”. |
| Listing files in client | lls | Gives an abbreviated list of files in the client. To see dates of creation, size, etc, type “ls -l” |
| Getting help | Help | |
| help command | “?” or “help” gives you a list of available commands. “help” followed by a command name gives the definition of the command | |
| Make a directory in client | lmkdir | Creates a directory in the client (provided you have write access in the current directory) |
| Remove a directory in the client | lrm | Deletes a directory in the client, provided you have write access permission to it. |

