Getting started with Unix commands

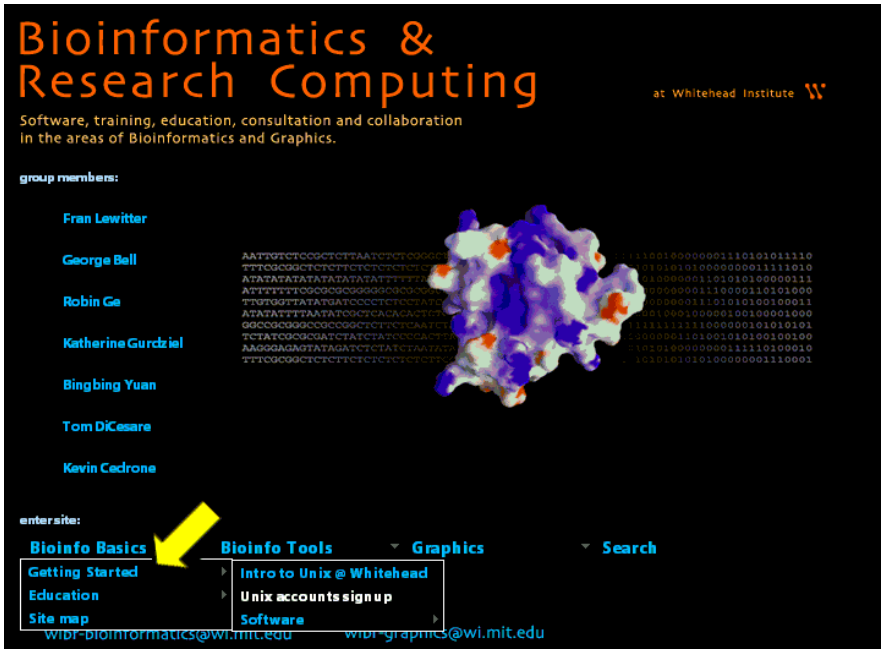
George Bell

Where can these be used?

- Real Unix computers
  - "barra", the Whitehead Linux server
  - Apply for an account on the BaRC page
- Mac computers
  - Come with Unix
- Windows computers
  - Need Cygwin:

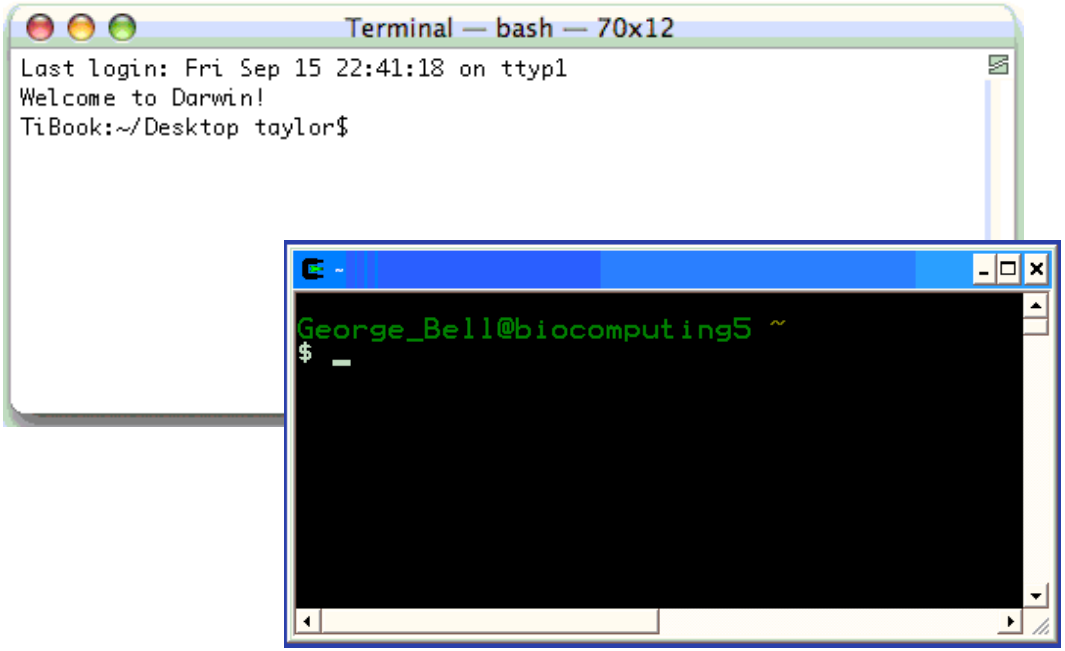
    Free from

    http://www.cygwin.com/



Getting to the terminal

- Macs:
  - Go to Applications => Utilities => Terminal
    or X11
- Windows:
  - Click on Cygwin



- To log in to barra:
  - ssh –l userName barra.wi.mit.edu

Intro to Unix commands

`command [options] [argument(s)]`

- Use up arrow, down arrow to re-use commands
- To get a blank screen: **`clear`**
- To get help for ‘find’ command: **`man find`**
- Avoid filenames with spaces
  - If necessary to use, refer to with quotes:
    `“My dissertation version 1 .txt”`

Where are you?

- Print the working directory

  `pwd`

- **`List all files/directories`**

  `ls` [only show names]

  `ls -l` [show other information too]

Where do you want to go?

- Change directories to where you want to go
- Going up the hierarchy: **`cd ..`**

```
George_Bell@biocomputing5 ~
$ pwd
/cygdrive/c/Documents and Settings/George_Bell

George_Bell@biocomputing5 ~
$ cd Desktop

George_Bell@biocomputing5 ~/Desktop
$ pwd
/cygdrive/c/Documents and Settings/George_Bell/Desktop

George_Bell@biocomputing5 ~/Desktop
$ cd ..

George_Bell@biocomputing5 ~
$ pwd
/cygdrive/c/Documents and Settings/George_Bell
```

Get organized

- Make a directory
  `mkdir my_data`
- Remove a directory (after emptying)
  `rmdir my_data`
- Rename a file or directory
  `mv oldFile newFile`
- Copy a file
  `cp oldFile newFileCopy`
- Delete (remove) a file
  `rm oldFile`

Input/output redirection

- Defaults: stdin = keyboard; stdout = screen
- To modify,
  `command < inputFile > outputFile`
- input examples
  `tr a-z A-Z < my_gene_list`
- output examples
  `ls > file_name` (make new file)
  `ls >> file_name` (append to file)
  `ls foo >! file_name` (overwrite)

Combining commands

- In a pipeline of commands, the output of one command is used as input for the next

- Link commands with the “pipe” symbol: |

  ex1: **`ls *.fa | wc -l`**

  ex2: **`grep “>” *.fa | sort`**

File permissions

- Who can read, write, or execute files?
- User, group, or others?

```
GWB @ barra=>ls -l
total 152
drwxr-x---   4 rodrigue wheel 83968 Aug 15 11:43 AOS1_RE/
-rw-r-----   1 rodrigue wheel  1375 Aug 15 14:33 dir_README.txt
drwxr-x---   4 rodrigue wheel  2048 Aug 15 11:40 hochwagen/
drwxr-x---   7 rodrigue wheel  1024 May  2  2007 lib/
drwxr-x---   4 rodrigue wheel  1024 Jul  3  2007 microarray/
drwxr-x---  14 rodrigue wheel  1024 Aug 15 11:43 minor/
drwxr-x---   6 rodrigue wheel  1024 May 25  2007 rodrigue/
drwxr-x---   4 rodrigue wheel  2048 Aug 15 17:20 status_reports/
drwxr-x---   2 rodrigue wheel  2048 Jun  6  2007 targetscan/

4:53pm /nfs/BaRC/rodrigue
GWB @ barra=>mv microarray foobar
mv: cannot move 'microarray' to 'foobar': Permission denied
```

-rw-r--r—

Only user can edit

-rwxrwxrwx

Anyone can edit

Changing permissions

- 9 choices (rwx or each type of person; default = 644)

  0 = no permission
  1 = execute only
  2 = write only
  3 = x + w
  4 = read only
  5 = r + x
  6 = r + w
  7 = r + w + x

```
chmod 644 myFile
chmod 660 myFile
chmod 755 myDirectory
```

Powerful Unix Commands

Bingbing Yuan

head/tail

- **Display first n lines of file: n=2**

  head –2 FILE

- **Display last n lines of file: n=1**

  tail –1 FILE

cat
concatenate files

- **cat file1 file2 file3 > bigFile**

- **more file:**

  A it

  B his

  D her

- **cat –A file:**

  A^Iit$

  B^Ihis$

  D^Iher$

-A	show all
^I	TAB (\t)
$	end of line ($)
^M	carriage return(\r)

wc:

print the number of **newlines**, **words**, and **bytes** in files

- **more FILE**

  AF045464_s_at 8.73E-05
  AF045564_at 0.093371109
  AF045564_g_at 0.000691539

- **wc FILE**

  3 6 73

- **wc –l FILE**

  3

-l	**line counts**
-w	**word counts**
-m	**character counts**

split

split a file into pieces

- **wc –l FILE**

  50000

- **split –l 10000 FILE | wc –l *** **(default PREFIX is `x‘)**

  50000 FILE
  10000 xaa
  10000 xab
  10000 xac
  10000 xad
  10000 xae

- **split –l 10000 –d FILE “FILE_” | wc –l FILE***

50000 FILE
10000 FILE_00
10000 FILE_01
10000 FILE_02
10000 FILE_03
10000 FILE_04

-l	put NUMBER lines per output file
-d	use numeric suffixes instead of alphabetic

sort

sort lines of text files

- more FILE

B	5	a	4
C	2	d	2
C	2	a	3
D	3	a	2
A	1	a	5

- sort FILE

A	1	a	5
B	5	a	4
C	2	d	2
C	2	a	3
D	3	a	2

- sort –k2,2n FILE

A	1	a	5
C	2	*a*	3
C	2	d	2
D	3	a	2
B	5	a	4

-k	Field
-n	numeric sort
-r	reverse
-t	field-separator. Default: space –t; -t\t –t'|'

sort

sort lines of text files

- more FILE

  >gi|164510870|emb|AM293347.1|
  >gi|157064938|gb|EF633691.1|
  >gi|145701034|gb|DQ336176.3|
  >gi|164510868|emb|AM293346.1|

- sort -t'|' -k3 FILE

  >gi|164510868|emb|AM293346.1|
  >gi|164510870|emb|AM293347.1|
  >gi|145701034|gb|DQ336176.3|
  >gi|157064938|gb|EF633691.1|

uniq

remove duplicate lines from a sorted file

- **more FILE**

chr6.fa 34314346 F
chr6.fa 52151626 R
chr6.fa 81889764 R
chr6.fa 52151626 R

- **uniq FILE**

chr6.fa 34314346 F
chr6.fa 52151626 R
chr6.fa 81889764 R
chr6.fa 52151626 R

-u	unique
-d	repeated

- **sort FILE**

chr6.fa 34314346 F
chr6.fa 52151626 R
chr6.fa 52151626 R
chr6.fa 81889764 R

- **sort FILE |uniq**

chr6.fa 34314346 F
chr6.fa 52151626 R
chr6.fa 81889764 R

- **sort FILE | uniq –d**

chr6.fa 52151626 R

- **sort FILE |uniq –u**

chr6.fa 34314346 F
chr6.fa 81889764 R

grep
print lines matching a pattern

- more FILE

  U0 chr19.fa 4126539 R
  U0 chr6.fa 81889764 R
  U0 Chr6.fa 77172493 R

- grep ‘chr6’ FILE

  U0 chr6.fa 81889764 R

- grep –i ‘chr6’ FILE

  U0 chr6.fa 81889764 R
  U0 Chr6.fa 77172493 R

- grep -v ‘chr19’ FILE

  U0 chr6.fa 81889764 R
  U0 Chr6.fa 77172493 R

- grep ‘chr6|chr19’ FILE

  U0 chr19.fa 4126539 R
  U0 chr6.fa 81889764 R

-v	select non-matching lines
-i	ignore case
|	or

grep
print lines matching a pattern

- **grep “>” seqFile.fa**

>gi|164510870|emb|AM293347.1|Schmidtea mediterranea mRNA for msh2 protein

- **> : is required to be at the beginning of the header line in fasta sequence**

- **grep –A 3 “>” seqFile.fa**

>gi|164510870|emb|AM293347.1| Schmidtea mediterranea mRNA for msh2 protein

ACAATCAATAAAATAAAATCATTGATCTCATA

GCCTCATTGGCTAATTGAATTGACTGCTTGA

AGCCTATCAGAAATTTTTACAGCGGAA

- **-A NUM**
  - **Print NUM of lines After the matching line**
- **-B NUM**
  - **Print NUM of lines Before the matching line**
- **-C NUM**
  - **Print NUM of lines Before and After the matching line**

cut

cut sections from each line of files

- more FILE

```
SOLEXA_2    GAAGTGGATTAGAGTGTGAATTGGCC    U0    1    0    0    chrX.fa 78426100    R    ..
SOLEXA_8    ATACCTGGATCTTCCAGCTTGGGGAC    U0    1    0    0    chr1.fa 77055965    F    ..
```

- cut –f1,2,7-9 FILE

```
SOLEXA_2    GAAGTGGATTAGAGTGTGAATTGGCC    chrX.fa 78426100    R
SOLEXA_8    ATACCTGGATCTTCCAGCTTGGGGAC    chr1.fa 77055965    F
```

- more FILE

```
>WICMT-SOLEXA_8:3:1:908:882
>WICMT-SOLEXA_8:3:1:113:668
```

- cut -d: -f1,4,5 FILE

```
>WICMT-SOLEXA_8:908:882
>WICMT-SOLEXA_8:113:668
```

-f	output only these fields
-d	field delimiter Default: TAB

paste
merge lines of files

- more FILE1

  AF045464_s_at 8.73E-05
  rc_H33614_at 5.757720815
  M10068mRNA_s_at 7.913310223

- more FILE2

  AF045464_s_at Akr7a3
  rc_H33614_at Zfp426
  M10068mRNA_s_at Por

- paste FILE1 FILE2 (require same number of lines)

  AF045464_s_at 8.73E-05 AF045464_s_at Akr7a3
  rc_H33614_at 5.757720815 rc_H33614_at Zfp426
  M10068mRNA_s_at 7.913310223 M10068mRNA_s_at Por

- Refer to 'join' for complicated table join

find

find files in a directory hierarchy

- **Find all the files by filename**
  - **find /nfs/young_ata/00_BaRC/ -name "*eland*“**

    /nfs/young_ata/00_BaRC/Run_Solexa/Test_data/eland_info.txt
- **Find all the files by keyword**
  - **find . -name "*.sh" | xargs grep “blastall"**
  - **More powerful than grep “blastall” file**
- **Find old files:**
  - **find . -mtime +6 -mtime -8**

-name	filename
-mtime n	last modified n*24 hours ago
-mmin n	last modified n minutes ago
xargs	reads standard input to specify arguments to that command

ps
report process status

- **ps –u byuan**

```
PID    TTY      TIME    CMD
24956 pts/11   00:00:00 tcsh
25751 pts/11   00:00:00 ps
25992 pts/11    00:12:40 /home/byuan/parse.pl FILE
```

- **kill 25992**
- **ps –u byuan**

```
PID    TTY      TIME    CMD
24956 pts/11   00:00:00 tcsh
25751 pts/11   00:00:00 ps
```

-u	User name
PID	Process ID
TTY	Terminal name
TIME	cumulated CPU time hh:mm:ss
CMD	Executable name

awk

Alfred Aho, Peter Weinberger, and Brian Kernighan

- more FILE

```
AATGCACTTCCTGTGCCAGTGCCCGC    U0    1    0    0    chr6.fa 34314346    F
ACCAGTGCCACGGTGTCTGCAGCTAA    U0    1    0    0    chr1.fa 33161725    F
AGACGGAAAACTTTGGCCGACCTTGC    U0    1    0    0    chr7.fa 126653439   R
```

- awk '{print FNR"\t"$0 }' FILE >FILE2 (add line number)

```
1    AATGCACTTCCTGTGCCAGTGCCCGC    U0    1    0    0    chr6.fa 34314346    F
2    ACCAGTGCCACGGTGTCTGCAGCTAA    U0    1    0    0    chr1.fa 33161725    F
3    AGACGGAAAACTTTGGCCGACCTTGC    U0    1    0    0    chr7.fa 126653439   R
```

- awk 'BEGIN {OFS="\t"} {print $1,$2,$7,$8,$9}' FILE2 >FILE3

```
1    AATGCACTTCCTGTGCCAGTGCCCGC    chr6.fa 34314346    F
2    ACCAGTGCCACGGTGTCTGCAGCTAA    chr1.fa 33161725    F
3    AGACGGAAAACTTTGGCCGACCTTGC    chr7.fa 126653439   R
```

- awk '{ print ">"$1"\n"$2 }' FILE3 >FILE4

```
>1
AATGCACTTCCTGTGCCAGTGCCCGC
```

FNR	Number of Record in the current File
OFS	Output Field Separator
$0	whole record

awk

- more FILE

```
U0  chr19.fa   4126539    R
U0  chr6.fa    81889764   R
U1  chr6.fa    77172493   R
U0  chr6.fa    4350033    F
```

- awk -F"\t" '$1~/U0/ && $2 ~/chr6.fa/ { print $0 }‘ FILE

```
U0      chr6.fa    81889764        R
U0      chr6.fa    4350033         F
```

- more FILE

```
A       1       a
B       5       ab
C       2       ac
D       3       ad
```

- awk '{ sum=sum+$2} END{print sum}' FILE

```
11
```

awk

Binary Operators

Operator	Type	Meaning
+	Arithmetic	Addition
-	Arithmetic	Subtraction
*	Arithmetic	Multiplication
/	Arithmetic	Division
%	Arithmetic	Modulo

Relational Operators

Operator	Meaning
==	Is equal
!=	Is not equal to
>	Is greater than
>=	Is greater than or equal to
<	Is less than
<=	Is less than or equal to

Regular Expression Operators

Operator	Meaning
~	Matches
!~	Doesn't match

Boolean operators

Operator	Meaning
&&	AND
||	OR

Others

symbol	Meaning
\t	TAB
\n	return
$	Field reference
-F	input field separator default: space

All commands for today

ssh	mv	wc	paste
pwd	cp	split	find
ls	rm	sort	ps
cd	chmod	uniq	awk
mkdir	cat	grep	logout
rmdir	more	cut	quit