

(Re)introduction to Unix

Sarah Medland





So Unix...

- Long and venerable history
 - <http://en.wikipedia.org/wiki/Unix>
- Numerous 'flavours' or shells
 - b – Bourne shell
 - k – Korn shell
 - tcsh – tenex
 - bash – Bourne Again Shell

Using Unix

- Servers
 - Linux
- Mac
 - Terminal
- PC
 - Cygwin





Getting started...

- Changing directory
 - `cd C://`
 - Going back one step `cd ..`
 - Finding out where you are `pwd`

Making a directory

```
mkdir wedtemp  
cd wedtemp
```

!!!!Unix is case sensitive!!!!



Formats

- NO SPACES in your file/directory names!!
- To get from DOS (windows) format to Unix use `dos2unix`
- To get from Unix to Dos `unix2dos`
 - Type: `dos2unix output.mxo`
- Wildcard `dos2unix *.mxo`



Input Output

○ Input

- Most commands don't need input signifiers
- `<` can be used to specify

○ Output

- Without specifying most output will print to the screen
- `>` can be used to direct
 - type: `echo 'this is a dummy file'`
 - `echo 'this is a dummy file' > dummy.txt`
 - | (pipe) | `more` pauses the output after a screen worth of text has appeared hit the space bar to get the next screens worth



The manual

- The `man` command can be used in conjunction with other commands to put up some basic instructions
- type: `man ls`
 - `ls` is the `list` command it pulls up a list of the files in the directory

Also many many helpful webpages w examples



Permissions

the ability to **read**, **write** and **execute** files

- type: `ls -l`

```
Integlio@Lapis /cygdrive/c/wedtemp
$ ls -l
total 32
-rw-r--r-- 1 Integlio mkpasswd 21 Mar  4 13:25 dummy.txt
```

- These are the permissions
- 1st a directory flag (d or -)
- then 3 letters to define the owners permissions
- 3 letters to define the groups permissions
- 3 letters to define the everyone else's permissions



Permissions

the ability to read, write and execute files

- read access
- write access
- execute
 - to 'run' script or a program the file must be made executable



Permissions

the ability to **read**, **write** and **execute** files

- To **change** the **mode**/permissions use **chmod**
 - a number of ways to do this
 - type: `ls -l`
 - `chmod +x dummy.txt`
 - `ls -l`
 - `chmod -x dummy.txt`
 - `ls -l`
 - what happened?



Useful 'one liners'

- `cp` copy
- `mv` move = rename
- `rm` remove
- `ls` list
- `echo`
- `head` looks at the top 10 lines
- `tail` looks at the last 10 lines
- `wc` counts number of lines, words, characters



Grep

- search **g**lobally for lines matching the **r**egular **e**xpression, and **p**rint them
 - For example output.mxo is output from a loop script which ran linkage at 59 loci (FEQmodel_Pihat1-59_DZibd.mx)
 - To extract the -2LL at these loci
 - Type: `grep 'of data' output.mxo > ll.txt`



Grep

- Useful flags

- -v
 - reverse grep select line that does not have the pattern
- -f filename
 - To obtain patterns from a file
- -n
 - Print the line number before the line
- Many more...



Awk

- derived from the surnames of its authors — Alfred **A**ho, Peter **W**einberger, and Brian **K**ernighan
- Many functions
- Very useful for restructuring data

Awk

- Ozbmi2.rec

115	0	0.21	1	2	58	57	1.7	1.7	20.0692	19.7232	20.9943	20.8726
121	0	0.24	1	2	54	53	1.6299	1.6299	20.3244	19.9481	21.0828	20.9519
158	0	0.21	1	2	55	50	1.6499	1.6799	20.202	17.7154	21.0405	20.121
172	0	0.21	1	2	66	76	1.5698	1.6499	26.7759	27.9155	23.0125	23.3043
182	0	0.19	1	2	50	48	1.6099	1.6299	19.2894	18.0662	20.7169	20.2583
199	0	0.26	1	2	60	60	1.5999	1.5698	23.4375	24.3418	22.0804	22.3454
221	0	0.23	1	2	65	65	1.75	1.7698	21.2245	20.7476	21.3861	21.227
239	0	0.29	1	2	40	39	1.5598	1.5298	16.4366	16.6603	19.5966	19.6912
246	0	0.24	1	2	60	57	1.7598	1.7698	19.3698	18.194	20.746	20.3076

- awk '{ print \$1, \$10, \$11, \$4, \$5 ;}' ozbmi2.rec > new.rec

```
115 20.0692 19.7232 1 2
121 20.3244 19.9481 1 2
158 20.202 17.7154 1 2
172 26.7759 27.9155 1 2
182 19.2894 18.0662 1 2
199 23.4375 24.3418 1 2
221 21.2245 20.7476 1 2
239 16.4366 16.6603 1 2
246 19.3698 18.194 1 2
```



Awk

- \$1 = column 1
- Print \$0 = print whole line
- add subtract multiply etc
- change number of decimals
- Many functions



Sort

- Useful flags

- -f ignore case
- -n numeric sort
- -r reverse
- -c check if a file is sorted
- -u prints only unique lines
- -k2 sort starting at column2



Putting it together

- Making a 'shell' script to automate analyses

<contents of imaginary file inefficient.sh>

```
pedstats -p 1.ped -d 1.dat -pdf --prefix:1
```

```
merlin -p 1.ped -d 1.dat -m 1.map --vc --pdf --prefix:1
```

```
pedstats -p 2.ped -d 2.dat -pdf --prefix:2
```

```
merlin -p 2.ped -d 2.dat -m 2.map --vc --pdf --prefix:2
```

```
pedstats -p 3.ped -d 3.dat -pdf --prefix:3
```

```
merlin -p 3.ped -d 3.dat -m 3.map --vc --pdf --prefix:3
```

To run this make inefficient.sh executable then type `./inefficient.sh`



Loops 1

<contents of imaginary file loop_a.sh>

```
for $i in 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16  
17 18 19 20 21 22
```

```
do
```

```
  pedstats -p $i.ped -d $i.dat --pdf --prefix:$i
```

```
  merlin -p $i.ped -d $i.dat -m $i.map --vc --pdf --prefix:$i
```

```
done
```



Loops 2

<contents of imaginary file loop_b.sh>

```
for (( i = 1 ; i <= 22 ; i++ ))
```

```
do
```

```
    pedstats -p $i.ped -d $i.dat --pdf --prefix:$i
```

```
    merlin -p $i.ped -d $i.dat -m $i.map --vc --pdf --  
    prefix:$i
```

```
done
```



Permutation

```
#Permutation for sibpair linkage at 728 loci
for (( b = 1 ; b <= 728 ; b++ ))
do
cp use"$b" mx.dat
#permute & re-run analysis 10000 times at each locus
for (( c = 1 ; c <= 10000 ; c++ ))
do
echo m $b rep $c
#
awk 'BEGIN {srand()} {print $1, $2, $3, $4, $5 , rand() ;}' mx.dat |
    sort -k6 > perm
paste -d " " perm pheno.txt > use
./mx63.bat perm.mx
grep "of data" perm.mxo >> link"$b".txt
done
#_____
done
#_____
```



Permutation

```
#Permutation for sibpair linkage at 728 loci  
for (( b = 1 ; b <= 728 ; b++ ))  
do  
cp loci"$b" mx.dat
```

	Fam	ibd0	ibd1m	ibd1p	ibd2	trait1	sex1	trait2	sex2	cov1	cov2
80020	.0310	.1080	.3520	.5090	3260	1	3487	0	.00	24.00	
80030	.0600	.3080	.5050	.1270	3260	1	1786	1	6.00	20.00	
80033	.0050	.0640	.0640	.8660	3005	1	3544	0	.00	29.00	
80040	.0550	.7100	.0480	.1870	2552	0	2438	1	3.00	22.00	
80090	.0610	.7920	.0330	.1140	2920	1	2126	0	3.00	37.00	
80092	.0050	.0640	.0640	.8660	1928	0	1928	0	6.00	32.00	
80110	.0630	.7540	.1030	.0800	3260	1	2041	1	.00	29.00	
80140	.0250	.1150	.2670	.5920	2495	0	2381	0	.00	23.00	
80200	.4440	.3990	.0830	.0740	3033	1	3317	0	.00	30.00	



Permutation

```
#permute & re-run analysis 10000 times at each locus
for (( c = 1 ; c <= 10000 ; c++ ))
do
echo m $b rep $c
#
awk 'BEGIN {srand()} {print $1, $2, $3, $4, $5 , rand() ;}' mx.dat |
    sort -k6 > perm
```

Fam ibd0 ibd1m ibd1p ibd2 random (based on time)

82610	.0000	.0000	.9700	.0300	0.00145522
86007	.0000	.0140	.0120	.9740	0.00736606
81971	.0580	.0130	.8610	.0670	0.00794225
85834	.8280	.1570	.0120	.0020	0.00827437
85820	.0000	.0030	.1030	.8940	0.00837393
85620	.0680	.9250	.0060	.0010	0.0138645
85552	.0040	.1820	.0710	.7420	0.014908

Permutation

```
paste -d " " perm pheno.txt > use
```

perm						+	pheno.txt					
82610	.0000	.0000	.9700	.0300	0.00145522		3260	1	3487	0	.00	24.00
86007	.0000	.0140	.0120	.9740	0.00736606		3260	1	1786	1	6.00	20.00
81971	.0580	.0130	.8610	.0670	0.00794225		3005	1	3544	0	.00	29.00
85834	.8280	.1570	.0120	.0020	0.00827437		2552	0	2438	1	3.00	22.00
85820	.0000	.0030	.1030	.8940	0.00837393		2920	1	2126	0	3.00	37.00
85620	.0680	.9250	.0060	.0010	0.0138645		1928	0	1928	0	6.00	32.00
85552	.0040	.1820	.0710	.7420	0.014908		3260	1	2041	1	.00	29.00
							2495	0	2381	0	.00	23.00
>use												
82610	.0000	.0000	.9700	.0300	0.00145522	3260	1	3487	0	.00	24.00	
86007	.0000	.0140	.0120	.9740	0.00736606	3260	1	1786	1	6.00	20.00	
81971	.0580	.0130	.8610	.0670	0.00794225	3005	1	3544	0	.00	29.00	
85834	.8280	.1570	.0120	.0020	0.00827437	2552	0	2438	1	3.00	22.00	
85820	.0000	.0030	.1030	.8940	0.00837393	2920	1	2126	0	3.00	37.00	
85620	.0680	.9250	.0060	.0010	0.0138645	1928	0	1928	0	6.00	32.00	
85552	.0040	.1820	.0710	.7420	0.014908	3260	1	2041	1	.00	29.00	
88842	.0020	.4240	.4420	.1320	0.0222219	2495	0	2381	0	.00	23.00	
82730	.0130	.9860	.0000	.0010	0.0228809	3033	1	3317	0	.00	30.00	



Permutation

```
./mx63.bat perm.mx  
grep "of data" perm.mxo >> link"$b".txt  
done  
# _____  
done  
# _____
```

Link728.txt

```
-2 times log-likelihood of data >>> 10744.928  
-2 times log-likelihood of data >>> 10744.928  
-2 times log-likelihood of data >>> 10744.928  
-2 times log-likelihood of data >>> 10744.570  
-2 times log-likelihood of data >>> 10744.928  
-2 times log-likelihood of data >>> 10744.928  
-2 times log-likelihood of data >>> 10744.928  
-2 times log-likelihood of data >>> 10744.928  
-2 times log-likelihood of data >>> 10744.222  
-2 times log-likelihood of data >>> 10743.846
```



Other bits

- When working on servers
 - `bg &`
 - `fg`
 - `nohup`
 - `ctrl+c`
 - `ctrl+z`
 - `which`



Shutting down you unix session

- exit
- logout
- quit
- q

Time for coffee

