Cluster where, how and why?

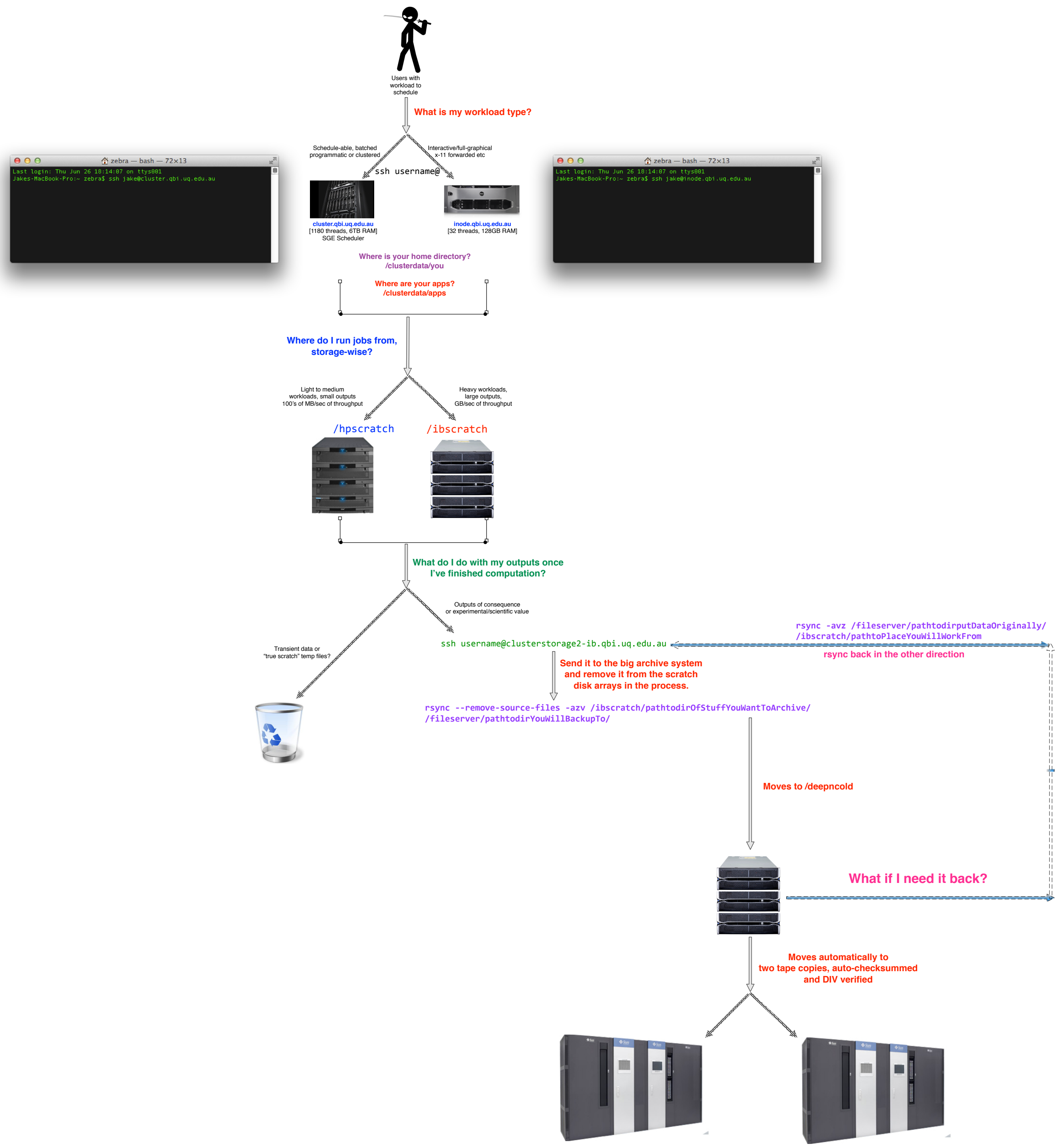
CNSG Teaching Scheme 2016

What will we cover?

- Structure and workflow
- Auto log-in, .bash_profile, .bashrc
- Paths, files and folders
- SGE qsub, array submission and interactive jobs
- Shell scripts and variables
- Job submission and monitoring
- Bash basics and advanced

Workflow and structure

QBI
HPC Workflow
Jake Carroll, Senior IT Manager
Version 1.0; 26/06/2014


Users with workload to schedule
What is my workload type?
Schedule-able, batched programmatic or clustered
Interactive/full-graphical x-11 forwarded etc
ssh username@
cluster.qbi.uq.edu.au
[1180 threads, 6TB RAM]
SGE Scheduler
inode.qbi.uq.edu.au
[32 threads, 128GB RAM]
Where is your home directory?
/clusterdata/you
Where are your apps?
/clusterdata/apps
Where do I run jobs from, storage-wise?
Light to medium workloads, small outputs 100's of MB/sec of throughput
Heavy workloads, large outputs, GB/sec of throughput
/hpscratch
/ibscratch
What do I do with my outputs once I've finished computation?
Transient data or "true scratch" temp files?
Outputs of consequence or experimental/scientific value
Send it to the big archive system and remove it from the scratch disk arrays in the process.
rsync back in the other direction
Moves to /deepncold
What if I need it back?
Moves automatically to two tape copies, auto-checksummed and DIV verified

```
zebra — bash — 72×13
Last login: Thu Jun 26 18:14:07 on ttys001
Jakes-MacBook-Pro:~ zebra$ ssh jake@cluster.qbi.uq.edu.au
```

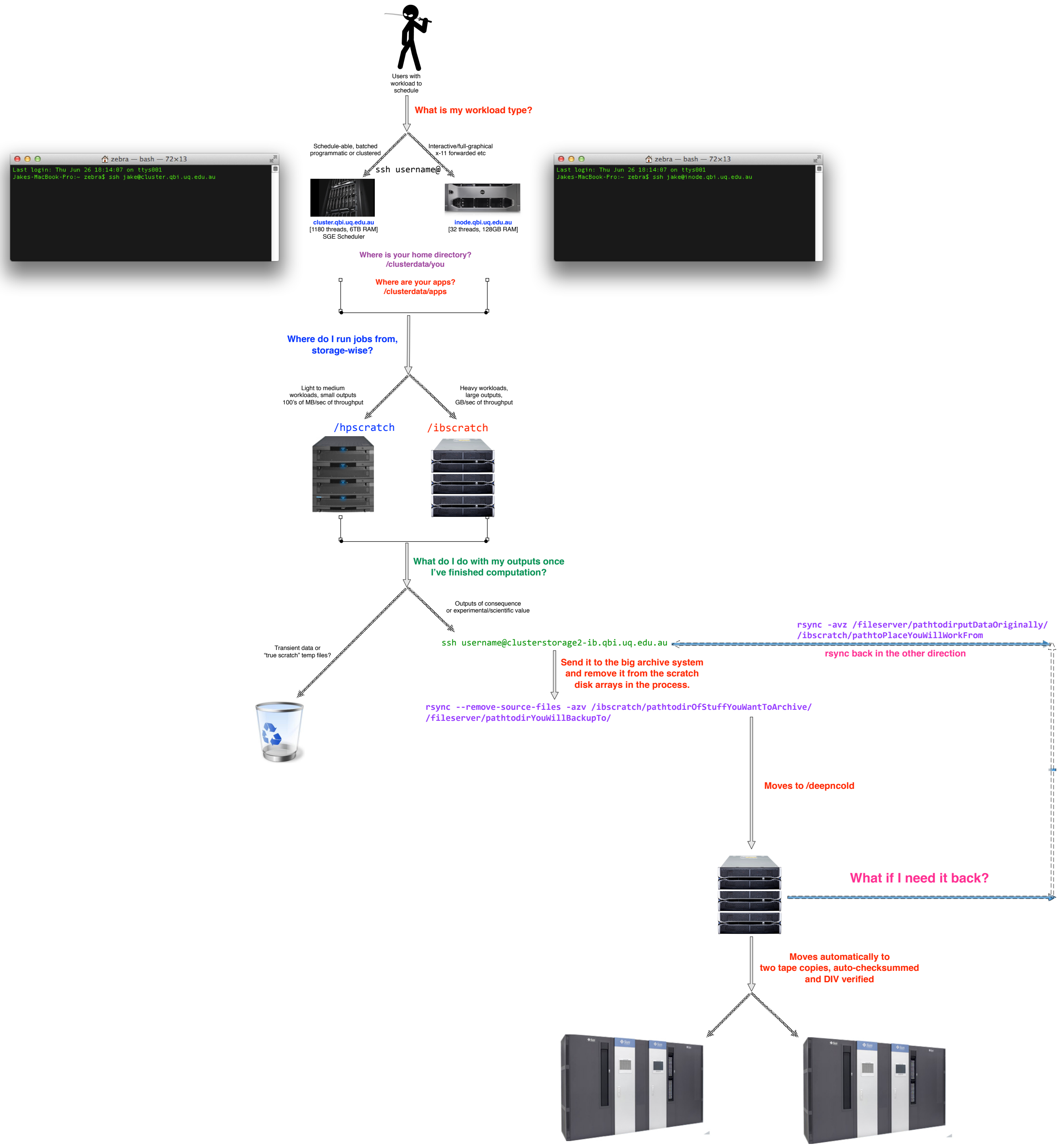
```
zebra — bash — 72×13
Last login: Thu Jun 26 18:14:07 on ttys001
Jakes-MacBook-Pro:~ zebra$ ssh jake@inode.qbi.uq.edu.au
```

```
ssh username@clusterstorage2-ib.qbi.uq.edu.au
```

```
rsync --remove-source-files -azv /ibscratch/pathtodirOfStuffYouWantToArchive/
/fileserver/pathtodirYouWillBackupTo/
```

```
rsync -avz /fileserver/pathtodirputDataOriginally/
/ibscratch/pathtoPlaceYouWillWorkFrom
```

QBI
HPC Workflow
Jake Carroll, Senior IT Manager
Version 1.0; 26/06/2014


Users with workload to schedule
What is my workload type?
Schedule-able, batched programmatic or clustered
Interactive/full-graphical x-11 forwarded etc
ssh username@
cluster.qbi.uq.edu.au
[1180 threads, 6TB RAM]
SGE Scheduler
inode.qbi.uq.edu.au
[32 threads, 128GB RAM]
zebra — bash — 72×13
Where is your home directory?
/clusterdata/you
Where are your apps?
/clusterdata/apps
Where do I run jobs from, storage-wise?
Light to medium workloads, small outputs 100's of MB/sec of throughput
Heavy workloads, large outputs, GB/sec of throughput
/hpscratch
/ibscratch
What do I do with my outputs once I've finished computation?
Transient data or "true scratch" temp files?
Outputs of consequence or experimental/scientific value
Send it to the big archive system and remove it from the scratch disk arrays in the process.
rsync back in the other direction
Moves to /deepncold
What if I need it back?
Moves automatically to two tape copies, auto-checksummed and DIV verified

```
Last login: Thu Jun 26 18:14:07 on ttys001
Jakes-MacBook-Pro:~ zebra$ ssh jake@cluster.qbi.uq.edu.au
```

```
Last login: Thu Jun 26 18:14:07 on ttys001
Jakes-MacBook-Pro:~ zebra$ ssh jake@inode.qbi.uq.edu.au
```

```
ssh username@clusterstorage2-ib.qbi.uq.edu.au
```

```
rsync -avz /fileserver/pathtodirputDataOriginally/
/ibscratch/pathtoPlaceYouWillWorkFrom
```

```
rsync --remove-source-files -azv /ibscratch/pathtodirOfStuffYouWantToArchive/
/fileserver/pathtodirYouWillBackupTo/
```

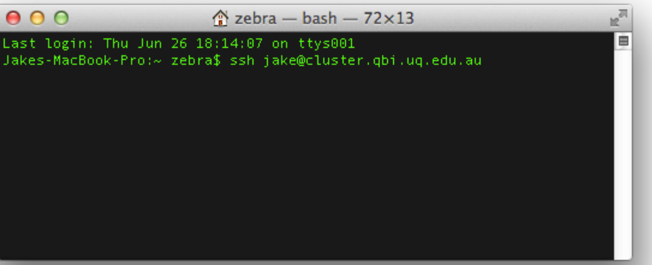
Users with workload to schedule

What is my workload type?

Schedule-able, batched programmatic or clustered

Interactive/full-graphical x-11 forwarded etc

ssh username@

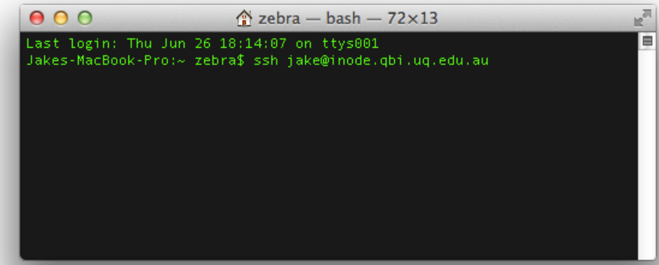


zebra — bash — 72×13

```
Last login: Thu Jun 26 18:14:07 on ttys001
Jakes-MacBook-Pro:~ zebra$ ssh jake@cluster.qbi.uq.edu.au
```

cluster.qbi.uq.edu.au
[1180 threads, 6TB RAM]
SGE Scheduler

inode.qbi.uq.edu.au
[32 threads, 128GB RAM]



zebra — bash — 72×13

```
Last login: Thu Jun 26 18:14:07 on ttys001
Jakes-MacBook-Pro:~ zebra$ ssh jake@inode.qbi.uq.edu.au
```

Where is your home directory?
/clusterdata/you

Where are your apps?
/clusterdata/apps

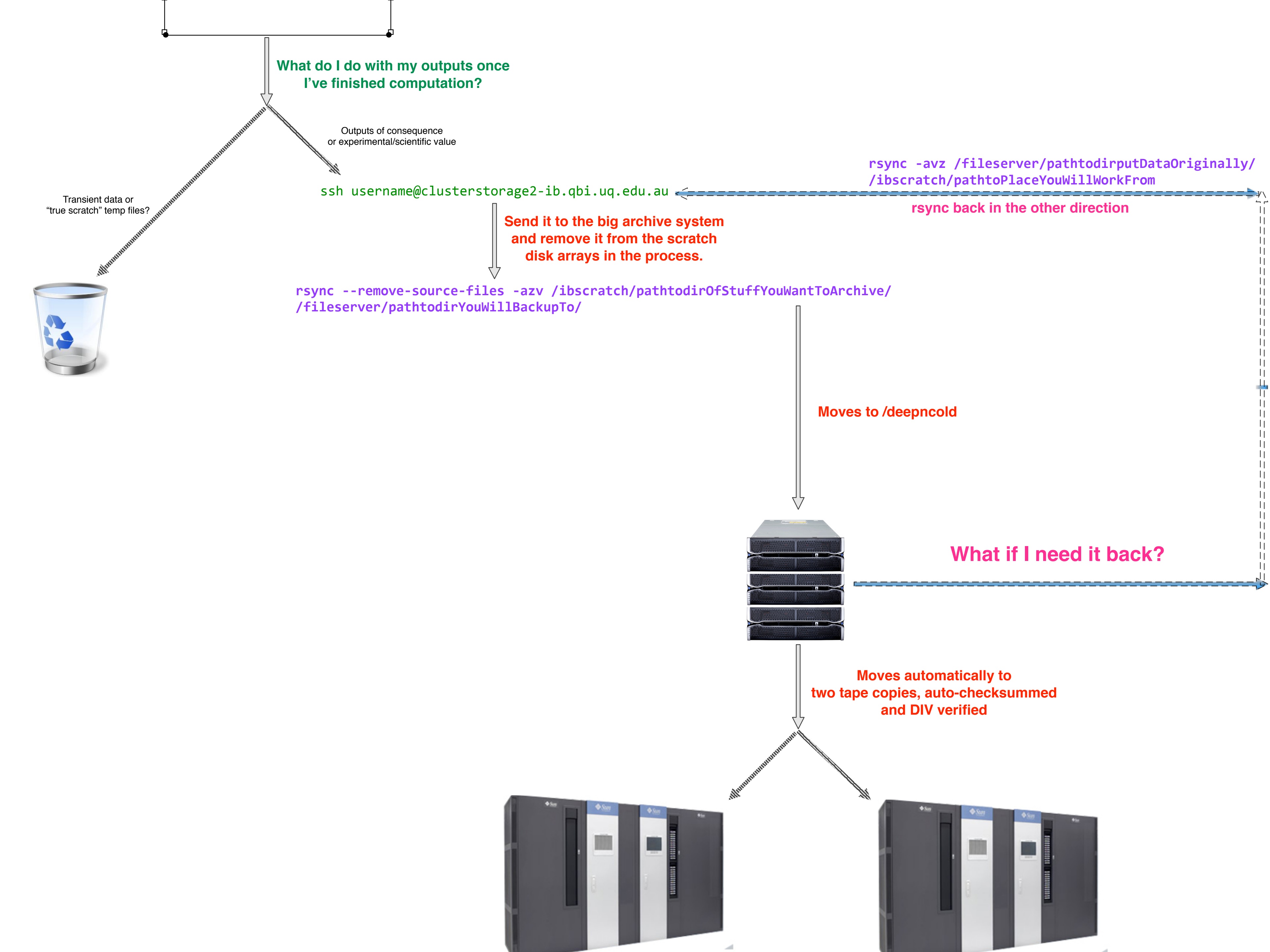
Where do I run jobs from, storage-wise?

Light to medium workloads, small outputs 100's of MB/sec of throughput

Heavy workloads, large outputs, GB/sec of throughput

/hpscratch

/ibscratch


What do I do with my outputs once I've finished computation?
Outputs of consequence or experimental/scientific value
Transient data or "true scratch" temp files?
Send it to the big archive system and remove it from the scratch disk arrays in the process.
Moves to /deepncold
What if I need it back?
rsync back in the other direction
Moves automatically to two tape copies, auto-checksummed and DIV verified

```
ssh username@clusterstorage2-ib.qbi.uq.edu.au
```

```
rsync --remove-source-files -azv /ibscratch/pathtodirOfStuffYouWantToArchive/
/fileserver/pathtodirYouWillBackupTo/
```

```
rsync -avz /fileserver/pathtodirputDataOriginally/
/ibscratch/pathtoPlaceYouWillWorkFrom
```

Basic log-in

- Basic protocol for remote connection uses Secure Shell (SSH) tunnels:

```
ssh username@cluster.qbi.uq.au
```

It connects via a secure port which by default is set to 22 but often this has to be specified:

```
ssh -p 22 username@cluster.qbi.uq.au
```

To take advantage of X11 graphical capabilities

```
ssh –Y username@cluster.qbi.uq.au
```

Auto log-in

- Step 1. On local machine generate public/private rsa key pair

```
ssh-keygen –t rsa –b 2048
```

follow onscreen instructions:

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/username/.ssh/id_rsa):  <change if necessary>
Enter passphrase (empty for no passphrase):  <password>
Enter same passphrase again:  <password>
Your identification has been saved in /home/username/.ssh/id_rsa.
Your public key has been saved in /home/username/.ssh/id_rsa.pub.
```

- Step 2. Copy the **public** key to the remote server:

```
scp /home/uqmtrzas/.ssh/id_rsa.pub uqmtrzas@cluster.qbi.uq.au:~/
```

Auto log-in

- Step 3. Log in to the server:

```
ssh uqmtrzas@cluster.qbi.uq.edu.au
```

- Step 4. copy content of the public key to authorized keys:

```
cat ~/id_rsa.pub >> .ssh/authorize_keys
```

- Step 5. delete and log out:

```
rm ~/id_rsa.pub
logout
```

- Step 6. log back in to the server:

```
ssh uqmtrzas@cluster.qbi.uq.edu.au
```

Setting up environment (.bash_profile, .bashrc)

- .bash_profile is automatically read by bash making its content available upon login
- .bashrc can be added and sourced by specifying it in .bash_profile

```
if [ -f ~/.bashrc ]
then
        .~/.bashrc
fi
```

Setting up environment (.bash_profile, .bashrc)

- Adding paths to working directory:

```
PATH=$PATH:$HOME/bin:$HOME/library
```

Or

```
export PATH="/clusterdata/apps/R-3.1.2/bin:$PATH"
```

- Adding alieses:

```
lsm="ls –lah --color=auto"
cluster="ssh –Y uqmtrzas@cluster.qbi.uq.edu.au"
```

Paths, files and folders

- Extremely important to have a good organisational ethics. Keep it:
    - Simple (avoid non-informative names)
    - Explicit (use dates to track progress in long projects)
    - Consistent (find your style and stick to it)

Paths, files and folders

ROOT	mdd	out	res_160518
	original_data	res_160518	plots
	qced_data	res_160519	data
mdd	log	res_160520	
	out		
	src		

qsub, array jobs and interactive jobs

- Interactive jobs use ‘inode’
  - NOTE: use only when developing a pipeline or running a simple R job
- qsub (SGE submitted jobs) ‘cluster’
  - NOTE: computationally demanding jobs and in particular job that can be run in parallel environment (MOST work that we do!)

cluster structure

To see structure of your cluster what nodes are available, how busy they are use: qhost

```
HOSTNAME            ARCH       NCPU  LOAD  MEMTOT  MEMUSE  SWAPTO  SWAPUS
-------------------------------------------------------------------------------
global              -             -     -     -       -       -       -
compute-0-0         lx26-amd64    24 10.26   94.6G   16.4G  996.2M   37.1M
compute-0-1         lx26-amd64    24 12.36   94.6G   28.6G  996.2M   38.5M
compute-0-10        lx26-amd64    24 14.05   94.6G   19.5G  996.2M   92.2M
compute-0-11        lx26-amd64    24  9.27   94.6G   15.2G  996.2M   37.9M
compute-0-12        lx26-amd64    24  6.58   94.6G   34.9G  996.2M   40.6M
compute-0-13        lx26-amd64    24 10.63   94.6G   14.4G  996.2M   34.1M
compute-0-14        lx26-amd64    24 11.33   94.6G   27.6G  996.2M   55.0M
compute-0-15        lx26-amd64    24 13.04   94.6G   14.7G  996.2M   29.1M
compute-0-2         lx26-amd64    24 11.56   94.6G   12.2G  996.2M   35.7M
compute-0-3         lx26-amd64    24 13.53   94.6G   33.6G  996.2M  154.9M
compute-0-4         lx26-amd64    24 12.07   94.6G   50.6G  996.2M   36.3M
compute-0-5         lx26-amd64    24  9.96   94.6G   37.5G  996.2M   70.5M
compute-0-6         lx26-amd64    24 15.71   94.6G   36.2G  996.2M   38.5M
compute-0-7         lx26-amd64    24 12.32   94.6G   28.4G  996.2M   30.3M
compute-0-8         lx26-amd64    24 18.87   94.6G   48.8G  996.2M   60.0M
compute-0-9         lx26-amd64    24  7.30   94.6G   21.4G  996.2M   30.4M
compute-1-0         lx26-amd64    80 43.12  378.7G  125.5G  996.2M   11.4M
compute-1-1         lx26-amd64    80 36.14  378.7G  155.4G  996.2M   78.3M
compute-1-2         lx26-amd64    80 21.48  378.7G   63.0G  996.2M   16.8M
compute-1-3         lx26-amd64    80 33.83  378.7G   60.2G  996.2M   37.7M
compute-1-4         lx26-amd64    80 19.66  378.7G  159.2G  996.2M  337.6M
compute-1-5         lx26-amd64    80 29.52  378.7G  210.3G  996.2M  636.5M
compute-1-6         lx26-amd64    80 23.43  378.7G  132.6G  996.2M   33.6M
compute-1-7         lx26-amd64    80 33.03  378.7G   61.3G  996.2M   30.4M
compute-2-0         lx26-amd64    32 24.53   47.2G   18.6G 1000.0M   84.9M
compute-2-1         lx26-amd64    32 23.56   47.2G   19.1G 1000.0M   44.1M
compute-2-2         lx26-amd64    32 20.10   47.2G   18.0G 1000.0M   34.7M
compute-2-3         lx26-amd64    32 17.01   47.2G   26.8G 1000.0M   34.6M
compute-2-4         lx26-amd64    48 25.89  252.4G   95.6G 1000.0M   97.2M
compute-2-5         lx26-amd64    48 29.31  252.4G   81.7G 1000.0M   40.0M
compute-2-6         lx26-amd64    48 27.07  252.4G  109.9G 1000.0M  330.1M
```

qsub

```
qsub -q <queue> -w e -N <job_name> -l h_vmem=<memory, e.g. 4G> -l h_rt=<time> -l s_rt=<time> -pe smp <num_slots> -o <outputlogfile> -e <errorlogfile> <pathtoScript> <arg1> <arg2>
```

-q <queue> set the queue. Often you will use the standard queue, so no need to set this up.

-V will pass all environment variables to the job

-v var[=value] will specifically pass environment variable 'var' to the job

-b y allow command to be a binary file instead of a script.

-w e verify options and abort if there is an error

-N <jobname> name of the job. This you will see when you use qstat, to check status of your jobs.

-l h_vmem=size specify the amount of maximum memory required (e.g. 3G or 3500M) (NOTE: This is memory per processor slot. So if you ask for 2 processors total memory will be 2 * hvmem_value)

-l h_rt=<hh:mm:ss> specify the maximum run time (hours, minutes and seconds)

-l s_rt=hh:mm:ss specify the soft run time limit (hours, minutes and seconds) - Remember to set both s_rt and h_rt.

-pe smp <n_slots> This specifies the parallel environment. smp runs a parallel job using shared-memory and n_processors amount of cores.

-cwd run in current working directory

-wd <dir> Set working directory for this job as <dir>

-o <output_logfile> name of the output log file

-e <error_logfile> name of the error log file

-m ea Will send email when job ends or aborts

-P <projectName> set the job's project

-M <emailaddress> Email address to send email to

-t <start>-<end>:<incr> submit a job array with start index , stop index in increments using

qsub

- Shell script:

```
for CHR in {1..22}
do
   echo "plink --bfile mdd/data/mdd_geno --chr ${CHR} --assoc --out mdd/out/res_160518/
mdd_gwa_chr${CHR} > mdd/out/res_160518/mdd_gwa_chr${CHR}.log" > mdd/src/
run_gwas_chr${CHR}.
done
```

- Single job submission:

```
for CHR in $(seq 1 22)
do
        sleep 2
        qsub -cwd -N mdd_gwas_chr${CHR} -l h_vmem=3G -j y -o mdd/log/ ./run_gwas_${CHR}.sh
done
```

- Array job submission:

```
qsub -cwd -t 1-22:1 -N mdd_gwas_chr${SGE_TASK_ID} -l h_vmem=3G -j y -o mdd/log/ ./
run_gwas_${SGE_TASK_ID}
```

more practical qsub

- Make qsub script which you can reuse:

```
cat >>

#!/bin/bash
scriptname=$(mktemp)
if [ -z $4 ]; then
        name=$(basename ${scriptname})
else
        name=$4
fi
mkdir -p qsub_reports

echo '#!/bin/bash' > $scriptname
echo "#$ -N ${name}" >> $scriptname
echo "#$ -S /bin/bash" >> $scriptname
echo "#$ -o qsub_reports/" >> $scriptname
echo "#$ -e qsub_reports/" >> $scriptname
echo "#$ -cwd" >> $scriptname
echo "#$ -pe onehost $2" >> $scriptname
echo "#$ -l h_vmem=$3G" >> $scriptname
echo "#$ -w e" >> $scriptname
echo $1 >> $scriptname

qsub $scriptname

>>  submit.sh
```

Example: `submit.sh "plink --bfile mdd/data/mdd_geno --assoc --out mdd/out/res_160518/mdd_gwa > mdd/out/res_160518/mdd_gwa.log" > mdd/src/run_gwas" 1 3 mdd_gwa.out`

looking up active jobs

- to see what jobs are running and how many of them use
  qstat

```
job-ID  prior   name       user       state submit/start at     queue                          slots ja-task-ID
-----------------------------------------------------------------------------------------------------------------
8974394 0.00000 pract1.job uqmtrzas     qw    05/17/2016 14:32:05                                 1
8974395 0.00000 prac2.sh   uqmtrzas     hqw   05/17/2016 14:32:05                                 1
```

job monitoring / modifications

- You can place a submitted job on hold using

```
qhold <job_id>
```

- Job that has been put on hold can be released using:

```
qrls <job_id>          (also can be applied to array jobs)
qalter -h U
```

- You can condition on job on another:

```
submit.sh "zcat MDD_CONV_2Sep2015-INTERNAL.txt.gz | awk '(\$1==22){print \$1,\$2,\
$3,\$4,\$8}' > chr22.tmp" 1 1 pract1.job
echo "wc -l chr22.tmp" > prac2.sh
qsub -cwd -hold_jid "pract1.job" -N pract2.job -l h_vmem=1G -j y -o mdd/log/ ./prac2.sh
```

job monitoring / modifications

- You can watch progress of your job using:

  watch 'qstat'

```
job-ID  prior   name       user       state submit/start at     queue                     slots ja-task-ID
-----------------------------------------------------------------------------------------------------------------
8974394 0.00000 pract1.job uqmtrzas    qw    05/17/2016 14:32:05                              1
8974395 0.00000 prac2.sh   uqmtrzas    hqw   05/17/2016 14:32:05                              1
```

- You can delete running job with qdel
- You can check the status or your last job: qacct

qacct -j 'pract1.job'

```
==============================================================
qname        medium.q
hostname     compute-2-3.local
group        wrayvisscher
owner        uqmtrzas
project      NONE
department   defaultdepartment
jobname      pract1.job
jobnumber    8974402
taskid       undefined
account      sge
priority     0
qsub_time    Tue May 17 14:39:49 2016
start_time   Tue May 17 14:40:01 2016
end_time     Tue May 17 14:40:06 2016
granted_pe   NONE
slots        1
failed       0
exit_status  0
ru_wallclock 5
ru_utime     6.458
ru_stime     0.245
ru_maxrss    2700
ru_ixrss     0
ru_ismrss    0
ru_idrss     0
ru_isrss     0
ru_minflt    39316
ru_majflt    1
ru_nswap     0
ru_inblock   211736
ru_oublock   5360
ru_msgsnd    0
ru_msgrcv    0
ru_nsignals  0
ru_nvcsw     14066
ru_nivcsw    214
cpu          6.703
mem          0.249
io           0.856
iow          0.000
maxvmem      216.391M
arid         undefined
```

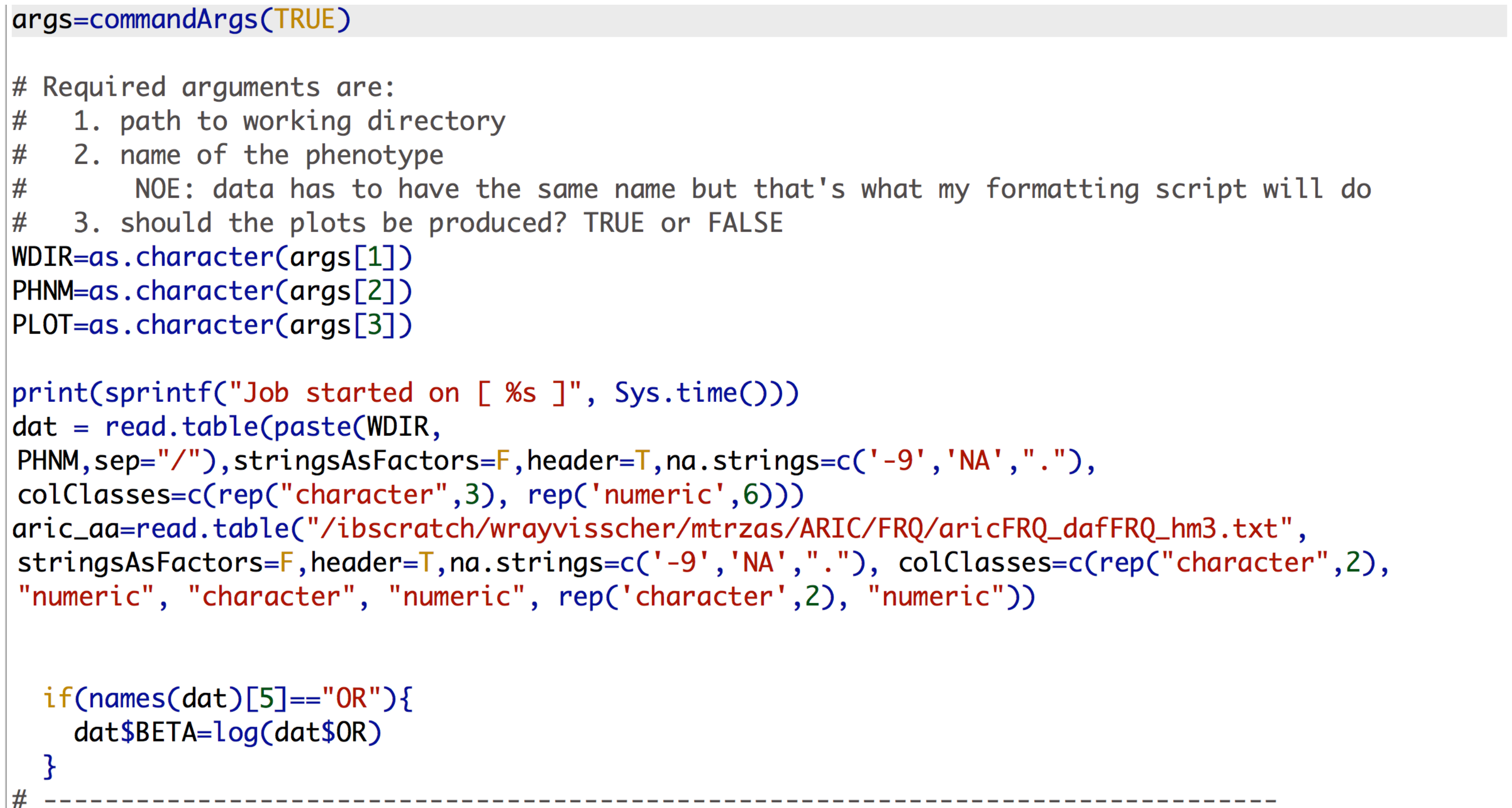
Moving files

- Files can be copied to and from the remote server using two protocols:

```
scp ~/Documents/myfile   cluster:~/
rsync -avzhe ssh --progress ~/Documents/myfile   cluster:~/
```

Submitting R jobs via qsub

- Write an R script and make sure it works perfectly.
- Change fixed values by variable name and add to the top of the script:



```
args=commandArgs(TRUE)

# Required arguments are:
#   1. path to working directory
#   2. name of the phenotype
#       NOE: data has to have the same name but that's what my formatting script will do
#   3. should the plots be produced? TRUE or FALSE
WDIR=as.character(args[1])
PHNM=as.character(args[2])
PLOT=as.character(args[3])

print(sprintf("Job started on [ %s ]", Sys.time()))
dat = read.table(paste(WDIR,
 PHNM,sep="/"),stringsAsFactors=F,header=T,na.strings=c('-9','NA',"."),
 colClasses=c(rep("character",3), rep('numeric',6)))
aric_aa=read.table("/ibscratch/wrayvisscher/mtrzas/ARIC/FRQ/aricFRQ_dafFRQ_hm3.txt",
 stringsAsFactors=F,header=T,na.strings=c('-9','NA',"."), colClasses=c(rep("character",2),
 "numeric", "character", "numeric", rep('character',2), "numeric"))


  if(names(dat)[5]=="OR"){
    dat$BETA=log(dat$OR)
  }
# ------------------------------------------------------------------
```

Submitting R jobs via qsub

- Submit using standard qsub and calling 'Rscript':

```
DIR="/clusterdata/apps/R-2.15.1/bin"
WRKD="/ibscratch/wrayvisscher/mtrzas/Sleep"
submit.sh "${DIR}/Rscript ${WRKD}/src/qc_check_sumstats.sh ${WRKD}
Sleep_duration TRUE" 1 2 sleep.check.log
```

Bash basics

- creating files and folders:

```
mkdir   <folder name>
touch   <file name>
vi      <file name>
```

- Finding files (useful particularly when needing a list with files and full paths to them, for example for meta-analysis)

```
DIR="/home/uqmtrzas/test_project"
find ${DIR}/ -name "*.qassoc" -exec ls '{}' \; > meta.list
```

Bash advanced

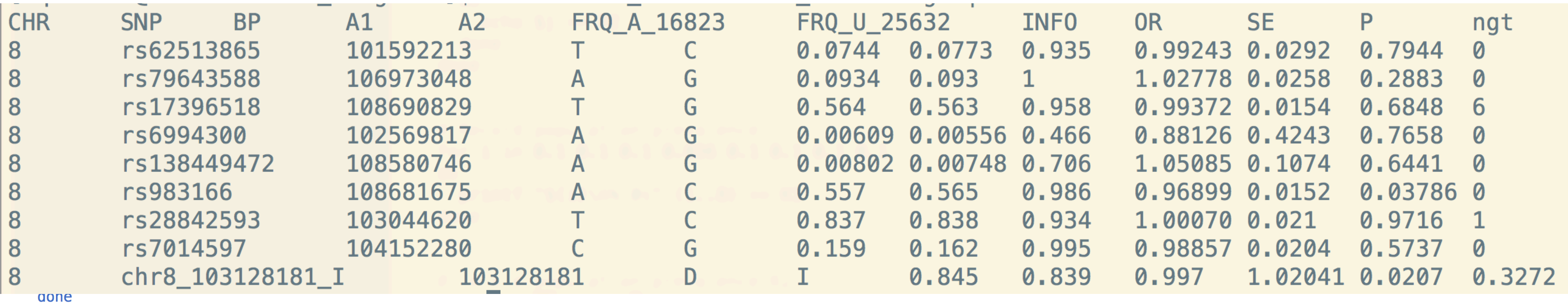
Variables and arrays

```
# Calculate sample prevalence
for FLE in $(ls ${WRKD}/MDD_latest_data_250216/data)
do
  OUT=$(echo $FLE | sed 's/daner_//;s/.gz//')
  nca=$(zcat ${WRKD}/MDD_latest_data_250216/data/${FLE} | sed 1q | awk '{print $6}' | sed 's/FRQ_A_//')
  nco=$(zcat ${WRKD}/MDD_latest_data_250216/data/${FLE} | sed 1q | awk '{print $7}' | sed 's/FRQ_U_//')
  echo -n "$OUT " >> ${WRKD}/MDD_latest_data_250216/allSamplePrevalence
  echo "scale=7;${nca}/(${nca}+${nco})" | bc >> ${WRKD}/MDD_latest_data_250216/allSamplePrevalence
done




# record sample prevalence P1
for i in $(awk '{print $2}' allSamplePrevalence)
do
  printf "${i}\n%.0s" {1..8} >> P1
done

# record sample prevalence P2
for i in $(seq 1 8)
do
  for j in $(awk '{print $2}' allSamplePrevalence)
  do
    echo $j >> P2
  done
done


# Record population prevalence
for i in 0.1 0.1 0.1 0.036 0.1 0.1 0.1 0.1
do
  printf "${i}\n%.0s" {1..8} >> K1
done


# record population prevalence K2
for i in $(seq 1 8)
do
  for j in 0.1 0.1 0.1 0.036 0.1 0.1 0.1 0.1
  do
    echo $j >> K2
  done
done


# read values from all the files into bash arrays
readarray P1 < P1
readarray P2 < P2
readarray K1 < K1
readarray K2 < K2
```

Bash advanced

Variables and arrays

```
# Calculate sample prevalence
for FLE in $(ls ${WRKD}/MDD_latest_data_250216/data)
do
  OUT=$(echo $FLE | sed 's/daner_//;s/.gz//')
  nca=$(zcat ${WRKD}/MDD_latest_data_250216/data/${FLE} | sed 1q | awk '{print $6}' | sed 's/FRQ_A_//')
  nco=$(zcat ${WRKD}/MDD_latest_data_250216/data/${FLE} | sed 1q | awk '{print $7}' | sed 's/FRQ_U_//')
  echo -n "$OUT " >> ${WRKD}/MDD_latest_data_250216/allSamplePrevalence
  echo "scale=7;${nca}/(${nca}+${nco})" | bc >> ${WRKD}/MDD_latest_data_250216/allSamplePrevalence
done
```

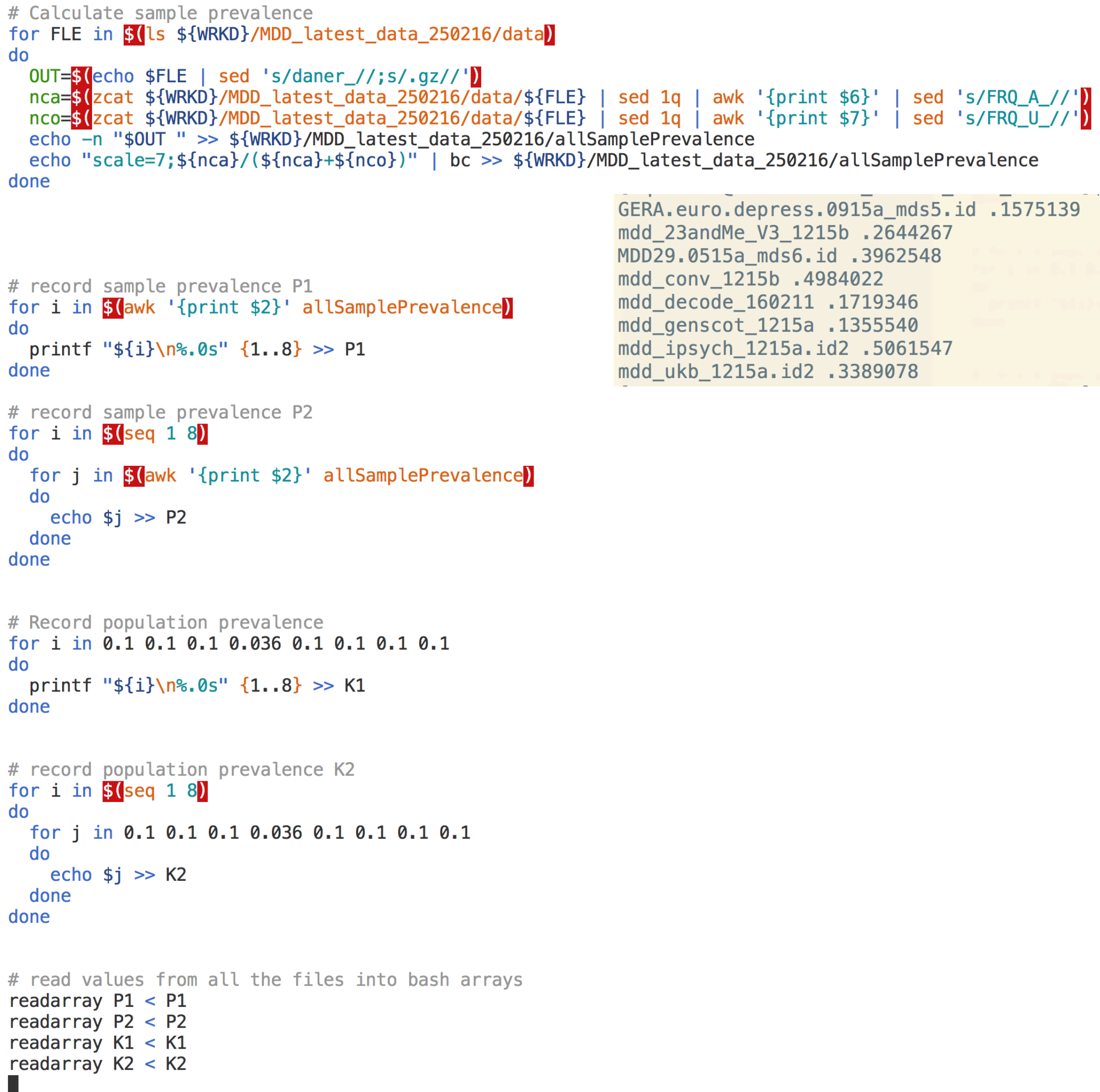


```
CHR     SNP     BP      A1      A2      FRQ_A_16823     FRQ_U_25632     INFO    OR      SE      P       ngt
8       rs62513865      101592213       T       C       0.0744  0.0773  0.935   0.99243 0.0292  0.7944  0
8       rs79643588      106973048       A       G       0.0934  0.093   1       1.02778 0.0258  0.2883  0
8       rs17396518      108690829       T       G       0.564   0.563   0.958   0.99372 0.0154  0.6848  6
8       rs6994300       102569817       A       G       0.00609 0.00556 0.466   0.88126 0.4243  0.7658  0
8       rs138449472     108580746       A       G       0.00802 0.00748 0.706   1.05085 0.1074  0.6441  0
8       rs983166        108681675       A       C       0.557   0.565   0.986   0.96899 0.0152  0.03786 0
8       rs28842593      103044620       T       C       0.837   0.838   0.934   1.00070 0.021   0.9716  1
8       rs7014597       104152280       C       G       0.159   0.162   0.995   0.98857 0.0204  0.5737  0
8       chr8_103128181_I        103128181       D       I       0.845   0.839   0.997   1.02041 0.0207  0.3272
```

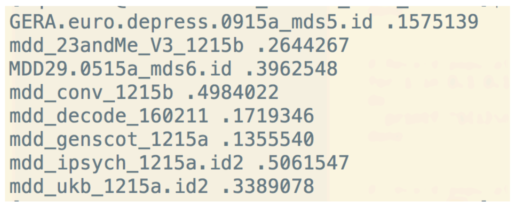
```
done


# Record population prevalence
for i in 0.1 0.1 0.1 0.036 0.1 0.1 0.1 0.1
do
  printf "${i}\n%.0s" {1..8} >> K1
done


# record population prevalence K2
for i in $(seq 1 8)
do
  for j in 0.1 0.1 0.1 0.036 0.1 0.1 0.1 0.1
  do
    echo $j >> K2
  done
done


# read values from all the files into bash arrays
readarray P1 < P1
readarray P2 < P2
readarray K1 < K1
readarray K2 < K2
```

Bash advanced

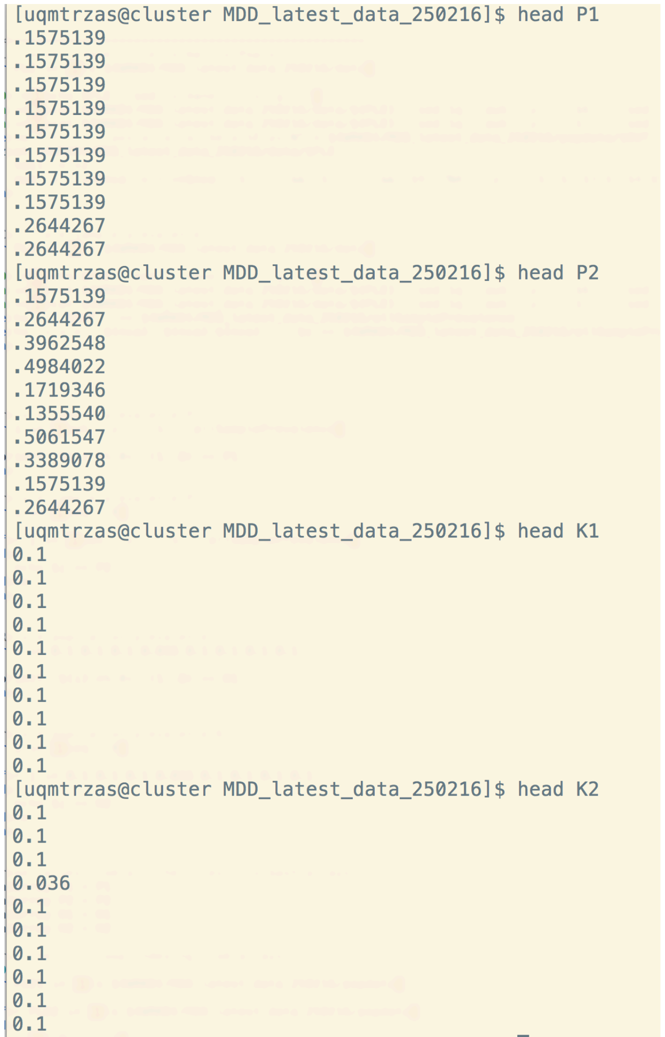
Variables and arrays



```
# Calculate sample prevalence
for FLE in $(ls ${WRKD}/MDD_latest_data_250216/data)
do
  OUT=$(echo $FLE | sed 's/daner_//;s/.gz//')
  nca=$(zcat ${WRKD}/MDD_latest_data_250216/data/${FLE} | sed 1q | awk '{print $6}' | sed 's/FRQ_A_//')
  nco=$(zcat ${WRKD}/MDD_latest_data_250216/data/${FLE} | sed 1q | awk '{print $7}' | sed 's/FRQ_U_//')
  echo -n "$OUT " >> ${WRKD}/MDD_latest_data_250216/allSamplePrevalence
  echo "scale=7;${nca}/(${nca}+${nco})" | bc >> ${WRKD}/MDD_latest_data_250216/allSamplePrevalence
done


# record sample prevalence P1
for i in $(awk '{print $2}' allSamplePrevalence)
do
  printf "${i}\n%.0s" {1..8} >> P1
done

# record sample prevalence P2
for i in $(seq 1 8)
do
  for j in $(awk '{print $2}' allSamplePrevalence)
  do
    echo $j >> P2
  done
done


# Record population prevalence
for i in 0.1 0.1 0.1 0.036 0.1 0.1 0.1 0.1
do
  printf "${i}\n%.0s" {1..8} >> K1
done


# record population prevalence K2
for i in $(seq 1 8)
do
  for j in 0.1 0.1 0.1 0.036 0.1 0.1 0.1 0.1
  do
    echo $j >> K2
  done
done


# read values from all the files into bash arrays
readarray P1 < P1
readarray P2 < P2
readarray K1 < K1
readarray K2 < K2
```

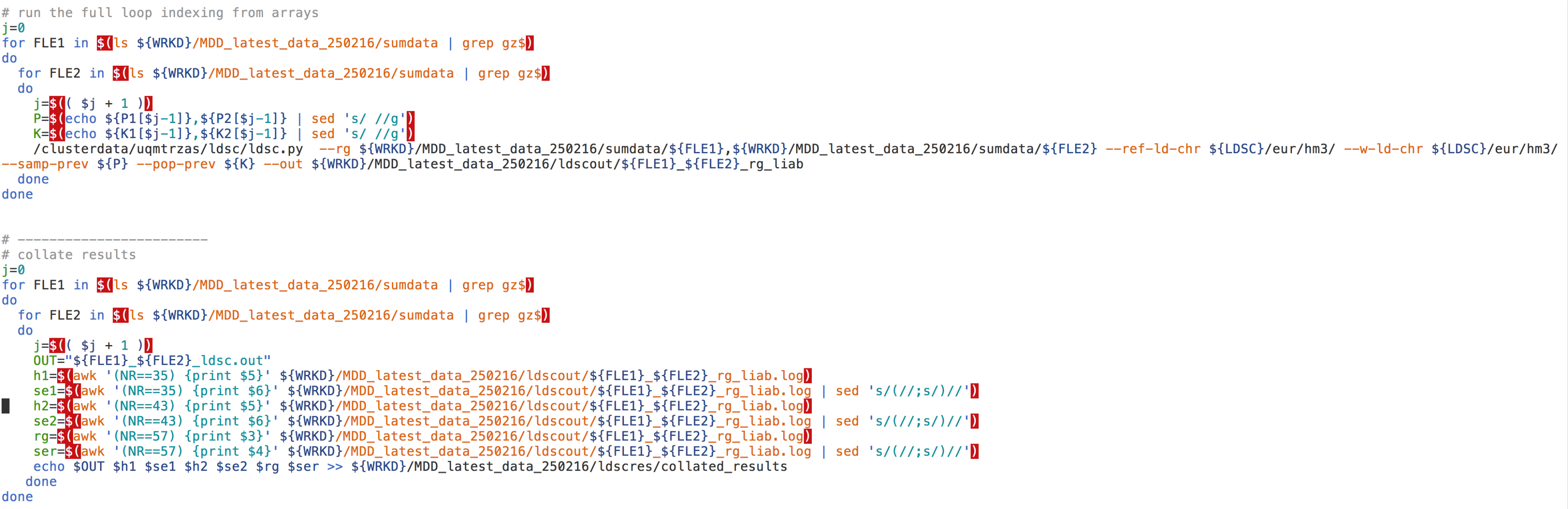


```
GERA.euro.depress.0915a_mds5.id .1575139
mdd_23andMe_V3_1215b .2644267
MDD29.0515a_mds6.id .3962548
mdd_conv_1215b .4984022
mdd_decode_160211 .1719346
mdd_genscot_1215a .1355540
mdd_ipsych_1215a.id2 .5061547
mdd_ukb_1215a.id2 .3389078
```

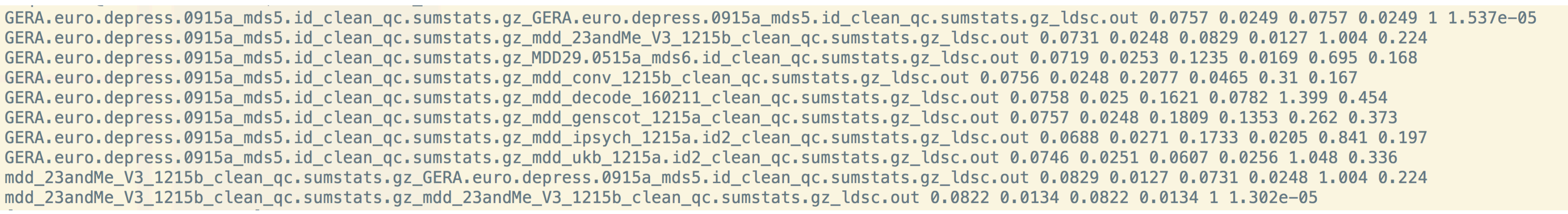


```
[uqmtrzas@cluster MDD_latest_data_250216]$ head P1
.1575139
.1575139
.1575139
.1575139
.1575139
.1575139
.1575139
.1575139
.2644267
.2644267
[uqmtrzas@cluster MDD_latest_data_250216]$ head P2
.1575139
.2644267
.3962548
.4984022
.1719346
.1355540
.5061547
.3389078
.1575139
.2644267
[uqmtrzas@cluster MDD_latest_data_250216]$ head K1
0.1
0.1
0.1
0.1
0.1
0.1
0.1
0.1
0.1
0.1
[uqmtrzas@cluster MDD_latest_data_250216]$ head K2
0.1
0.1
0.1
0.036
0.1
0.1
0.1
0.1
0.1
0.1
```

Bash advanced



```
# run the full loop indexing from arrays
j=0
for FLE1 in $(ls ${WRKD}/MDD_latest_data_250216/sumdata | grep gz$)
do
  for FLE2 in $(ls ${WRKD}/MDD_latest_data_250216/sumdata | grep gz$)
  do
    j=$(( $j + 1 ))
    P=$(echo ${P1[$j-1]},${P2[$j-1]} | sed 's/ //g')
    K=$(echo ${K1[$j-1]},${K2[$j-1]} | sed 's/ //g')
    /clusterdata/uqmtrzas/ldsc/ldsc.py  --rg ${WRKD}/MDD_latest_data_250216/sumdata/${FLE1},${WRKD}/MDD_latest_data_250216/sumdata/${FLE2} --ref-ld-chr ${LDSC}/eur/hm3/ --w-ld-chr ${LDSC}/eur/hm3/
--samp-prev ${P} --pop-prev ${K} --out ${WRKD}/MDD_latest_data_250216/ldscout/${FLE1}_${FLE2}_rg_liab
  done
done


# ------------------------
# collate results
j=0
for FLE1 in $(ls ${WRKD}/MDD_latest_data_250216/sumdata | grep gz$)
do
  for FLE2 in $(ls ${WRKD}/MDD_latest_data_250216/sumdata | grep gz$)
  do
    j=$(( $j + 1 ))
    OUT="${FLE1}_${FLE2}_ldsc.out"
    h1=$(awk '(NR==35) {print $5}' ${WRKD}/MDD_latest_data_250216/ldscout/${FLE1}_${FLE2}_rg_liab.log)
    se1=$(awk '(NR==35) {print $6}' ${WRKD}/MDD_latest_data_250216/ldscout/${FLE1}_${FLE2}_rg_liab.log | sed 's/(//;s/)//')
    h2=$(awk '(NR==43) {print $5}' ${WRKD}/MDD_latest_data_250216/ldscout/${FLE1}_${FLE2}_rg_liab.log)
    se2=$(awk '(NR==43) {print $6}' ${WRKD}/MDD_latest_data_250216/ldscout/${FLE1}_${FLE2}_rg_liab.log | sed 's/(//;s/)//')
    rg=$(awk '(NR==57) {print $3}' ${WRKD}/MDD_latest_data_250216/ldscout/${FLE1}_${FLE2}_rg_liab.log)
    ser=$(awk '(NR==57) {print $4}' ${WRKD}/MDD_latest_data_250216/ldscout/${FLE1}_${FLE2}_rg_liab.log | sed 's/(//;s/)//')
    echo $OUT $h1 $se1 $h2 $se2 $rg $ser >> ${WRKD}/MDD_latest_data_250216/ldscres/collated_results
   done
done
```



```
GERA.euro.depress.0915a_mds5.id_clean_qc.sumstats.gz_GERA.euro.depress.0915a_mds5.id_clean_qc.sumstats.gz_ldsc.out 0.0757 0.0249 0.0757 0.0249 1 1.537e-05
GERA.euro.depress.0915a_mds5.id_clean_qc.sumstats.gz_mdd_23andMe_V3_1215b_clean_qc.sumstats.gz_ldsc.out 0.0731 0.0248 0.0829 0.0127 1.004 0.224
GERA.euro.depress.0915a_mds5.id_clean_qc.sumstats.gz_MDD29.0515a_mds6.id_clean_qc.sumstats.gz_ldsc.out 0.0719 0.0253 0.1235 0.0169 0.695 0.168
GERA.euro.depress.0915a_mds5.id_clean_qc.sumstats.gz_mdd_conv_1215b_clean_qc.sumstats.gz_ldsc.out 0.0756 0.0248 0.2077 0.0465 0.31 0.167
GERA.euro.depress.0915a_mds5.id_clean_qc.sumstats.gz_mdd_decode_160211_clean_qc.sumstats.gz_ldsc.out 0.0758 0.025 0.1621 0.0782 1.399 0.454
GERA.euro.depress.0915a_mds5.id_clean_qc.sumstats.gz_mdd_genscot_1215a_clean_qc.sumstats.gz_ldsc.out 0.0757 0.0248 0.1809 0.1353 0.262 0.373
GERA.euro.depress.0915a_mds5.id_clean_qc.sumstats.gz_mdd_ipsych_1215a.id2_clean_qc.sumstats.gz_ldsc.out 0.0688 0.0271 0.1733 0.0205 0.841 0.197
GERA.euro.depress.0915a_mds5.id_clean_qc.sumstats.gz_mdd_ukb_1215a.id2_clean_qc.sumstats.gz_ldsc.out 0.0746 0.0251 0.0607 0.0256 1.048 0.336
mdd_23andMe_V3_1215b_clean_qc.sumstats.gz_GERA.euro.depress.0915a_mds5.id_clean_qc.sumstats.gz_ldsc.out 0.0829 0.0127 0.0731 0.0248 1.004 0.224
mdd_23andMe_V3_1215b_clean_qc.sumstats.gz_mdd_23andMe_V3_1215b_clean_qc.sumstats.gz_ldsc.out 0.0822 0.0134 0.0822 0.0134 1 1.302e-05
```

- Getting extra tips:

  http://intranet.qbi.uq.edu.au/technical-services/it/qbi-cluster