GWAS QC + PCA

Connecting to the cluster

Go ahead and connect to the HPC:

```
# remember to use your actual username
ssh USERNAME@kennedy.st-andrews.ac.uk
```

(or use PuTTY on windows)

Biologists unprepared for big data

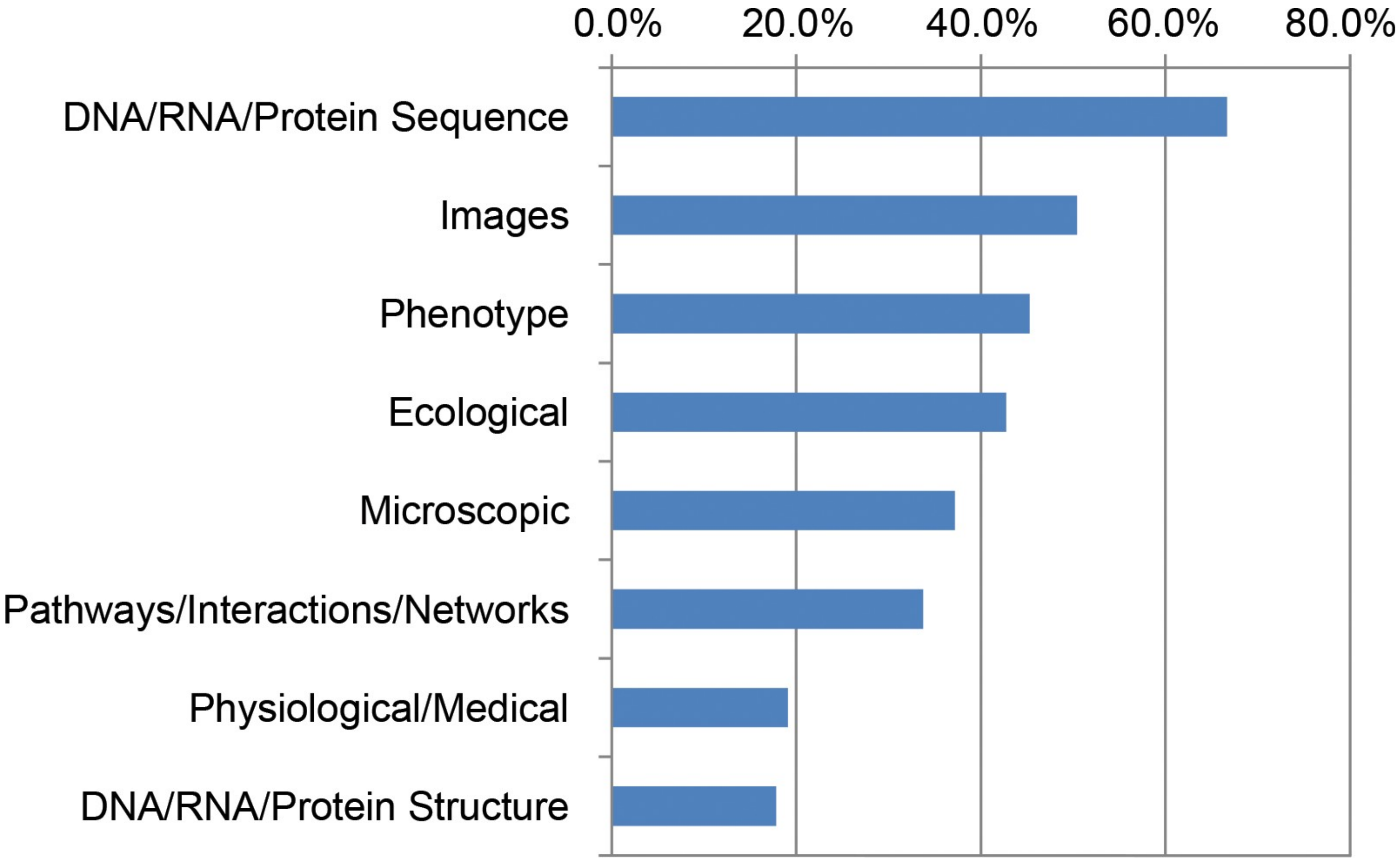
94% of students/faculty/researchers use large data sets or will in the near future (n = 1,097)

47% rated their bioinformatics skill level as “beginner,” (n = 608)

58% felt their institutions do not provide all the computational resources needed for their research (n = 1,024)

https://doi.org/10.1371/journal.pcbi.1005755

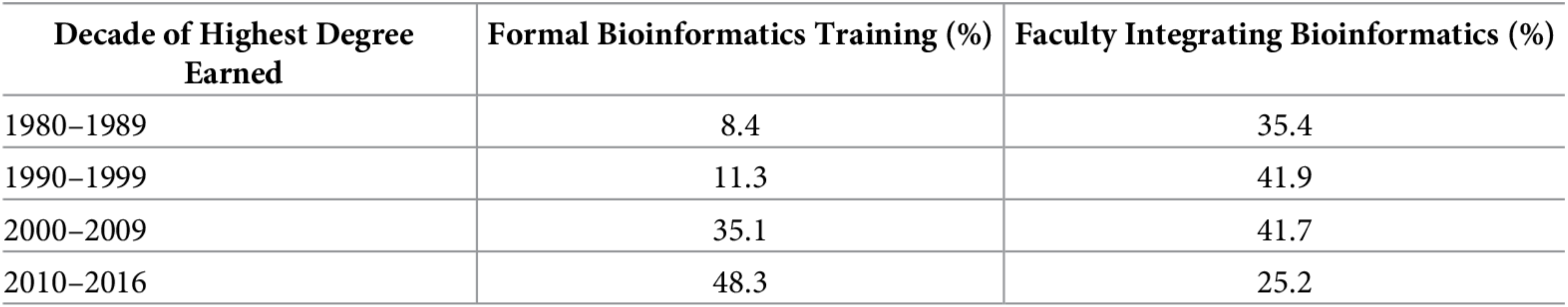
Biologists receive little training in bioinformatics...



but most projects require it..

https://doi.org/10.1371/journal.pcbi.1005755

Barriers to teaching bioinformatics



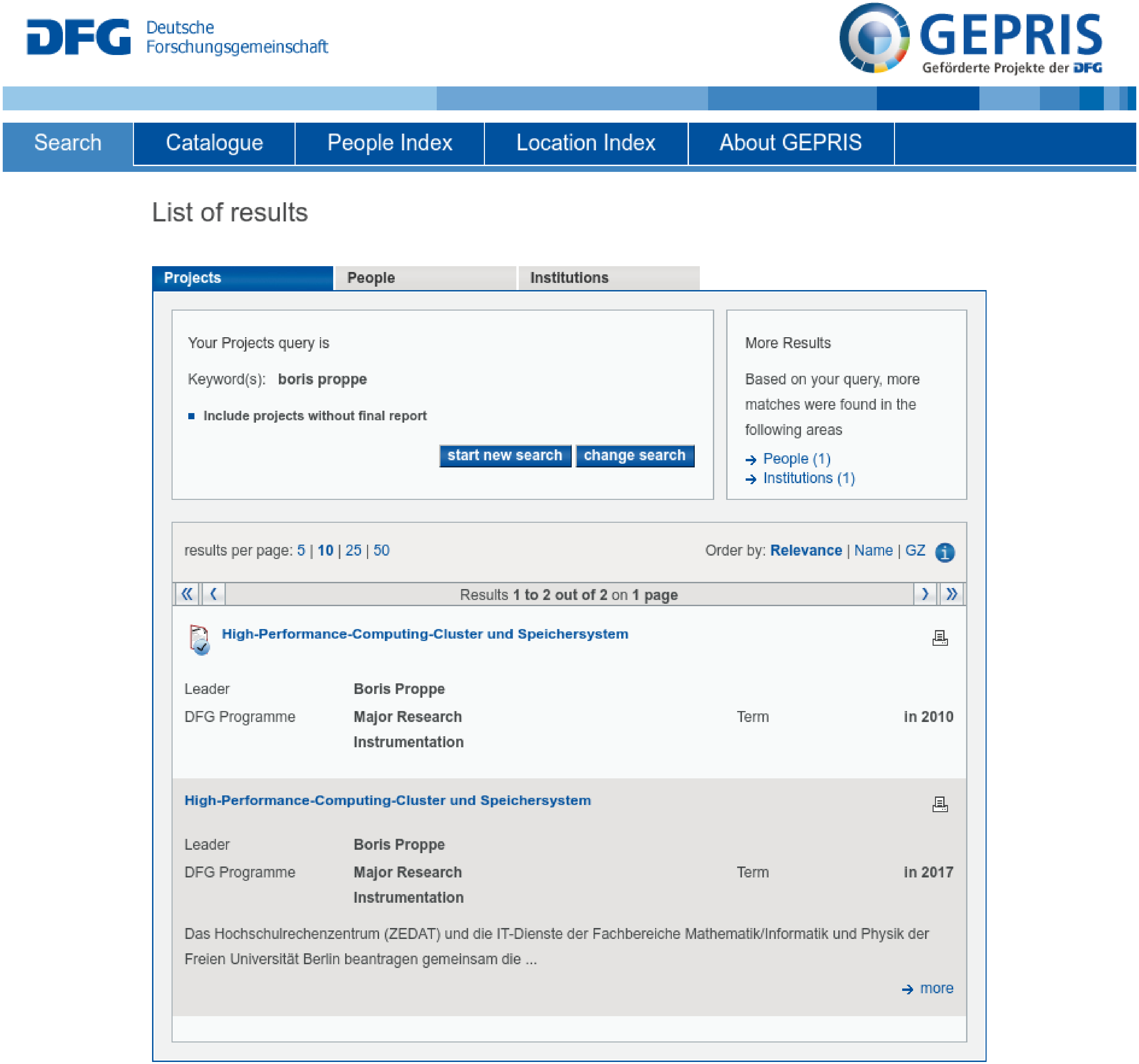
Decade of Highest Degree Earned	Formal Bioinformatics Training (%)	Faculty Integrating Bioinformatics (%)
1980–1989	8.4	35.4
1990–1999	11.3	41.9
2000–2009	35.1	41.7
2010–2016	48.3	25.2

https://doi.org/10.1371/journal.pone.0224288

“These studies suggest a scenario of big data inundating unprepared biologists.”

https://doi.org/10.1371/journal.pcbi.1005755

Computing cluster (HPC)
100,000s-millions €



List of results

Your Projects query is

Keyword(s): **boris proppe**

- **Include projects without final report**

More Results

Based on your query, more matches were found in the following areas

→ People (1)
→ Institutions (1)

results per page: 5 | **10** | 25 | 50

Order by: **Relevance** | Name | GZ

Results **1 to 2 out of 2** on **1 page**

High-Performance-Computing-Cluster und Speichersystem

Leader	**Boris Proppe**		
DFG Programme	**Major Research Instrumentation**	Term	**in 2010**

High-Performance-Computing-Cluster und Speichersystem

Leader	**Boris Proppe**		
DFG Programme	**Major Research Instrumentation**	Term	**in 2017**

Das Hochschulrechenzentrum (ZEDAT) und die IT-Dienste der Fachbereiche Mathematik/Informatik und Physik der Freien Universität Berlin beantragen gemeinsam die ...

→ more

Important points

Every university has (access to) a cluster

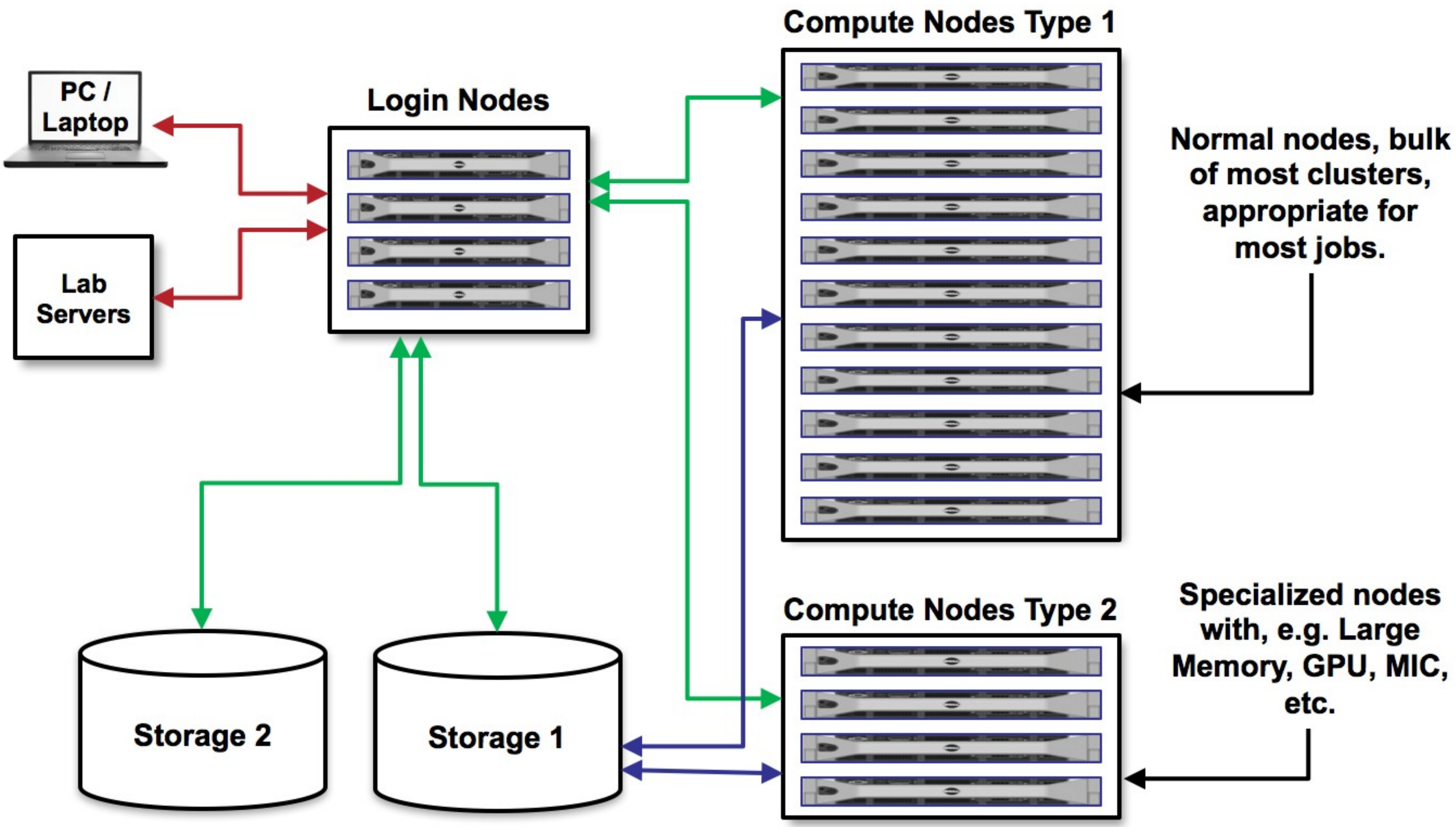
Usage is generally free or cheap

There are university employees to provide support

The system is highly robust and stable

i.e. you have access to a performant computer

Typical cluster


Compute Nodes Type 1
PC / Laptop
Login Nodes
Normal nodes, bulk of most clusters, appropriate for most jobs.
Lab Servers
Compute Nodes Type 2
Specialized nodes with, e.g. Large Memory, GPU, MIC, etc.
Storage 2
Storage 1

Terminology

Job: reservation to run commands

Node: physical machine, part of cluster

Core/CPU: processing unit, nodes contain many CPUs

Partition: nodes may be organized into partition

- e.g. high-memory nodes for big jobs
- e.g. Herbert created gd5302 partition for us

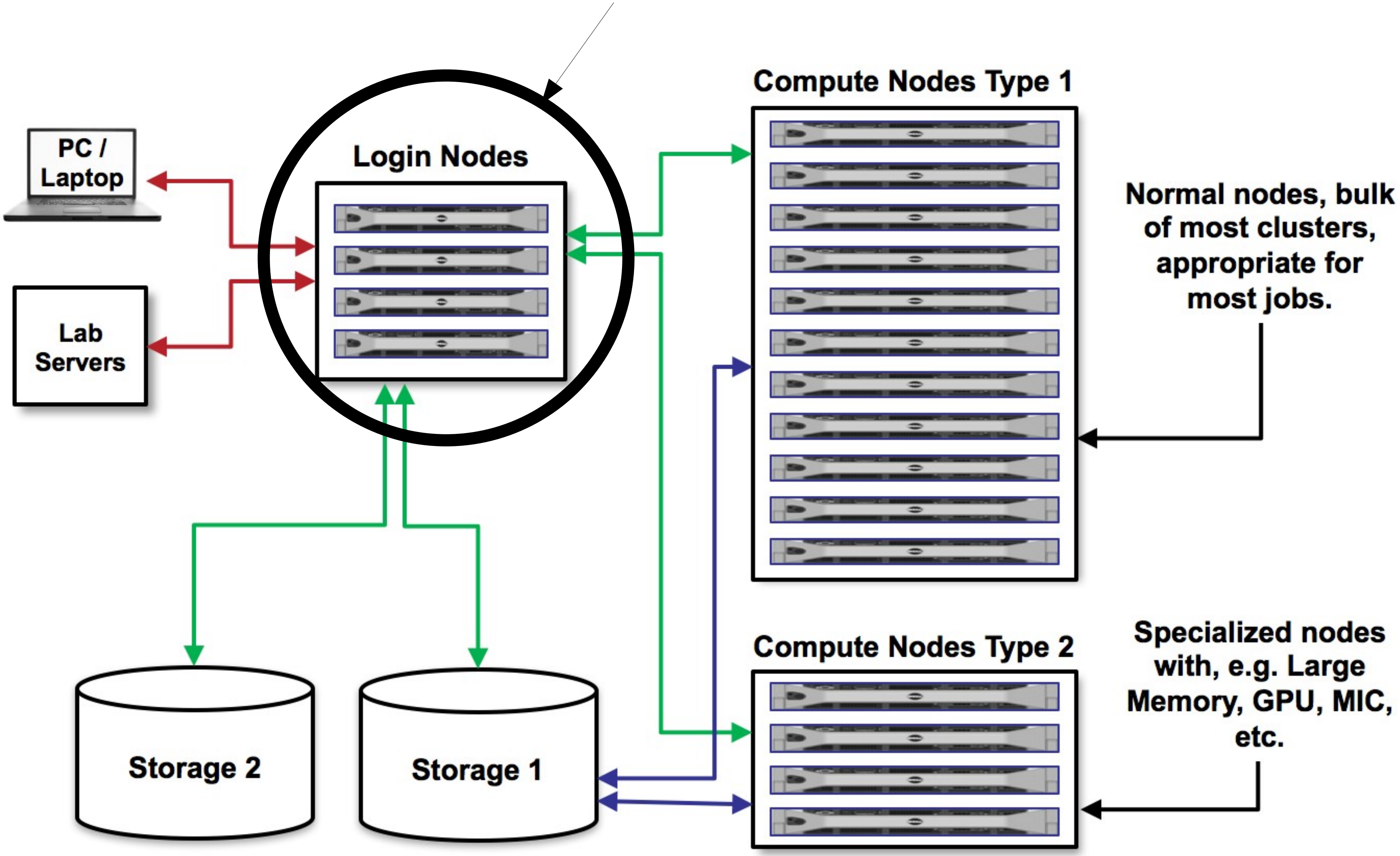
Connecting to the cluster

Go ahead and connect to the HPC:

```
# remember to use your actual username
ssh USERNAME@kennedy.st-andrews.ac.uk
```

(or use PuTTY on windows)

You are now on one of these nodes


PC / Laptop
Lab Servers
Login Nodes
Compute Nodes Type 1
Normal nodes, bulk of most clusters, appropriate for most jobs.
Compute Nodes Type 2
Specialized nodes with, e.g. Large Memory, GPU, MIC, etc.
Storage 2
Storage 1

Login nodes

These are for basic tasks:

- Transferring data	# scp, rsync, wget, etc
- Managing files	# cp, mv, gunzip
- Compiling software	# configure, make
- Editing scripts	# nano, vim
- Checking/managing jobs	# squeue

Important note

The login nodes are for setting things up and submitting jobs to the compute nodes

Running commands on the login nodes is bad

- It slows/crashes the node for other users
- You become unpopular with admin/other users

Compute nodes

Job scripts are sent here to run

- resources allocated by workload manager (Slurm)

- other workload managers/schedulers exist
  e.g. PBS, SGE, LFS

- resources are allocated according to:
  system load
  fair share algorithms

Basic conventions

/home/$USER/

- limited space
- backed up regularly
- store software, config files

/scratch/bioinf/gd5302/$USER/

- lots of storage space
- no backup
- default working space

Different cultures exist in different clusters

e.g. precise location of scratch, backup policy

Workshop materials

we have installed software under:

```
/gpfs1/scratch/bioinf/BL4273/miniforge3/envs/gd5302/bin/
```

the dataset is under:

```
/scratch/bioinf/gd5302/$USER/data/p1/01_dataset/
```

scripts for QC steps are under:

```
/scratch/bioinf/gd5302/$USER/data/p1/02_data_qc/
```

scripts for PCA:

```
/scratch/bioinf/gd5302/$USER/data/p1/03_pca/
```

(bash will replace $USER with your actual username)

Based on GWAS tutorial by Yunye He, Univeristy of Tokyo
https://github.com/Cloufield/GWASTutorial # detailed explanation
https://cloufield.github.io/GWASTutorial/ # code

Dataset

504 east asian individuals

genotyped by 1000 genomes project phase 3
- whole exome sequencing (WES)
- low-pass whole genome sequencing (WGS)

1,235,116 variants
4 files:

```
1KG.EAS.auto.snp.norm.nodup.split.rare002.common015.missing.bed
1KG.EAS.auto.snp.norm.nodup.split.rare002.common015.missing.bim
1KG.EAS.auto.snp.norm.nodup.split.rare002.common015.missing.fam
integrated_call_samples_v3.20130502.ALL.panel
```

Dataset

Lets look at the .bim input file:

```
# go to the data directory
cd /scratch/bioinf/gd5302/$USER/data/p1/01_dataset/
# display the first 10 lines
head 1KG.EAS.auto.snp.norm.nodup.split.rare002.common015.missing.bim
```

```
1     1:14930:A:G       0     14930       G     A
1     1:15774:G:A       0     15774       A     G
1     1:15777:A:G       0     15777       G     A
1     1:57292:C:T       0     57292       T     C
1     1:77874:G:A       0     77874       A     G
1     1:87360:C:T       0     87360       T     C
1     1:92917:T:A       0     92917       A     T
1     1:104186:T:C            0     104186      C     T
1     1:125271:C:T            0     125271      T     C
1     1:232449:G:A            0     232449      A     G
```

Full explanation of file formats:

https://www.cog-genomics.org/plink/1.9/formats

Dataset

Do the same for the .fam file:

```
# go to the data directory
cd /scratch/bioinf/gd5302/$USER/data/p1/01_dataset/
# display the first 10 lines
head 1KG.EAS.auto.snp.norm.nodup.split.rare002.common015.missing.fam
```

```
HG00403    HG00403    0    0    0    -9
HG00404    HG00404    0    0    0    -9
HG00406    HG00406    0    0    0    -9
HG00407    HG00407    0    0    0    -9
HG00409    HG00409    0    0    0    -9
```

1. Family ID ('FID')
2. Within-family ID ('IID'; cannot be '0')
3. Within-family ID of father ('0' if father isn't in dataset)
4. Within-family ID of mother ('0' if mother isn't in dataset)
5. Sex code ('1' = male, '2' = female, '0' = unknown)
6. Phenotype value ('1' = control, '2' = case, '-9'/'0'/non-numeric = missing data if case/control)

Dataset

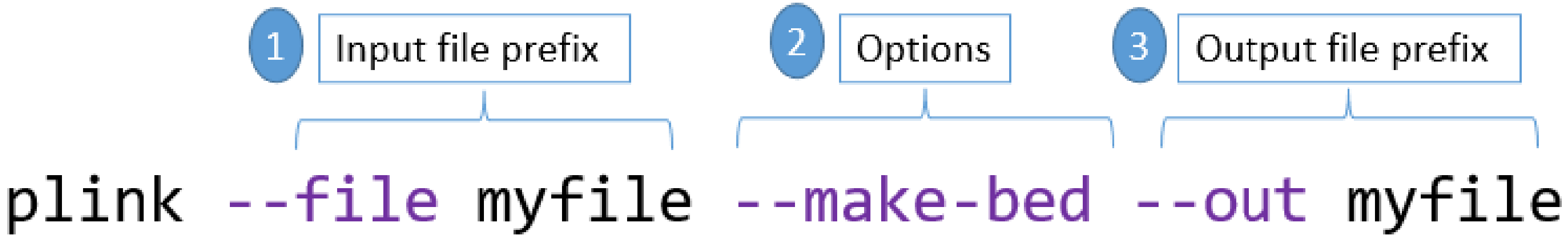
And for the population labels:

```
# go to the data directory
cd /scratch/bioinf/gd5302/$USER/data/p1/01_dataset/
# display the first 5 lines
head -n 5 integrated_call_samples_v3.20130502.ALL.panel
```

```
sample  pop super_pop   gender
HG00096     GBR     EUR     male
HG00097     GBR     EUR     female
HG00099     GBR     EUR     female
HG00100     GBR     EUR     female
```

We will use these labels later to color our PCA plot

Example commands



1 Input file prefix

2 Options

3 Output file prefix

```
plink --file myfile --make-bed --out myfile
```

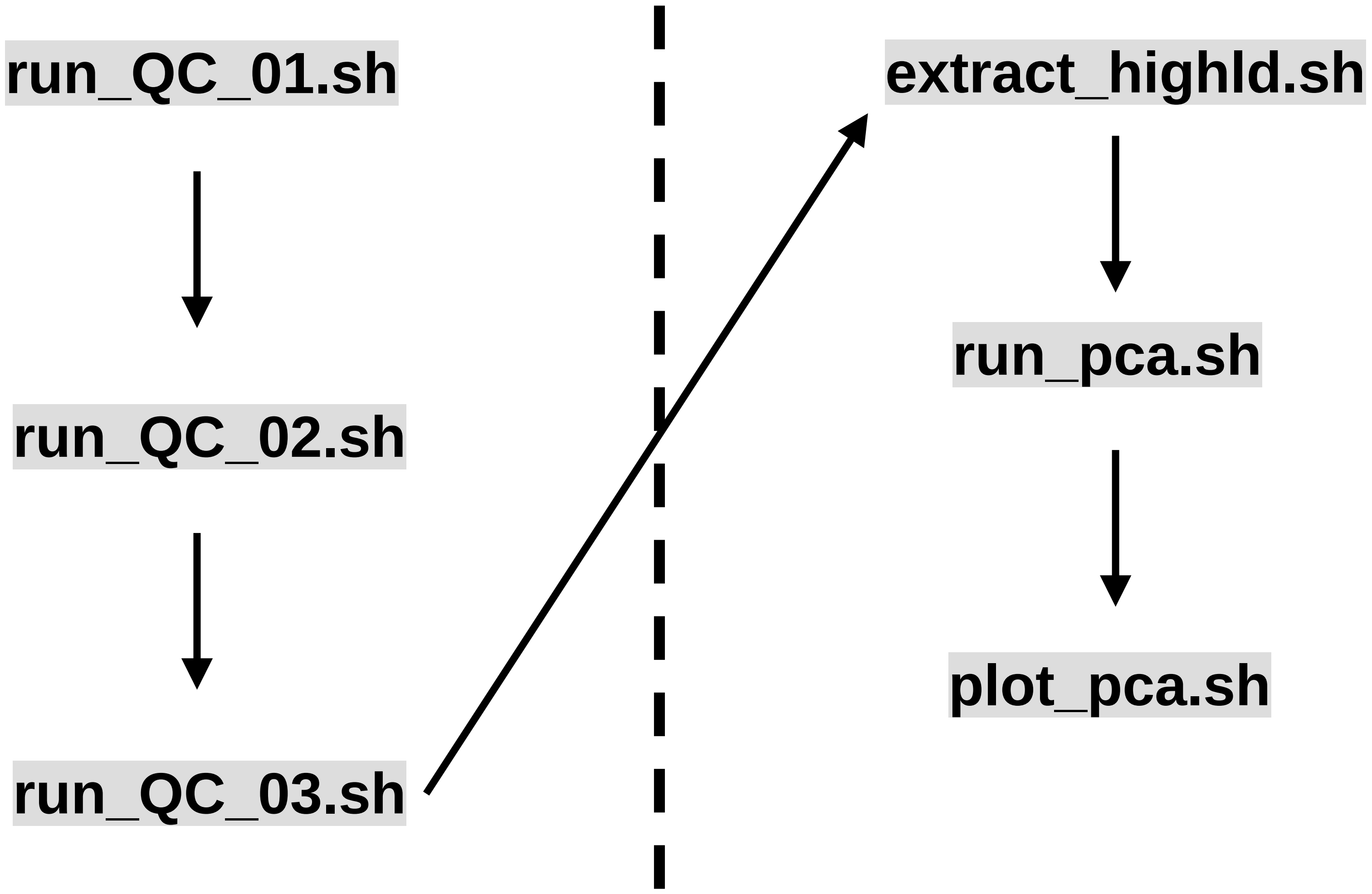
```
# for example
plink \
    --bfile ${genotypeFile} \
    --hardy \
    --out plink_results
```

- the above command produces `plink_results.hwe`

QC

PCA

- we will run each script in order
- we will wait for each to finish before submitting the next



run_QC_01.sh

run_QC_02.sh

run_QC_03.sh

extract_highld.sh

run_pca.sh

plot_pca.sh

First QC script run_QC_01.sh

```
#! /bin/bash
#SBATCH --chdir=/scratch/bioinf/gd5302/prj3/data/p1/02_data_qc/   # directory
#SBATCH --job-name=qc01                                         # job name
#SBATCH --ntasks=1                                              # no. of tasks in job
#SBATCH --cpus-per-task=1                                       # requested CPUs
#SBATCH --nodes=1                                               # requested nodes
#SBATCH --mem=4G                                                # requested RAM
#SBATCH –partition=singlenode,gd5302                            # specify partition
#SBATCH –output=slurm-out-qc01.txt                             # log file

export PATH=/gpfs1/scratch/bioinf/BL4273/miniforge3/envs/gd5302/bin/:$PATH

plink \
        --bfile ${genotypeFile} \
        --missing \
        --freq \
        --hardy \
        --out plink_results
```

First QC script run_QC_01.sh

Go to the directory containing the first QC script:

```
cd /scratch/bioinf/gd5302/$USER/data/p1/02_data_qc/
```

Submit the first QC script for execution:

```
sbatch run_QC_01.sh
```

- sbatch is a command from the slurm workload/schedule managing software

- squeue is used to monitor the queue (or squeue --me)

```
$ squeue --me
             JOBID PARTITION     NAME     USER ST       TIME  NODES
NODELIST(REASON)
            234920 singlenod      qc01     prj3 PD       0:00      1 (Resources)
```

First QC script run_QC_01.sh

What did the script do?

```
plink \
        --bfile ${genotypeFile} \
        --missing \
        --freq \
        --hardy \
        --out plink_results
```

We created five summary files:

plink_results.lmiss	**(variant-based missing data report)**
plink_results.imiss	**(sample-based missing data report)**
plink_results.frq	**(basic allele frequency report)**
plink_results.hwe	**(Hardy-Weinberg equilibrium exact test statistic report)**

We will use these to filter the data in later steps

Missing data reports

.lmiss gives the missing rate for each SNP

```
head -n4 plink_results.lmiss
```

```
 CHR           SNP     N_MISS   N_GENO      F_MISS
   1   1:14930:A:G          2      504    0.003968
   1   1:15774:G:A          3      504    0.005952
   1   1:15777:A:G          3      504    0.005952
```

.imiss gives the missing rate for each sample

```
head -n4 plink_results.imiss
```

```
      FID       IID  MISS_PHENO   N_MISS   N_GENO   F_MISS
  HG00403   HG00403          Y     10020    1235116 0.008113
  HG00404   HG00404          Y     9192     1235116 0.007442
  HG00406   HG00406          Y     15751    1235116  0.01275
```

https://www.cog-genomics.org/plink/1.9/formats#lmiss

https://www.cog-genomics.org/plink/1.9/formats#imiss

Frequency reports

.frq reports Minor Allele Frequency (MAF)

```
head -4 plink_results.frq
```

```
 CHR           SNP   A1   A2          MAF  NCHROBS
   1   1:14930:A:G    G    A       0.4133     1004
   1   1:15774:G:A    A    G      0.02794     1002
   1   1:15777:A:G    G    A      0.07385     1002
```

Can be used to categorize variants:

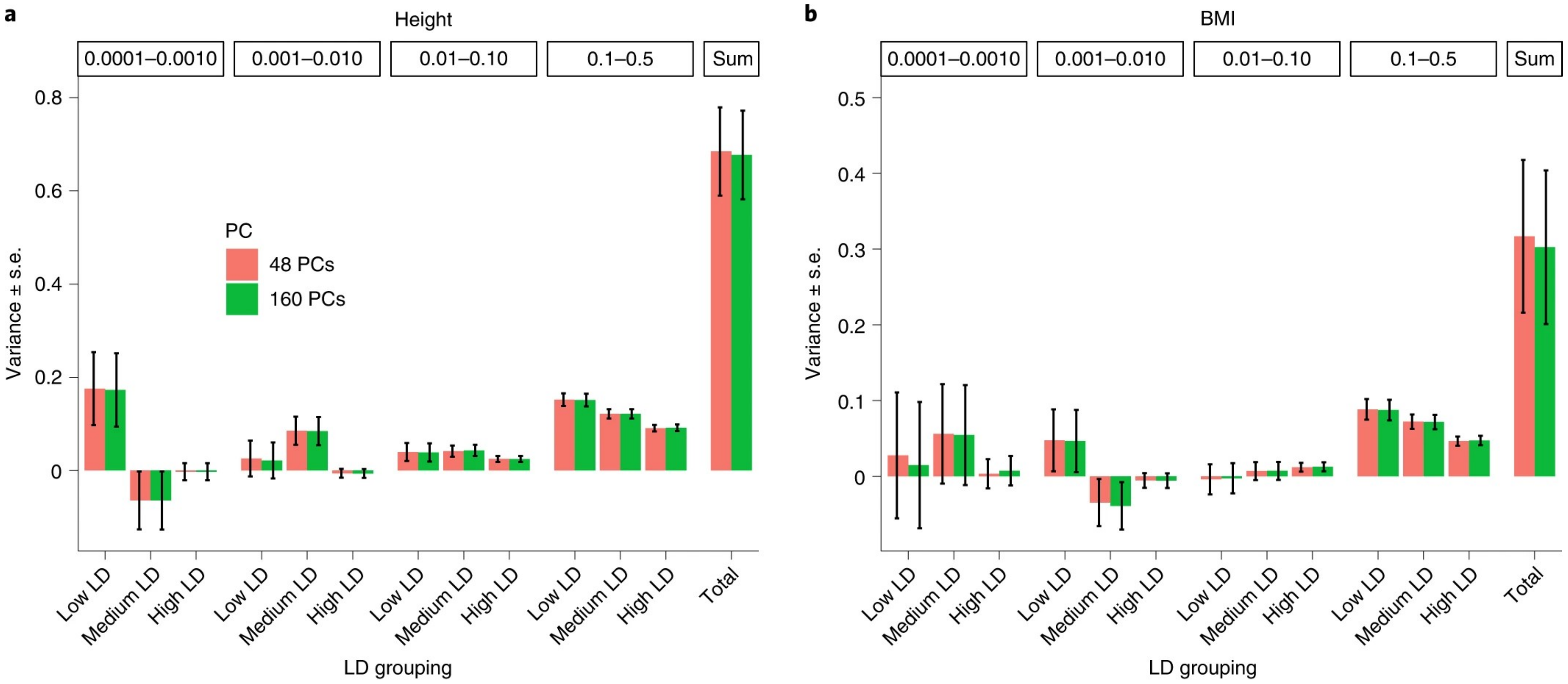
common variants, MAF>=0.05

low-frequency variants : 0.01<=MAF<0.05

rare variants : MAF<0.01

https://www.cog-genomics.org/plink/1.9/basic_stats#freq

Frequency reports



e.g. may be useful to know the frequency of GWAS hits

Wainschtein, P., Jain, D., Zheng, Z. et al. Assessing the contribution of rare variants to complex trait heritability from whole-genome sequence data. Nat Genet 54, 263–273 (2022). https://doi.org/10.1038/s41588-021-00997-7

Hardy-Weinberg equilibrium

.hwe reports Hardy-Weinberg equilibrium tests

```
head -4 plink_results.hwe
```

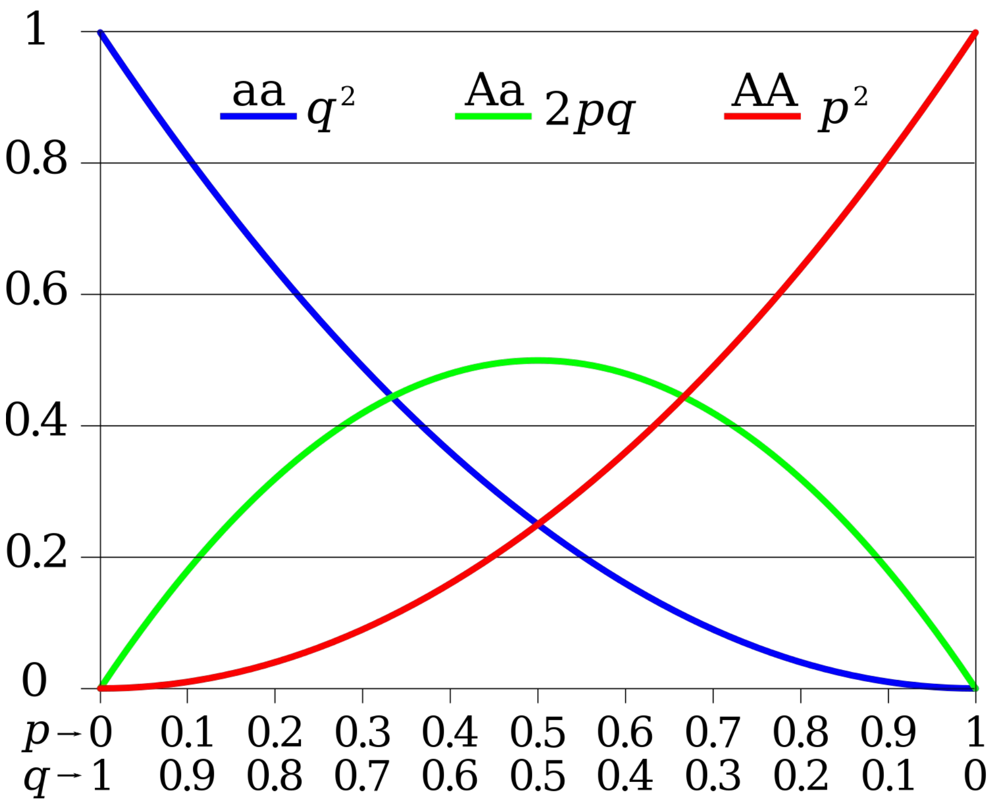
```
 CHR           SNP    TEST   A1   A2              GENO   O(HET)   E(HET)
  P
   1     1:14930:A:G  ALL(NP)    G    A          4/407/91   0.8108    0.485
4.864e-61
   1     1:15774:G:A  ALL(NP)    A    G          0/28/473  0.05589  0.05433
   1
   1     1:15777:A:G  ALL(NP)    G    A          1/72/428   0.1437   0.1368
0.5053
```

Natural selection, genetic drift, and gene flow all cause deviation from Hardy-Weinberg equilibrium

But we are only using it to detect technical problems

https://www.cog-genomics.org/plink/1.9/filter#hwe

Hardy-Weinberg equilibrium



technical problems such as allelic dropout can cause deviation

i.e. we are simply calculating HWE to use it as a QC filter

Example log file

Each script will generate a log file

```
# display the contents of the first log file
cat slurm-out-qc01.txt
```

```
1235116 variants loaded from .bim file.
504 people (0 males, 0 females, 504 ambiguous) loaded from .fam.
Ambiguous sex IDs written to plink_results.nosex .
Using 1 thread (no multithreaded calculations invoked).
Before main variant filters, 504 founders and 0 nonfounders present.
Calculating allele frequencies... done.
Total genotyping rate is 0.993828.
--freq: Allele frequencies (founders only) written to plink_results.frq .
--missing: Sample missing data report written to plink_results.imiss,
and
variant-based missing data report written to plink_results.lmiss.
--hardy: Writing Hardy-Weinberg report (founders only) to
plink_results.hwe ...
done.
1235116 variants and 504 people pass filters and QC.
```

Second QC script run_QC_02.sh

Submit the second QC script for execution:

```
sbatch run_QC_02.sh
```

Second QC script run_QC_02.sh

Let’s look at the second QC script:
- the goal is to identify samples with high/low inbreeding
- we will then filter them later

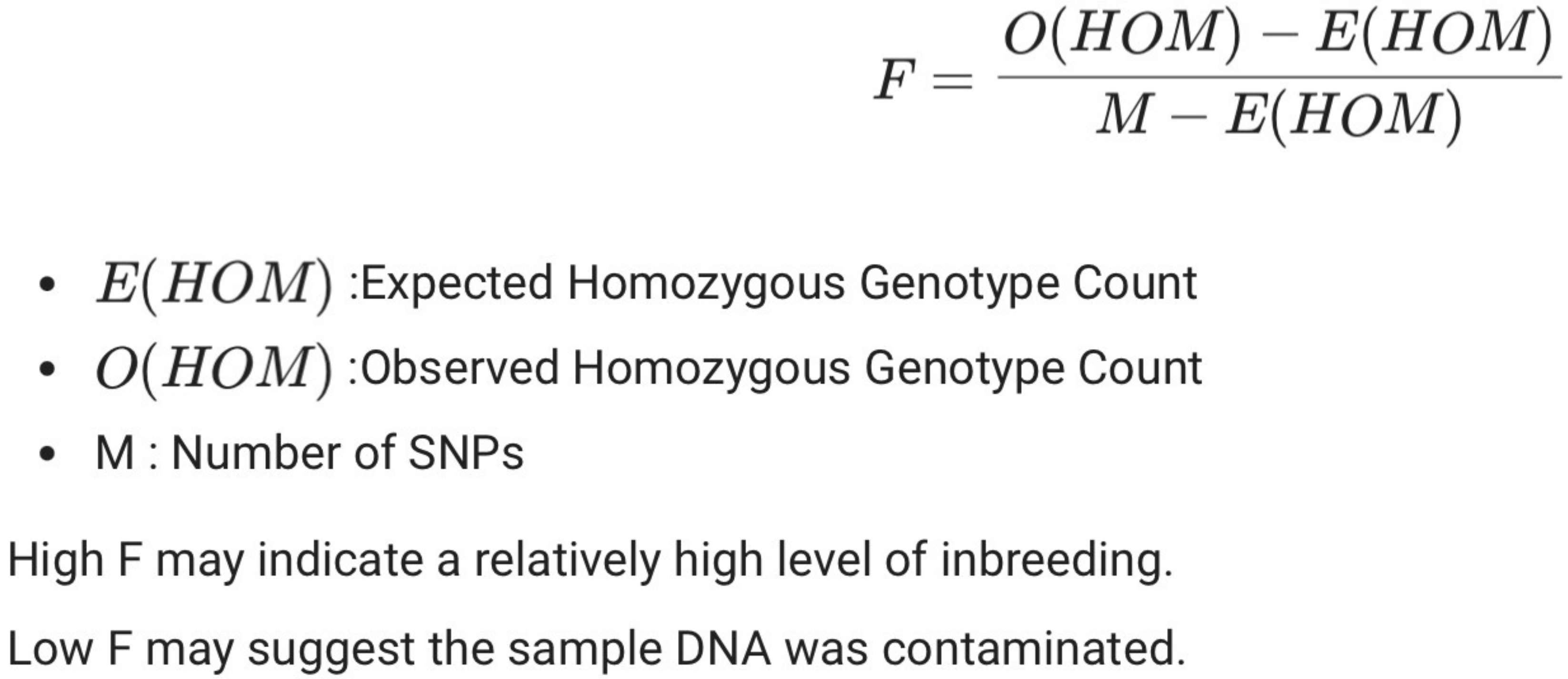
```
plink \
        --bfile ${genotypeFile} \
        --maf 0.01 \
        --geno 0.02 \
        --mind 0.02 \
        --hwe 1e-6 \
        --indep-pairwise 50 5 0.2 \
        --out plink_results

plink \
        --bfile ${genotypeFile} \
        --extract plink_results.prune.in \
        --het \
        --out plink_results

awk 'NR>1 && $6>0.1 || $6<-0.1 {print $1,$2}' plink_results.het > high_het.sample
```

Second QC script run_QC_02.sh

The goal is to calculate the inbreeding coefficient:



$$F = \frac{O(HOM) - E(HOM)}{M - E(HOM)}$$

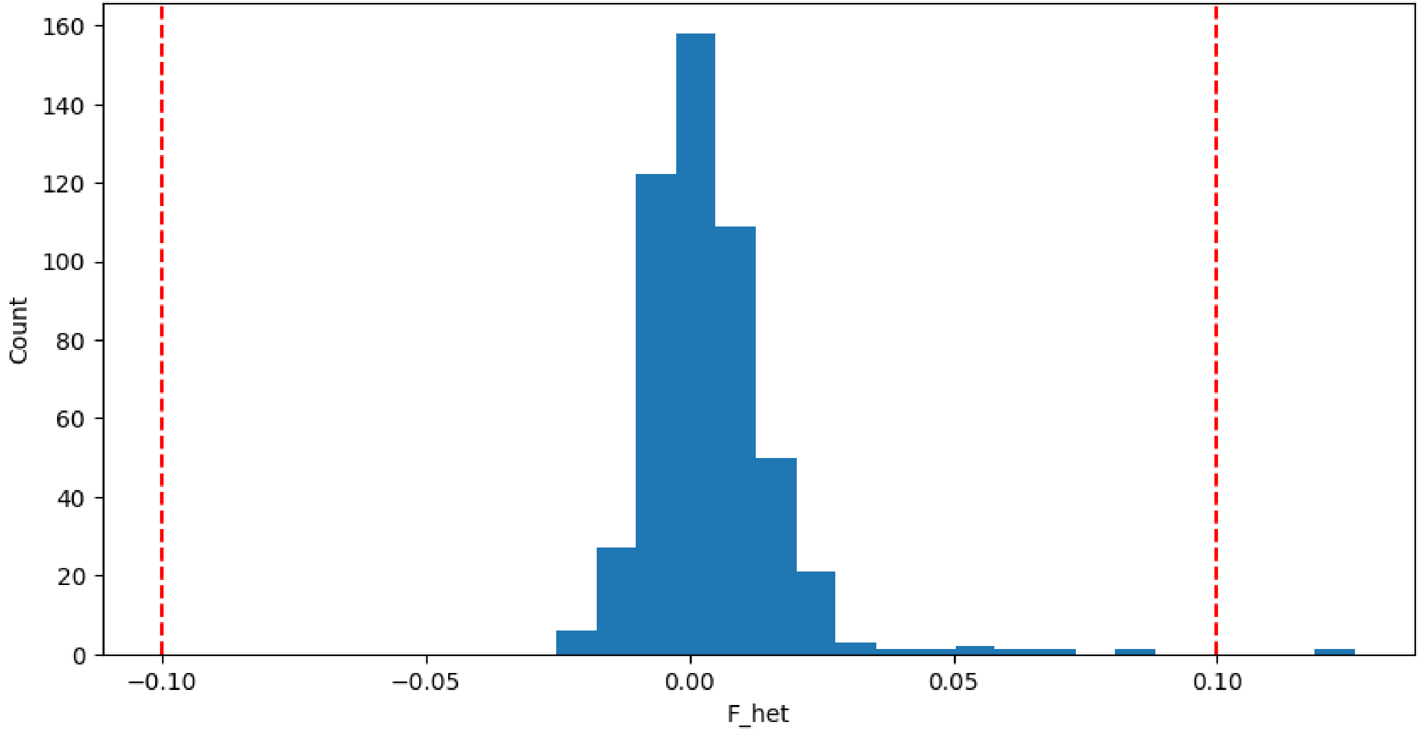
- $E(HOM)$:Expected Homozygous Genotype Count
- $O(HOM)$:Observed Homozygous Genotype Count
- M : Number of SNPs

High F may indicate a relatively high level of inbreeding.

Low F may suggest the sample DNA was contaminated.

Second QC script run_QC_02.sh

The goal is to calculate the inbreeding coefficient:



- true inbreeding (high values) could generate false associations
- low values could indicate genotyping errors/contamination

Second QC script run_QC_02.sh

Before we calculate inbreeding, we first filter on LD
(so that LD doesn’t affect inbreeding calculation)

```
plink \
        --bfile ${genotypeFile} \
        --maf 0.01 \
        --geno 0.02 \
        --mind 0.02 \
        --hwe 1e-6 \
        --indep-pairwise 50 5 0.2 \
        --out plink_results

plink \
        --bfile ${genotypeFile} \
        --extract plink_results.prune.in \
        --het \
        --out plink_results

awk 'NR>1 && $6>0.1 || $6<-0.1 {print $1,$2}' plink_results.het > high_het.sample
```

Second QC script run_QC_02.sh

Then we calculate inbreeding (heterozygosity)

```
plink \
        --bfile ${genotypeFile} \
        --maf 0.01 \
        --geno 0.02 \
        --mind 0.02 \
        --hwe 1e-6 \
        --indep-pairwise 50 5 0.2 \
        --out plink_results

plink \
        --bfile ${genotypeFile} \
        --extract plink_results.prune.in \
        --het \
        --out plink_results

awk 'NR>1 && $6>0.1 || $6<-0.1 {print $1,$2}' plink_results.het > high_het.sample
```

Second QC script run_QC_02.sh

Then we make a list of samples with inbreeding coef.
 - below -0.1
 - or above 0.1

```
plink \
        --bfile ${genotypeFile} \
        --maf 0.01 \
        --geno 0.02 \
        --mind 0.02 \
        --hwe 1e-6 \
        --indep-pairwise 50 5 0.2 \
        --out plink_results

plink \
        --bfile ${genotypeFile} \
        --extract plink_results.prune.in \
        --het \
        --out plink_results

awk 'NR>1 && $6>0.1 || $6<-0.1 {print $1,$2}' plink_results.het > high_het.sample
```

Third QC script run_QC_03.sh

Submit the third QC script for execution:

```
sbatch run_QC_03.sh
```

Third QC script run_QC_03.sh

Here we use the various statistics from steps 1&2 to filter

```
plink \
        --bfile ${genotypeFile} \
        --maf 0.01 \                            # Minor allele frequency
        --geno 0.02 \                           # variant mssing rate
        --mind 0.02 \                           # sample missing rate
        --hwe 1e-6 \                            # hardy-weinberg eq.
        --remove high_het.sample \              # inbreeding coefficient
        --keep-allele-order \
        --make-bed \                            # make binary file for future work
        --out sample_data.clean                 # prefix for filtered files
```

This will create the following filtered files for use next week:

```
sample_data.clean.bed
sample_data.clean.bim
sample_data.clean.fam
```

Third QC script run_QC_03.sh

The log file contains key stats on the remaining samples/SNPs:

```
...
1235116 variants loaded from .bim file.
504 people (0 males, 0 females, 504 ambiguous) loaded from .fam.
...
--remove: 503 people remaining.
3 people removed due to missing genotype data (--mind).
Total genotyping rate in remaining samples is 0.993936.
375 variants removed due to missing genotype data (--geno).
--hwe: 10637 variants removed due to Hardy-Weinberg exact test.
95372 variants removed due to minor allele threshold(s)
(--maf/--max-maf/--mac/--max-mac).
1128732 variants and 500 people pass filters and QC.
...
```

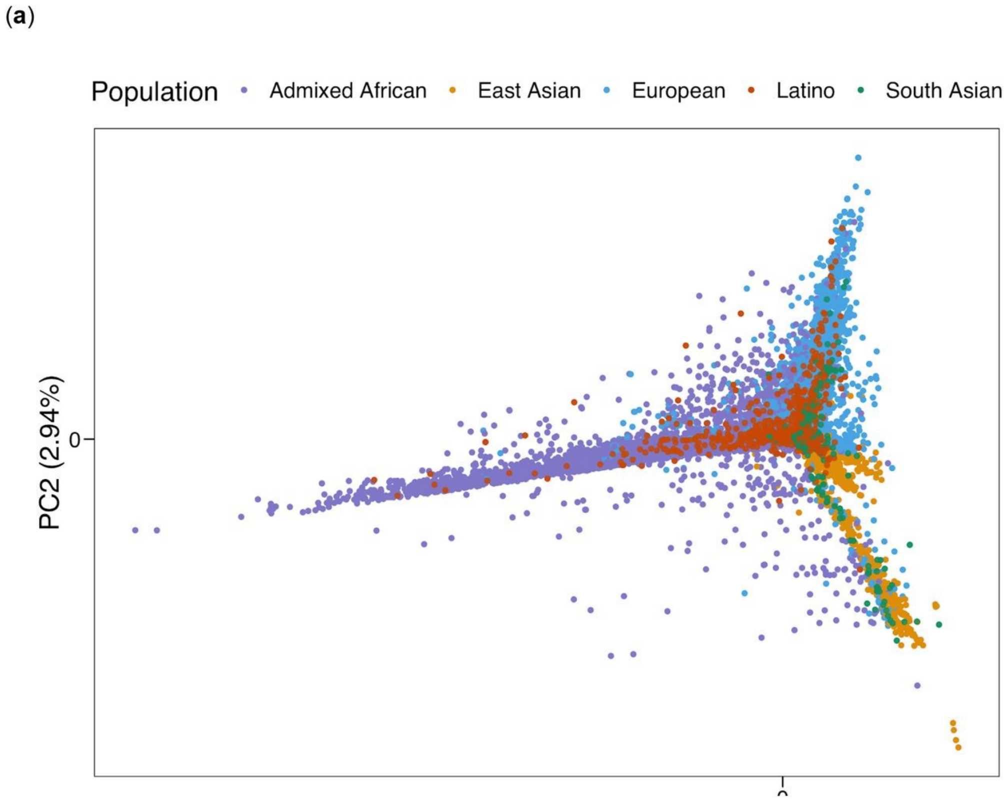
First PCA script extract_highld.sh

Submit the first PCA script for execution:

```
sbatch extract_highld.sh
```

First PCA script extract_highld.sh

Some regions of the genome are under strong LD



e.g.
Human Leukocyte Antigens
(HLA, immune genes)

- we want our PCA to show ancestry
- we don’t want it to show who had plague 1000 years ago
- so we want to remove these regions in case they distort PCA

Second PCA script run_pca.sh

Submit the second PCA script for execution:
(make sure the previous script has finished first)

```
sbatch run_pca.sh
```

Third PCA script plot_pca.sh

Submit the third PCA script for execution:

```
sbatch plot_pca.sh
```

Third PCA script plot_pca.sh

This is potentially confusing:

- we are submitting a bash script that executes a python script
- we specify a version of python that we installed

```
#! /bin/bash
#SBATCH --chdir=/scratch/bioinf/gd5302/prj3/data/p1/03_pca/
#SBATCH --job-name=run_pca
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=1
#SBATCH --nodes=1
#SBATCH --partition=singlenode,gd5302
#SBATCH --time=00:15:00
#SBATCH --mem=4G
#SBATCH --output=slurm-out-plot_pca.txt

export PATH=/gpfs1/scratch/bioinf/BL4273/miniforge3/envs/gd5302/bin/:$PATH

/gpfs1/scratch/bioinf/BL4273/miniforge3/envs/gd5302/bin/python3.11 plot_pca.py
```

Plot PCA script plot_pca.py

The python script is just a simple scatter plot of the PCA data

```
#! /gpfs1/scratch/bioinf/BL4273/miniforge3/envs/gd5302/bin/python3.11

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Read PCA data
pca =
pd.read_table("/scratch/bioinf/gd5302/prj3/data/p1/03_pca/plink_results2_projected.s
score", sep="\t")

# Read ped data
ped =
pd.read_table("/scratch/bioinf/gd5302/prj3/data/p1/01_dataset/integrated_call_sampl
es_v3.20130502.ALL.panel", sep="\t")

# Merge PCA and ped data
pcaped = pd.merge(pca, ped, right_on="sample", left_on="IID", how="inner")

# Plot PCA components
plt.figure(figsize=(10, 10))
```

Transfer PCA plot from HPC (macOS)

- we have generated a PCA plot
- we need to transfer the file from the cluster to view it

```
# remember to use your own username
# open a new terminal on your computer
scp USERNAME@host:/path/to/file path/to/destination/
# example on kennedy with my own username
# . is shorthand for the current directory
scp prj3@kennedy:/scratch/bioinf/gd5302/prj3/data/p1/03_pca/pca_plot.pdf .
#  change prj3 to your own username
# you could give any path you want instead of .
```

- reverse the order to copy *to* the remote host

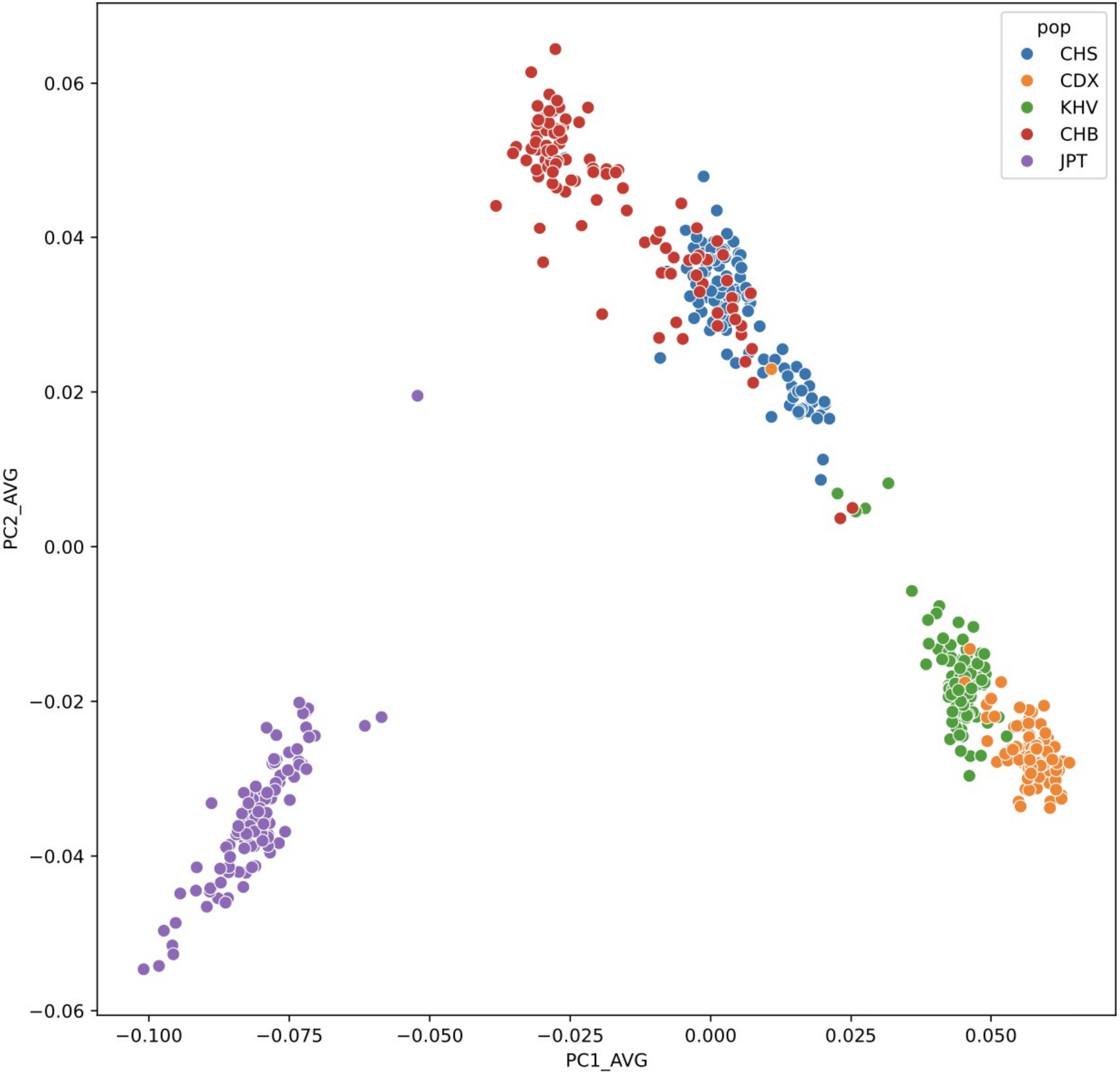
(but we don’t need to copy anything *to* the HPC today)

```
# remember to use your own username
scp path/to/local/file USERNAME@host:/path/to/destination/
```

Transfer PCA plot from HPC (windows)

- launch pscp.exe (installed with PuTTY)

```
# remember to use your own username instead of USER
pscp USER@kennedy:/scratch/bioinf/gd5302/USER/data/p1/03_pca/pca_plot.pdf%USERPROFILE%\Documents\pca_plot.pdf
```


pop
CHS
CDX
KHV
CHB
JPT
PC2_AVG
0.06
0.04
0.02
0.00
−0.02
−0.04
−0.06
PC1_AVG
−0.100
−0.075
−0.050
−0.025
0.000
0.025
0.050