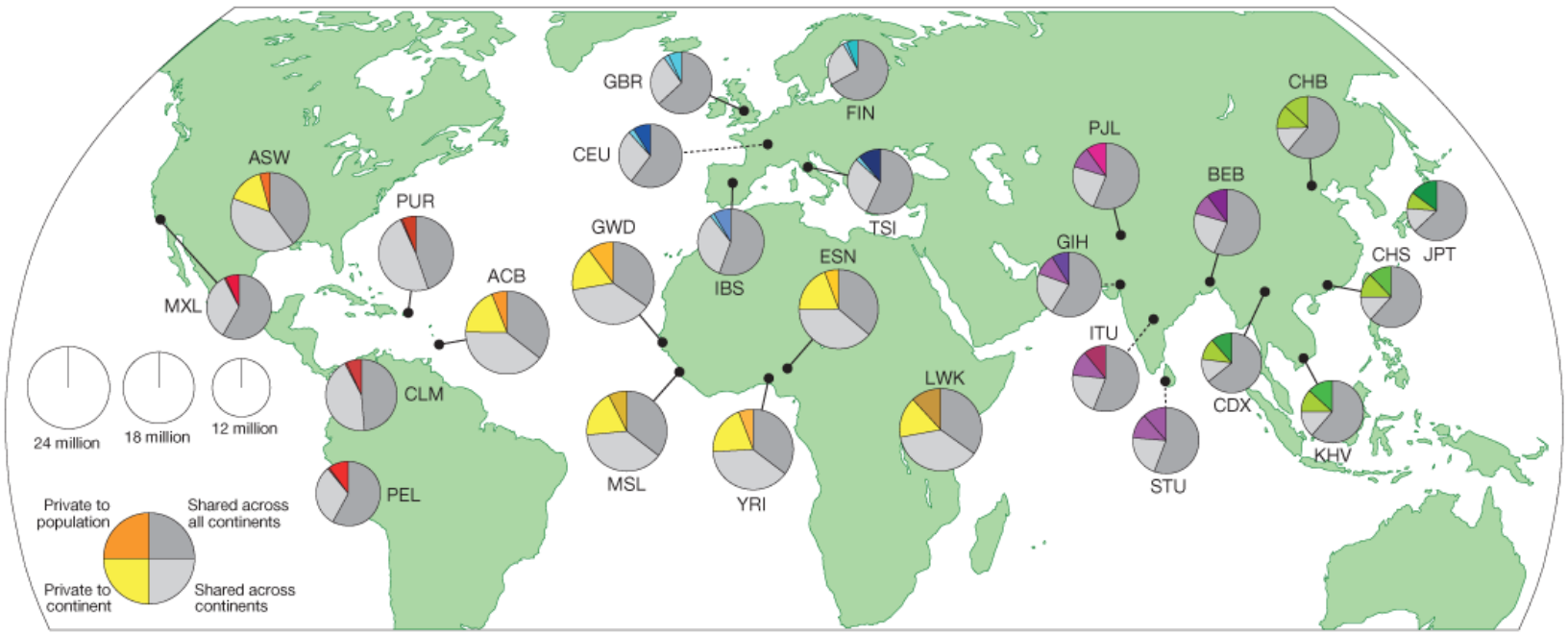
Analysis of Genetic Data 1: Inferring Population Structure

Peter Carbonetto

Research Computing Center and the Dept. of Human Genetics
University of Chicago



Workshop aims

1. Work through the steps of a basic population structure analysis in human genetics, starting with the **"raw" source data**, and ending with a **visualization of population structure** estimated from the genetic data.
2. Get acquainted with standard file formats for storing large genetic data sets.
3. Use command-line tools to inspect and manipulate genetic data.
4. Get experience running a genetic data analysis on the RCC cluster.
5. Follow Best Practices for data analysis on a HPC cluster.

Workshop aims

- This is a *hands-on workshop*—you will get the most out of this workshop if you work through the examples on your computer.
- All the examples are intended to run on the RCC cluster.
- You may try to run the examples on your laptop. *However, I cannot guarantee all examples will work the same on your laptop. My instructions will assume you are using midway3.*

Software tools we will use today

1. PLINK
2. R
3. Several R packages such as ggplot2.
4. Some basic shell commands.

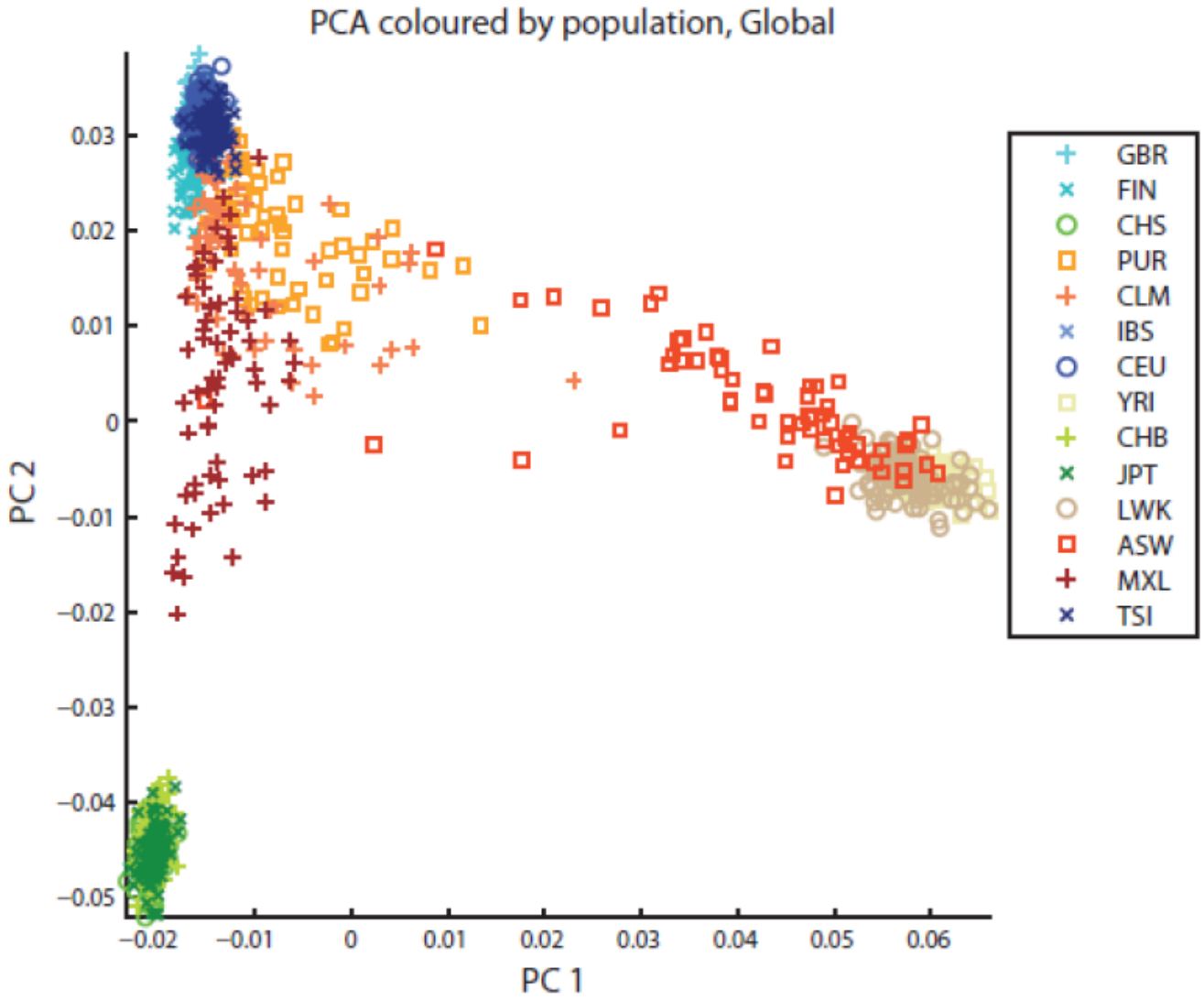
Our research task

We will simulate a population structure analysis commonly done in human genetics studies.

1. We have collected genotype samples as part of our study.
2. We would like to uncover population structure in this sample.
3. We will use the most commonly used statistical technique—principal components analysis (PCA)—to investigate population structure from the genetic data.

We will use the data from the "1000 Genomes" study (Abecasis et al, *Nature*, 2012), which is publicly available on the Web.

Our research task



Outline of workshop

- **Preliminaries**
- Programming challenges:
  1. Set up your environment for genetic data analysis.
  2. Prepare genotype data for PCA.
  3. Run PCA on genotype data.
  4. Visualize and interpret PCA results.

Preliminaries

- WiFi.
- Power outlets.
- Reading what I type.
- Pace, questions, and what to do if you get stuck.

Preliminaries

- The workshop packet is a repository on GitHub. Go to:
  - ▷ github.com/rcc-uchicago/genetic-data-analysis-1
- Download the workshop packet to your computer.

What's included in the workshop packet

- **slides.pdf:** These slides.
- **slides.Rmd:** R Markdown source used to create these slides.
- **pca.R**, **pca.sbatch**: Example R and Slurm script implementing the PCA analysis.
- **functions.R**: Some R functions used to run PCA and visualize the PCs.
- **omni_samples.20141118.panel**: 1000 Genomes population labels.
- **1kg.pop**: description of population labels.
- **20140625_related_individuals.txt**: 31 samples identified as "closely related".

Outline of workshop

- Preliminaries
- Programming challenges:
  1. **Set up your environment for genetic data analysis.**
  2. Prepare genotype data for PCA.
  3. Run PCA on genotype data.
  4. Visualize and interpret PCA results.

Challenge #1: Set up your HPC environment

- Aim: Configure your HPC environment for analysis of the genetic data.
- Steps:
  1. Connect to midway3.
  2. Download workshop packet.
  3. Install PLINK (optional).
  4. Download 1000 Genomes data.
  5. Connect to a midway3 compute node.
  6. Launch R, and check your R environment.
  7. Set up R for plotting.

Connect to midway3

- **If you have an RCC account:** I'm assuming you already know how to connect to midway3. *ThinLinc is recommended if you do not know how activate X11 forwarding in SSH.* See: rcc.uchicago.edu/docs/connecting

Download workshop packet

Once you have connected to a midway3 login node, download the workshop packet to your *scratch directory* on the cluster (note there are no spaces in the URL below):

```
cd /scratch/midway3/cnetid
git clone https://github.com/rcc-uchicago/
  genetic-data-analysis-1.git
```

Alternatively, download the packet as a ZIP file (note there are no spaces in the URL):

```
wget https://codeload.github.com/rcc-uchicago/
  genetic-data-analysis-1/zip/refs/heads/master
unzip master
mv genetic-data-analysis-1-master
  genetic-data-analysis-1
```

Install PLINK (optional)

PLINK is already installed on midway3. To use the previously installed PLINK, run:

```
module load plink/1.90b6.21
plink --version
```

Alternatively, if you want to use the latest version of PLINK, you may download and install PLINK in the same place as the other workshop materials.

- URL: www.cog-genomics.org/plink2

Download 1000 Genomes data

Copy or download the 1000 Genomes data from the European Bioinformatics Institute. *Downloading from the EBI may take a long time (10–20 minutes), so it is better to copy the previously downloaded files.*

- Copy previously downloaded data:

```
cd genetic-data-analysis-1
cp ~pcarbo/share/1kg.vcf.gz .
```

- Or download the data from EBI:
  - ▷ FTP: `ftp.1000genomes.ebi.ac.uk`
  - ▷ Directory: `vol1/ftp/release/20130502/supporting/hd_genotype_chip`
  - ▷ Download this file: `ALL.chip.omni_broad_sanger_combined.20140818.snps.genotypes.vcf.gz`
  - ▷ Rename it as `1kg.vcf.gz`.

Connect to a midway3 compute node

Set up an interactive session on a midway3 compute node with 8 CPUs and 19 GB of memory:

```
screen -S workshop
sinteractive -p caslake -c 8 \
--reservation=gda1_workshop \
  --account=rcc-guest \
  --mem=19G --time=3:00:00
echo $HOSTNAME
```

Launch R

Start up an interactive R session:

```
cd genetic-data-analysis-1
module load R/4.2.0
which R
R
```

Check your R environment

Check that you are running R 4.2.0:

```
version
```

Check that you are starting with an empty environment:

```
ls()
```

Check that you have the correct working directory—it should be set to the “genetic-data-analysis-1” repository:

```
getwd()
```

Set up R for plotting

Make sure you can load ggplot2, and create a plot.

```
library(ggplot2)
data(cars)
qplot(cars$dist,cars$speed)
```

You should see a scatterplot.

Quit R

Quit R. We will return to it later.

```
quit()
```

At this point, you have completed the initial setup. You are now ready to move on to the next programming challenge.

Outline of workshop

- Preliminaries
- Programming challenges:
  1. Set up your environment for genetic data analysis.
  2. **Prepare genotype data for PCA.**
  3. Run PCA on genotype data.
  4. Visualize and interpret PCA results.

Challenge #2: Prepare genotype data

- Aim: Prepare genotype data for PCA analysis in R.
- Steps:
  1. Examine VCF file.
  2. Convert VCF file to PLINK format.
  3. Convert PLINK text file to binary format.
  4. Remove related 1000 Genomes samples.
  5. Prune SNPs in LD.

Examine the VCF file

Run a few shell commands to inspect the genotype data stored in the VCF file.

```
cd genetic-data-analysis-1
ls -lh 1kg.vcf.gz
zcat 1kg.vcf.gz | less -S
```

- Reference:
  `www.cog-genomics.org/plink2/formats#vcf`

What is a VCF file?

- The Variant Call Format (VCF) is a text format for storing many types of DNA variant data (e.g., SNPs, deletions, insertions), and for annotating these variants.
- It is one of the most commonly used data formats in genetics.
- It is not the most efficient way to store genotype data.
- See also:
  - `vcftools.github.io`
  - `samtools.github.io/hts-specs`
  - `doi:10.1093/bioinformatics/btr330`

Convert VCF to PLINK

Run this command to convert the genotypes on *chromosome 1 only* from VCF to the PLINK text format. *This may take a few minutes.*

```
plink --vcf 1kg.vcf.gz --recode \
  --chr 1 --allow-extra-chr \
  --out 1kg
```

Explore PLINK files

Run a few simple shell commands to inspect the genotype data stored in the PLINK files.

```
head 1kg.map
tail 1kg.map
wc -l 1kg.map
less -S 1kg.ped
wc -l -w 1kg.ped
```

PLINK files: concepts

- Probably most commonly used format for storing human genoytpe data.
- Less flexible than VCF.
- Easy to view and manipulate with simple shell commands (e.g., wc, grep, cat, cut, paste).
- Use PLINK binary (.bed) format for faster processing and more efficient storage. However, it is not “human readable”.
- See:
  `www.cog-genomics.org/plink/1.9/formats#ped`

Convert to binary (“bed”) format

To speed up the data processing steps, convert to binary PLINK format. *This step may take several minutes to complete.*

```
rm 1kg.map 1kg.ped
plink --vcf 1kg.vcf.gz --make-bed \
  --chr 1-22 --allow-extra-chr \
  --geno 0.01 --out 1kg
```

This command should create *three* new files. What are they?

Remove related samples

Remove 29 of 31 related samples (many population structure analyses are not designed to handle related samples):

```
cut -f 1 20140625_related_individuals.txt \
  > temp.txt
paste temp.txt temp.txt > samples.txt
plink --bfile 1kg --make-bed \
  --remove samples.txt --out 1kg_unrelated
```

Prune SNPs in LD

Many basic population structure analyses such as PCA assume (sometimes implicitly) that the SNPs are independent. A common step is to "prune" SNPs that are strongly correlated with each other (*i.e.*, in "linkage disequibilirium"). *This step may take a few minutes.*

```
plink --bfile 1kg_unrelated \
  --indep-pairwise 1000 500 0.08
plink --bfile 1kg_unrelated --make-bed \
  --extract plink.prune.in \
  --out 1kg_pruned
```

Data preparation: take-home points

- VCFtools and PLINK have *many* commands for manipulating genotype data.
- You can also do a lot with basic shell commands (e.g., awk, grep, cut, head, cat, paste).
- Data processing is important—careless data processing can lead to a poor quality analysis.
- It is helpful to record the data processing steps you took.

Outline of workshop

- Preliminaries
- Programming challenges:
  1. Set up your environment for genetic data analysis.
  2. Prepare genotype data for PCA.
  3. **Run PCA on genotype data.**
  4. Visualize and interpret PCA results.

Challenge #3: Run PCA on genotype data

- Aim: Compute principal components (PCs) from genotype data, which we will use to gain insight into the genetic data.
- Steps:
  1. Convert genotypes to a matrix.
  2. Import genotypes into R.
  3. Fill in missing genotypes.
  4. Compute PCs using rsvd package, and save results.

Convert genotype data to a matrix

The input to PCA should be an $n \times p$ matrix, where n is the number of samples and p is the number of SNPs. The "recode" function in PLINK can be used to create such a matrix.

```
plink --bfile 1kg_pruned \
  --recode A --out 1kg_recoded
gzip 1kg_recoded.raw
```

Note: For those who were unable to follow all the processing steps, I have made a copy of the processed data available here:

```
~pcarbo/share/1kg_recoded.raw.gz
```

Import genotypes into R

Start up an interactive R session:

```
R
```

I wrote a function to import the genotypes into a matrix. It uses the `fread` function from the `data.table` package.

```
library(data.table)
source("functions.R")
```

Load the genotype matrix into R. This may take a minute or two.

```
geno <- read.geno.raw("1kg_recoded.raw.gz")
```

Examine genotypes

Run a few commands to examine the genotype matrix:

```
class(geno)
nrow(geno)
ncol(geno)
geno[1:4,1:4]
```

Fill in missing genotypes

Problem: A small fraction (<1%) of the genotypes are missing:

```
mean(is.na(geno))
```

To run PCA, we need to "fill in" these missing genotypes. In this case, we can reasonably set these missing entries to the mean genotype:

```
p <- ncol(geno)
for (j in 1:p) {
  i          <- which(is.na(geno[,j]))
  geno[i,j] <- mean(geno[,j],na.rm = TRUE)
}
```

Now check that there are no more missing genotypes:

```
sum(is.na(geno))
```

Compute PCs

Use the “rpca” function from the rsvd package to compute the top 2 PCs—that is, the 2 components that explain the most variation in the genotypes. *This step may take a few minutes.*

```
library(rsvd)
pca <- rpca(geno,k = 2,center = TRUE,
            scale = FALSE,retx = TRUE)
```

Compute PCs

Take a quick look at the PCA results:

```
summary(pca)
```

The “x” PCA output gives the projection of the genotype samples onto the 2 PCs:

```
head(pca$x)
```

Let’s save our results:

```
save(file = "1kg_pca.RData",
     list = "pca")
```

PCA analysis: take-home points

- The input to PCA must be a (numeric) matrix with no missing values.
- Other software deals more elegantly with missing data. Here it does not matter much.
- We skipped over some details about encoding genotypes as a matrix.
- See file “pca.sbatch” for an example of automating the PCA analysis using Slurm.

Outline of workshop

- Preliminaries
- Programming challenges:
  1. Set up your environment for genetic data analysis.
  2. Prepare genotype data for PCA.
  3. Run PCA on genotype data.
  4. **Visualize and interpret PCA results.**

Challenge #4: Visualize and interpret PCA results

- Aim: Create plots from the PCA results to gain insight into the genetic data.
- Steps:
  1. Set up R for plotting.
  2. Create a basic PC plot.
  3. Create a PC plot with population labels.

Set up R for plotting

Load the plotting packages and some functions I defined for creating the PCA plots.

```
library(rsvd)
library(ggplot2)
library(cowplot)
source("functions.R")
```

If necessary, load the PCA results.

```
load("1kg_pca.RData")
```

Create a basic PC plot

Use function `basic.pc.plot` to plot all the samples projected onto the top 2 PCs:

```
colnames(pca$x) <- c("PC1","PC2")
p1 <- basic.pc.plot(pca,x = "PC1",y = "PC2",
                    size = 2)
print(p1)
```

You may want to adjust the “size” argument.

Create a PC plot with population labels

To create a plot with labels, we first need to load the 1000 Genomes population labels. They are stored in omni_samples.20141118.panel.

```
labels <-
  read.table("omni_samples.20141118.panel",
             sep = " ",header = TRUE,
             as.is = "id")
```

Create a PC plot with population labels

Align the rows of the "labels" data frame with the rows of the PC matrix:

```
ids <- sapply(strsplit(rownames(pca$x),"_"),"[[",1)
rows <- match(ids,labels$id)
labels <- labels[rows,]
```

Create the PC plot with labels:

```
p2 <- labeled.pc.plot(pca,labels$pop,x = "PC1",
                      y = "PC2",size = 2)
print(p2)
```

Compare to Supplementary Figure 4 of the 1000 Genomes paper (doi:10.1038/nature11632).

Save the labeled PC plot

Save your work as a PDF file using the “ggsave” from the ggplot2 package:

```
ggsave("1kg_pca.pdf",p2)
```

Visualizing and interpreting PCA results

- PCA is the most commonly used approach to infer population structure from genotype data.
- One reason PCA is so popular is that it can produce evocative visualizations of population structure.
- However, there are many pitfalls in interpreting PCA results... proceed with caution!

Recap

Some data analysis techniques we used today:

- Using PLINK and basic shell commands to inspect and process genotype data.
- Importing genotype data into R.
- Analyzing genotype data in R.
- Using ggplot2 to visualize the results of the PCA analysis.
- Following Best Practices on an HPC cluster.