Noname manuscript No.
(will be inserted by the editor)

Exploring population structure with admixture models and principal components analysis

Chi-Chun Liu · Suyash Shringapure · Kenneth Lange · John Novembre ·

Received: date / Accepted: date

Abstract Population structure is a commonplace feature of genetic variation data, and it has importance in numerous application areas, including evolutionary genetics, conservation genetics, and human genetics. Understanding the structure in a sample is necessary before more sophisticated analyses are undertaken. Here we provide a protocol for running principal component analysis (PCA) and admixture proportion inference - two of the most commonly used approaches in describing population structure. Along with hands-on examples with CEPH-Human Genome Diversity Panel and pragmatic caveats, readers will be able to analyze, and visualize population structure on their own data.

Keywords population structure · admixture · principal component analysis · population stratification ·

Chi-Chun Liu
Department of Human Genetics, University of Chicago, Chicago, IL.

Suyash Shringapure
23andme, Mountain View, CA.

Kenneth Lange
Departments of Biomathematics, Human Genetics, and Statistics. University of California, Los Angeles, Los Angeles, CA.

John Novembre
Department of Human Genetics, Ecology and Evolution. University of Chicago. Chicago, IL.

1 Introduction

Population structure is a commonplace feature of genetic variation data, and it has importance in numerous application areas, including evolutionary genetics, conservation genetics, and human genetics. At a broad level, population structure is the existence of differing levels of genetic relatedness among some subgroups within a sample. This may arise for a variety of reasons, but a common cause is that samples have been drawn from geographically isolated groups or different locales across a geographic continuum. Regardless of the cause, understanding the structure in a sample is necessary before more sophisticated analyses are undertaken. For example, to infer divergence times between two populations requires knowing two populations even exist and which individuals belong to each.

Two of the most commonly used approaches to describe population structure in a sample are principal components analysis (Menozzi, Piazza, and Cavalli-Sforza 1978; Cavalli-Sforza, Menozzi, and Piazza 1994; Price et al. 2006; Patterson, Price, and Reich 2006) and admixture proportion inference (Pritchard, Stephens, and Donnelly 2000; Novembre 2016). In brief, principal components analysis reduces a multi-dimensional dataset to a much smaller number of dimensions that allows for visual exploration and compact quantitative summaries. In its application to genetic data, the numerous genotypes observed per individual are reduced to a few summary coordinates. With admixture proportion inference, individuals in a sample are modeled as having a proportion of their genome derived from each of several source populations. The goal is to infer the proportions of ancestry in each source populations, and these proportions can be used to produce compact visual summaries that reveal the existence of population structure in a sample.

The history and basic behaviors of both these approaches have been written about extensively, including by some of us, and so we refer readers to several previous publications to learn the basic background and interpretative nuances of these approaches and their derivatives (Pritchard, Stephens, and Donnelly 2000; Falush, Stephens, and Pritchard 2003; Noah A Rosenberg et al. 2005; Hubisz et al. 2009; Raj, Stephens, and Pritchard 2014; Novembre 2014; Falush, Dorp, and Lawson 2016; David H Alexander, Novembre, and Lange 2009; David H. Alexander and Lange 2011; Price et al. 2006; Patterson, Price, and Reich 2006; Novembre and Stephens 2008; McVean 2009; Novembre and Peter 2016). Here, in the spirit of this volume, we provide a protocol for running these analyses and share some pragmatic caveats that do not always arise in more abstract discussions regarding these methods.

2 Materials

The protocol we present is based on two pieces of software: 1) the `ADMIXTURE` software that our team developed (David H Alexander, Novembre, and Lange 2009) for efficiently estimating admixture proportions in the "Pritchard-Stephens-

Donnelly" model of admixture (Pritchard, Stephens, and Donnelly 2000; Novembre 2016). 2) The `smartpca` software developed by Nick Patterson and colleagues for carrying out PCA (Price et al. 2006). Both of these pieces of software are used widely. We also pair them with downstream tools for visualization, in particular `pong` (Behr et al. 2016), for visualizing output of admixture proportion inferences, and `PCAviz` (Williams et al. 2017), a novel R package for plotting PCA outputs. We also use `PLINK` (S. Purcell et al. 2007; Chang et al. 2015) as a tool to perform some basic manipulations of the data (See Chapter 3 for more background on `PLINK`).

The example data we use is derived from publicly available single-nucleotide polymorphism (SNP) genotype data from the CEPH-Human Genome Diversity Panel (Cann et al. 2002). Specifically, we will look at Illumina 650Y genotyping array data as first described by Li et al (Li et al. 2008). This sample is a global-scale sampling of human diversity with 52 populations in total, and the raw files are available from the following link: `http://hagsc.org/hgdp/files.html`. These data have been used in numerous subsequent publications and are an important reference set.

A few technical details are that the genotypes were filtered with a cutoff of 0.25 for the Illumina GenCall score (Illumina 2005) (a quality score generated by the basic genotype calling software). Further, individuals with a genotype call rate <98.5% were removed, with the logic being that if a sample has many missing genotypes it may due to poor quality of the source DNA, and so none of the genotypes from that individual should be trusted. Beyond this, to prepare the data, we have filtered down the individuals to a set of 938 unrelated individuals. We exclude related individuals as we are not interested in population structure that is due to family relationships and methods such as PCA and `ADMIXTURE` can inadvertently mistake family structure for population structure. The starting data are available as plink-formatted files `H938.bed` `H938.fam`, `H938.bim`, and an accompanying set of population identifiers `H938.clst.txt` in the `raw_input` sub-directory of the repository.

As a pragmatic side note, it is common (and recommended) when carrying out analyses of population structure to merge one's data with other datasets that contain populations which may be representative sources of admixing individuals. For example, in analyzing a dataset with African American individuals, it can be helpful to include datasets containing African and European individuals in the analysis. These datasets can be merged with your dataset using software such as `plink`. However, when merging several datasets, one should be aware of potential biases that can be introduced due to strand flips (i.e. one dataset reports genotypes on the '+' strand of the reference human genome, and another on the '-' strand). One precautionary step to detect strand flips is to group individuals by what dataset they derive from and then produce a scatterplot of allele frequencies for pairs of groups at a time. If strand flips are not being controlled correctly, one will observe numerous variants on the $y = 1 - x$ line, where x is the frequency in one dataset and y is the frequency in a second dataset. (Note: this rule of thumb assumes levels of

differentiation are low between datasets, as is the case in human datasets in general, but one should still keep this in mind interpreting results).

3 Methods

In this section we walk you through an example analysis using `ADMIXTURE` and `smartpca`. We assume the raw data files are in a directory `raw_input` that is below our working directory and that second directory `out` exists in which outputs can be placed. If following along in an `R` console, you should use the `setwd()` command to set the working directory correctly.

3.1 Subsetting Data

For running some simple examples below, we will first create a subset of the HGDP sample that is restricted to only European populations. The European populations in the HGDP have the labels 'Adygei', 'Basque', 'French', 'Italian', 'Orcadian', 'Russian', 'Sardinian' and 'Tuscan', so we create a list of individuals matching these labels using an `awk` command, and then use `plink`'s `--keep` option to make a new dataset with output prefix 'H938_Euro'.

```
awk '$3=="Adygei"||$3=="French_Basque"||$3=="French"|| \
$3=="North_Italian"||$3=="Orcadian"||$3=="Russian"|| \
$3=="Sardinian"||$3=="Tuscan" {print $0}' \
raw_input/H938.clst.txt > out/Euro.clst.txt

plink --bfile raw_input/H938 --keep out/Euro.clst.txt \
--make-bed --out out/H938_Euro
```

3.2 Filter out SNPs to remove linkage disequilibrium (LD)

SNPs in high LD with each other contain redundant information. More worrisome is the potential for some regions of the genome to have a disproportionate influence on the results and thus distort the representation of genome-wide structure. A nice empirical example of the problem is in Figure 5 of Tian et al (Tian et al. 2008), where PC2 of the genome-wide data is shown to be reflecting the variation in a 3.8Mb region of chromosome 8 that is known to harbor an inversion. A standard approach to address this issue is to filter out SNPs based on pairwise LD to produce a reduced set of more independent markers. Here we use `plink`'s commands to produce a new LD-pruned dataset with output prefix `H938_Euro.LDpruned`. The approach considers a chromosomal window of 50 SNPs at a time, and for any pair whose genotypes have an association r^2 value greater than 0.1, it removes a SNP from the pair. Then the window is shifted by 10 SNPs and the procedure is repeated:

```
plink --bfile out/H938_Euro --indep-pairwise 50 10 0.1
plink --bfile out/H938_Euro --extract plink.prune.in --make-bed \
--out out/H938_Euro.LDprune
```

(Advanced note: For particularly sensitive results, we recommend additional rounds of SNP filtering based on observed principal component loadings and/or population differentiation statistics. For example, a robust approach is to filter out large windows around any SNP with a high PCA loading, see Novembre et al. 2008).

3.3 Running ADMIXTURE

3.3.1 An example run with visualization

The ADMIXTURE software (v 1.3.0 here) comes as a pre-compiled binary executable file for either Linux or Mac operating systems. To install, simply download the package and move the executable into your standard execution path (e.g. '/usr/local/bin' on many linux systems). Once installed, it is straightforward to run ADMIXTURE with a fixed number of source populations, commonly denoted by K. For example, to get started let's run ADMIXTURE with K=6:

```
admixture out/H938_Euro.LDprune.bed 6
```

ADMIXTURE is a maximum-likelihood based method, so as the method runs, you will see updates to the log-likelihood as it converges on a solution for the ancestry proportions and allele frequencies that maximize the likelhood function. The algorithm will stop when the difference between successive iterations is small (the 'delta' value takes a small value). A final output is an estimated F_{ST} value (Holsinger and Weir 2009) between each of the source populations, based on the inferred allele frequencies. These estimates reflect how differentiated the source populations are, which is important for understanding whether the population structure observed in a sample is substantial or not (values closer to 0 reflect less population differentiation).

After running, ADMIXTURE produces two major output files. The file with suffix .P contains an $L \times K$ table of the allele frequencies inferred for each SNP in each population. The file with suffix .Q contains an $N \times K$ table of inferred individual ancestry proportions from the K ancestral populations, with one row per individual.

For our example dataset with K=6, this will be a file called H938.LDpruned.6.Q. This file can be used to generate a plot showing individual ancestry. In R, this can be done using the following commands:

```
library(RColorBrewer)
tbl <- read.table("out/H938_Euro.LDprune.6.Q")
```

```
par(mar = c(1.5, 4, 2.5, 2),cex.lab=0.75,cex.axis=0.6)
barplot(t(as.matrix(tbl)),
  col = brewer.pal(6, "Set1"), ylab = "Anc. Proportions",
  border = NA, space = 0
)
```

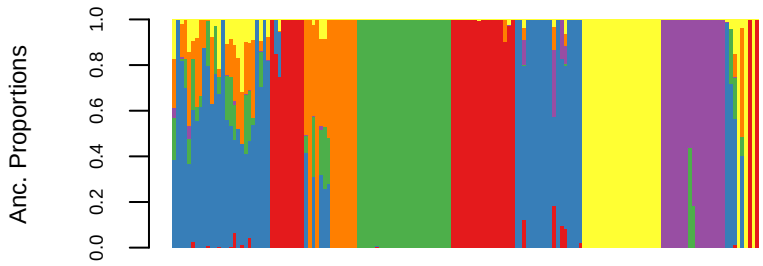

Fig. 1 Initial rough plot of the ADMIXTURE results for K=6 using R base graphics.

Each thin vertical line in the barplot represents one individual and each color represents one inferred ancestral population. The length of each color in a vertical bar represents the proportion of that individual's ancestry that is derived from the inferred ancestral population corresponding to that color. The above image suggests there are some genetic clusters in the data, but it's not a well-organized data display.

To improve the visualization, one can use a package dedicated to plotting ancestry proportions (Noah A. Rosenberg 2004; Kopelman et al. 2015; Behr et al. 2016). Here we use a post-processing tool `pong` (Behr et al. 2016), which visualizes individual ancestry with similarity between individuals within clusters. You will most likely want to install `pong` on a local machine as it initializes a local web server to display the results.

To run `pong` requires setting up a few files: 1) an `ind2pop` file that maps individuals to populations; 2) a `Qfilemap` file that points `pong` towards which '.Q' files to display; These are easy to build up from the command-line using the `Euro.clst.txt` file we built above, and an `awk` command to output tab-separated text to a file with the `Qfilemap` suffix added to whatever file prefix we're using to organize our runs:

```
cut -d' ' -f3 out/Euro.clst.txt > out/H938_Euro.ind2pop
FILEPREFIX=H938_Euro.LDPrune
K=6
awk -v K=$K -v file=$FILEPREFIX 'BEGIN{ \
printf("ExampleRun\t%d\t%s.%d.Q\n",K,file,K)
}' > out/$FILEPREFIX.Qfilemap
```

Note when building the `.Qfilemap` one needs to use tabs to separate the columns for `pong` to read the file correctly.

Then to run `pong`, we use following command:

```
pong -m out/H938_Euro.LDprune.Qfilemap -i out/H938_Euro.ind2pop
```

We open a web browser to `http://localhost:4000/` to view the results. Here's an example of what you should see:

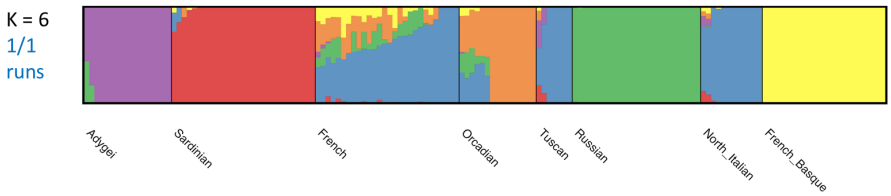

Fig. 2 Plot of the ADMIXTURE results for K=6 using PONG.

From this visualization, we can see the admixture model fits most individuals of the Adygei, Sardinian, Russian, French Basque samples as being derived each from a single source population (represented by purple, red, green, and yellow respectively). The French, Tuscan, and North Italian samples are generally estimated to have a majority component of ancestry from a single source population (blue) though with admixture with other sources. A first conclusions is that the population labels do not capture the complexity of the population structure. There is apparent cryptic structure within some samples (e.g., Orcadian) and minimal differentiation between other samples (North Italian and Tuscan samples for instance).

Because ADMIXTURE is a "greedy", "hill-climbing"" optimization algorithm it is good practice to do multiple runs from different initial random starting points. We can do this by using the -s flag to specify the random seed for each ADMIXTURE run.

```
K=6
prefix=H938_Euro.LDprune
# run admixture multiple times
for r in {1..10}
do
admixture -s ${RANDOM} out/H938_Euro.LDprune.bed $K
mv out/${prefix}.${K}.Q out/${prefix}.K${K}r${r}.Q
done
```

Pong has nice functionality for summarizing the output of the multiple ADMIXTURE runs. It can collect similar solutions into "modes" and display them in ranked order of the number of runs supporting each. In the interactive version, you use the `check to highlight multimodality` checkbox and whiten populations with ancestry matrices agreeing with the major mode. One

can also click on and visualize only one cluster. Here we set up the PONG input files and show an example output.

```
K=6
prefix=H938_Euro.LDprune
# create a pong parameter file
for r in {1..10}
do
awk -v K=$K -v r=$r -v file=${prefix}.K${K}r${r} 'BEGIN{ \
printf("K%dr%d\t%d\t%s.Q\n",K,r,K,file)
}' >> out/${prefix}.k6multiplerun.Qfilemap
done
# run pong
pong -m out/${prefix}.k6multiplerun.Qfilemap --greedy -s .95 \
-i out/H938_Euro.ind2pop
```

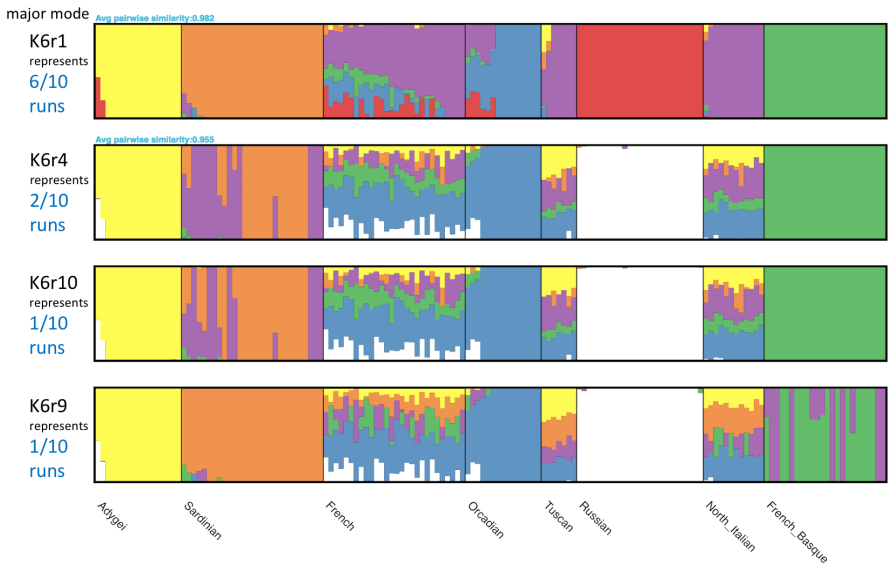

Fig. 3 PONG plot summarizing multiple ADMIXTURE runs with different random starting points. The top row shows the major mode (supported by 6 out of 10 runs as indicated in the blue text). The next three rows show three other solutions found by ADMIXTURE in 2, 1, and 1 runs respectively.

The resulting figure shows that 6 out of 10 runs converged to the same mode, which appears equivalent to our initial run above. We observe the appearance of structure within Sardinia in the second and third modes. The original run had North Italian and Tuscan samples as a mostly unadmixed, while all 3 minor modes model the two population as highly admixed. The

fourth mode (supporting by just one run) inferred sub-structure within the French Basque sample. This instability in the solution is a hint that the ADMIXTURE model with K=6 is not a perfect fit to this data.

3.3.2 Considering different values of K

In a typical analysis, one wants to explore the sensitivity of the results to the choice of K. One approach is to run ADMIXTURE with various plausible values of K and compare the performance of results visually and using cross-validation error rates. Here is a piece of `bash` command-line code that will run `ADMIXTURE` for values of K from 2 to 12, and that will build a file with a table of cross-validation error rates per value of K.

```
# Run for different values of K
prefix=H938_Euro.LDprune
Klow=1
Khigh=12
for ((K=$Klow;K<=$Khigh;K++)); \
do
  admixture --cv out/$prefix.bed $K | tee log.$prefix.${K}.out;
done
```

Then let's compile results on cross-validation error across values of K:

```
prefix=H938_Euro.LDprune
Klow=1
Khigh=12
echo '# CV results' > $prefix.CV.txt
for ((K=$Klow;K<=$Khigh;K++)); do
  awk -v K=$K '$1=="CV"{print K,$4}' out/log.$prefix.$K.out \
  >> out/$prefix.CV.txt;
done
```

Now let's inspect the outputs. First let's make a plot of the cross-validation error as a function of K:

```
tbl <- read.table("out/H938_Euro.LDprune.CV.txt")
par(mar = c(4, 4, 2, 2),cex=0.7)
plot(tbl$V1, tbl$V2, xlab = "K", ylab = "Cross-validation error",
     pch = 16, type = "l")
```

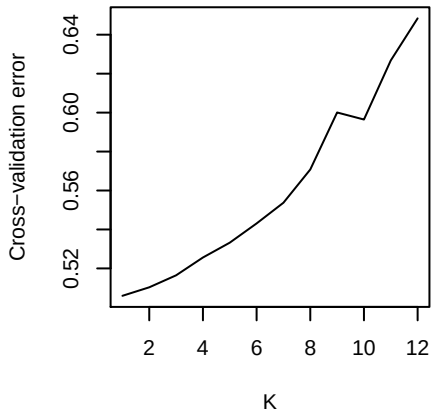

Fig. 4 Cross-validation error as a function of K for the example dataset.

The cross-validation error suggests a single source population can model the data adequately and larger values of K lead to over-fitting.

To inspect further, we can use the `pong` software to visualize the ancestry components inferred at different K across sevderal runs. We need to set-up some the results and input files first.

```
# Run for different values of K, each with 10 runs
prefix=H938_Euro.LDprune
for r in {1..10}; do for K in {2..12};
do
  admixture -s ${RANDOM} out/${prefix}.bed $K
  mv out/${prefix}.${K}.Q out/${prefix}.K${K}r${r}.Q
done; done

# create Qmap file for pong
createQmap(){
local r=$1
local K=$2
awk -v K=$K -v r=$r -v file=${prefix}.K${K}r${r} 'BEGIN{ \
printf("K%dr%d\t%d\t%s.Q\n",K,r,K,file)
}' >> out/${prefix}.multiplerun.Qfilemap
}
export -f createQmap
for K in {2..12}; do for r in {1..10}; do createQmap $r $K; \
done; done

#run pong
pong -m out/H938_Euro.LDprune.multiplerun.Qfilemap --greedy \
-s.95 -i out/H938_Euro.ind2pop
```

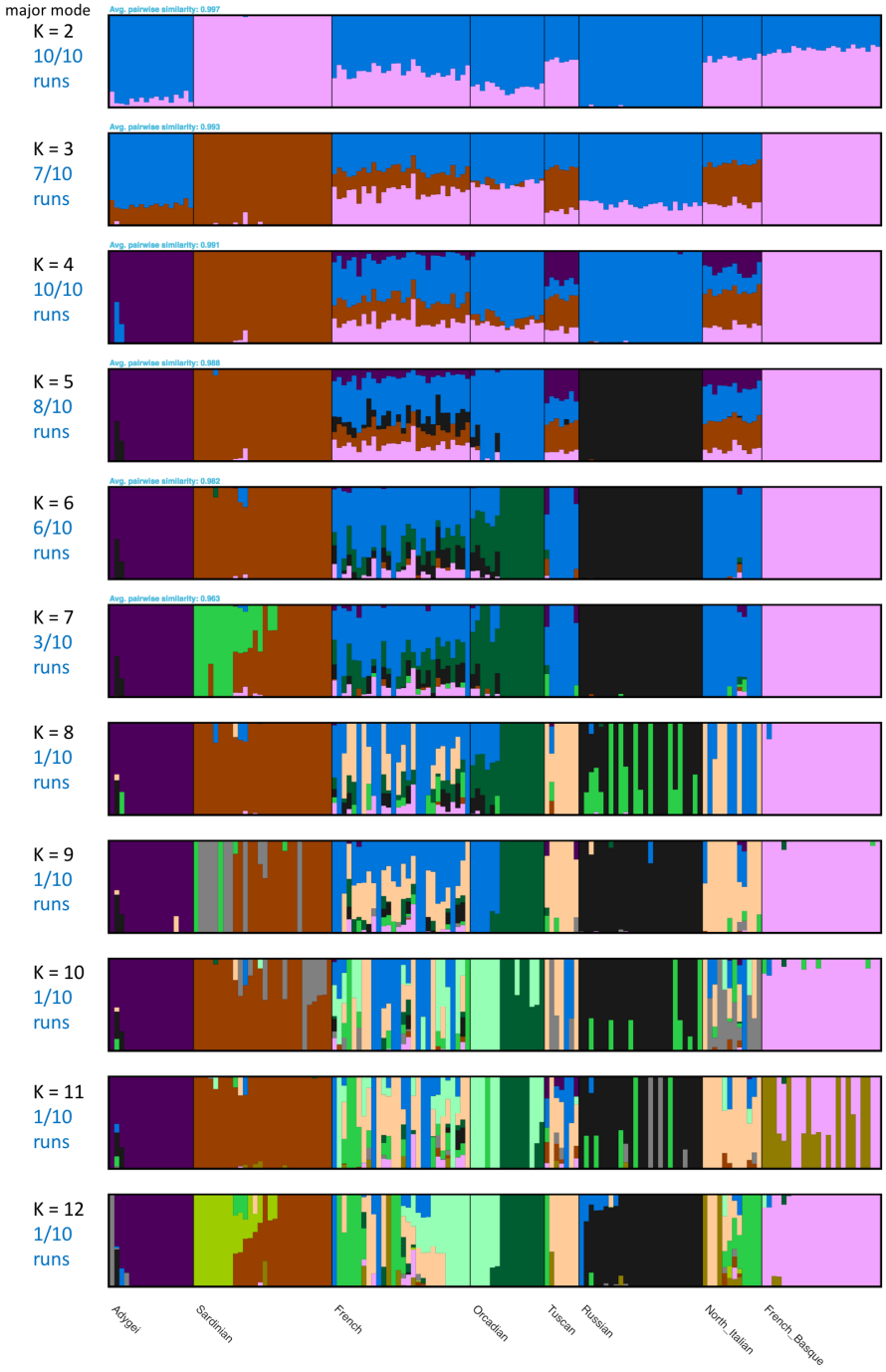

Fig. 5 Example PONG output showing results from across a range of K values (with ten ADMIXTURE runs per K value)

Here we find how as K increases through to $K = 6$, the Sardinian, Basque, Adygei, and Russian samples are typically modeled as descended from unique sources, and at K of 7, 8, 9 we find structure within the Sardinian, Russian, Orcadian, and Basque samples is revealed, though for each, the substructure is not very stable. The values of $K = 10$ and above make increasingly finer scale divisions that are difficult to interpret, and the major modesfor $K = 7$ and up only consist of one to three runs, suggesting a very multi-modal likelihood surface and a poor resolution of the population structure.

Overall, its interesting to note that the visual inspection of the results suggests several "real" clusters in the data, supported by an alignment of the clustering with known population labels, even though the cross-validation supports a value of $K = 1$. This highlights a long-standing known issue with admixutre modeling: the selection of K is a difficult problem to automate in a way that is robust.

3.3.3 Some advanced options

Running `ADMIXTURE` with the `-B` option provides estimates of standard errors on the ancestry proportion inferences. The `-l` flag runs `ADMIXTURE` with a penalized likelihood that favors more sparse solutions (i.e., ancestry proportions that are closer to zero). This is useful in settings where small, possibly erroneous ancestry proportions may be overinterpreted. By using the `-P` option, the population allele frequencies inferred from one dataset can be provided as input for inference of admixture proportions in a second dataset. This is useful when individuals of unknown ancestry are being analyzed against the background of a reference sample set. Please see the `ADMIXTURE` manual for a complete listing of options and more detail, and we encourage testing these options in test datasets such as the one provided here.

3.4 PCA with SMARTPCA

3.4.1 Running PCA

Comparing ADMIXTURE and PCA results often helps give insight and confirmation regarding population structure in a sample. To run PCA, a standard package that is well-suited for SNP data is the `smartpca` package maintained by Nick Patterson and Alkes Price (at `http://data.broadinstitute.org/alkesgroup/EIGENSOFT/`). To run it, we first set-up a basic `smartpca` parameter file from the command-line of a `bash` shell:

```
PREFIX=H938_Euro.LDprune
echo genotypename: out/$PREFIX.bed > out/$PREFIX.par
echo snpname: out/H938_Euro.LDprune.bim >> out/$PREFIX.par
echo indivname: out/H938_Euro.LDprune.PCA.fam >> out/$PREFIX.par
echo snpweightoutname: out/H938_Euro.LDprune.snpeigs \
```

```
>> out/$PREFIX.par
echo evecoutname: out/H938_Euro.LDprune.eigs >> out/$PREFIX.par
echo evaloutname: out/H938_Euro.LDprune.eval >> out/$PREFIX.par
echo phylipoutname: out/H938_Euro.LDprune.fst >> out/$PREFIX.par
echo numoutevec: 20 >> out/$PREFIX.par
echo numoutlieriter: 0 >> out/$PREFIX.par
echo outlieroutname: out/H938_Euro.LDprune.out >> out/$PREFIX.par
echo altnormstyle: NO >> out/$PREFIX.par
echo missingmode: NO >> out/$PREFIX.par
echo nsnpldregress: 0 >> out/$PREFIX.par
echo noxdata: YES >> out/$PREFIX.par
echo nomalexhet: YES >> out/$PREFIX.par
```

This input parameter file runs `smartpca` in its most basic mode (i.e. no automatic outlier removal or adjustments for LD - features which you might want to explore later).

As a minor issue, `smartpca` ignores individuals in the `.fam` file if they are marked as missing in the phenotypes column. This `awk` command provides a new `.fam` file that will automatically include all individuals.

```
awk '{print $1,$2,$3,$4,$5,1}' out/H938_Euro.LDprune.fam \
> out/H938_Euro.LDprune.PCA.fam
```

Now run `smartpca` with the following command.

```
smartpca -p ./out/H938_Euro.LDprune.par
```

You will find the output files in the `out` sub-directory as specified in the parameter file.

3.4.2 Plotting PCA results with PCAviz

The PCAviz package can be found at `https://github.com/NovembreLab/PCAviz`. It provides a simple interface for quickly creating plots from PCA results. It encodes several of our favored best practices for plotting PCA (such as using abbreviations for point characters and plotting median positions of each labelled group). To install the package use:

```
install.packages("devtools")
devtools::install_github("NovembreLab/PCAviz",
                         build_vignettes = TRUE)
```

The following command in `R` generates plots showing each individual sample's position in the PCA space and the median position of each labelled group in PCA space:

```
library(PCAviz)
library(cowplot)
prefix <- "out/H938_Euro.LDprune"
nPCs <- 20

# Read in individual coordinates on PCs and eignvalues
PCA <- read.table(paste(prefix, ".eigs", sep = ""))
names(PCA) <- c("ID", paste("PC", (1:nPCs), sep = ""),
                  "case.control")
PCA <- PCA[, 1:(nPCs + 1)] # Remove case/control column
eig.val <- sqrt(unlist(read.table(
  paste(prefix, ".eval", sep = "")))[1:nPCs])
sum.eig <- sum(unlist(read.table(
  paste(prefix, ".eval", sep = ""))))

# Read in snp weightings matrix
snpeigs <- read.table(paste(prefix, ".snpeigs", sep = ""))
names(snpeigs) <- c("ID", "chr", "pos",
                    paste("PC", (1:nPCs), sep = ""))
snpeigs$chr <- factor(snpeigs$chr)
rownames(snpeigs) <- snpeigs$ID
snpeigs <- snpeigs[, -1]

# Note smartpca pushes the plink family and individual
# ids together so we need to extract out the ids afresh
tmp <- unlist(sapply(as.character(PCA$ID), strsplit, ":"))
ids <- tmp[seq(2, length(tmp), by = 2)]
PCA$ID <- ids

# Read in the group/cluster labels
clst <- read.table("out/Euro.clst.txt")
# Order them to match the ids of PCA object
clst_unord <- clst$V3[match(ids, clst$V2)]
PCA <- as.data.frame(PCA)
PCA <- cbind(PCA, clst_unord)
names(PCA)[ncol(PCA)] <- "sample"

# Build the PCAviz object
hgdp <- pcaviz(dat = PCA, sdev = eig.val,
               var = sum.eig, rotation = snpeigs)
hgdp <- pcaviz_abbreviate_var(hgdp, "sample")

# Make PCA plots
geom.point.summary.params <- list(
  shape = 16, stroke = 1, size = 5,
```

```
  alpha = .7, show.legend = F
)
plot1 <- plot(hgdp,
  coords = paste0("PC", c(1, 2)), color = "sample",
  geom.point.summary.params = geom.point.summary.params,
  scale.pc.axes = 0.6
)
plot2 <- plot(hgdp,
  coords = paste0("PC", c(2, 3)), color = "sample",
  geom.point.summary.params = geom.point.summary.params,
  scale.pc.axes = 0.6
)
plot3 <- plot(hgdp,
  coords = paste0("PC", c(4, 5)), color = "sample",
  geom.point.summary.params = geom.point.summary.params,
  scale.pc.axes = 0.6
)
plot4 <- plot(hgdp,
  coords = paste0("PC", c(5, 6)), color = "sample",
  geom.point.summary.params = geom.point.summary.params,
  scale.pc.axes = 0.6
)

plot_grid(plot1, plot2, plot3, plot4)
```

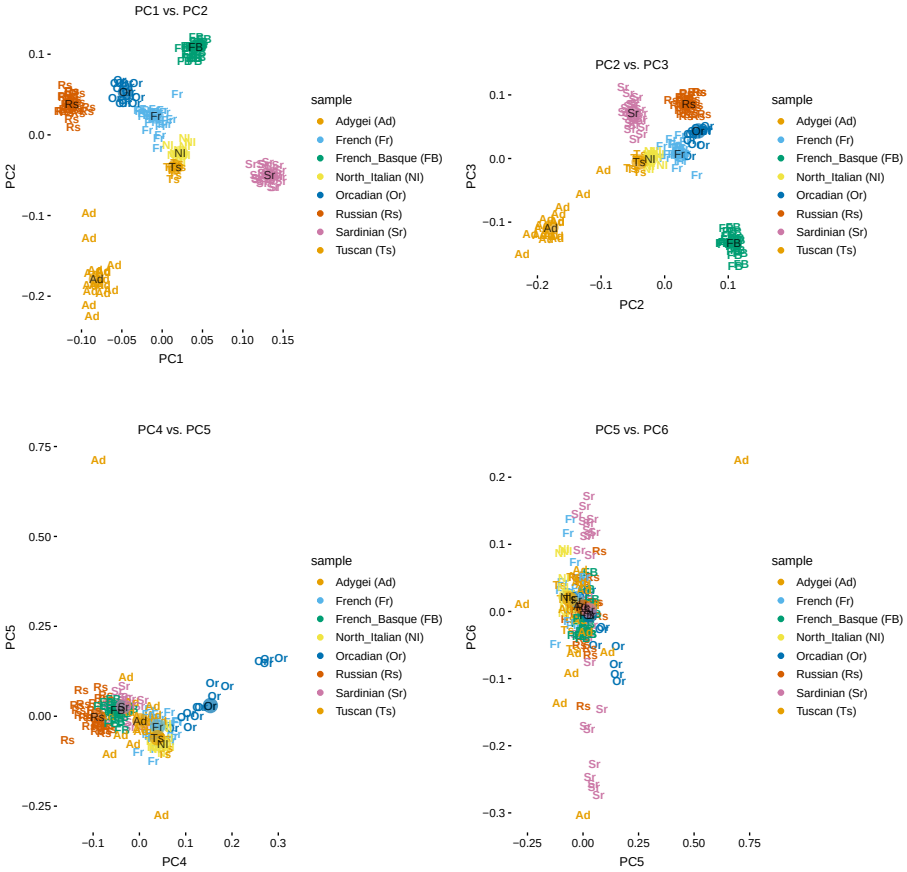

Fig. 6 Pairwise plots of PC scores generated using the PCAviz package.

First one may notice several populations are separated with PC1 and PC2, with the more isolated populations being those that were most distinguished from the others by `ADMIXTURE`. PC4 distinguishes a subset of Orcadian individuals and PC5 distinguishes two Adygei individuals. PC6 corresponds to the cryptic structure observed within Sardinians in the `ADMIXTURE` analysis.

As an alternative visualization, it can be helpful to see the distribution of PC coordinates per population for each labelled group in the data:

```
pcaviz_violin(hgdp, pc.dims = paste0("PC", c(1:3)),
              plot.grid.params = list(nrow = 3))
```

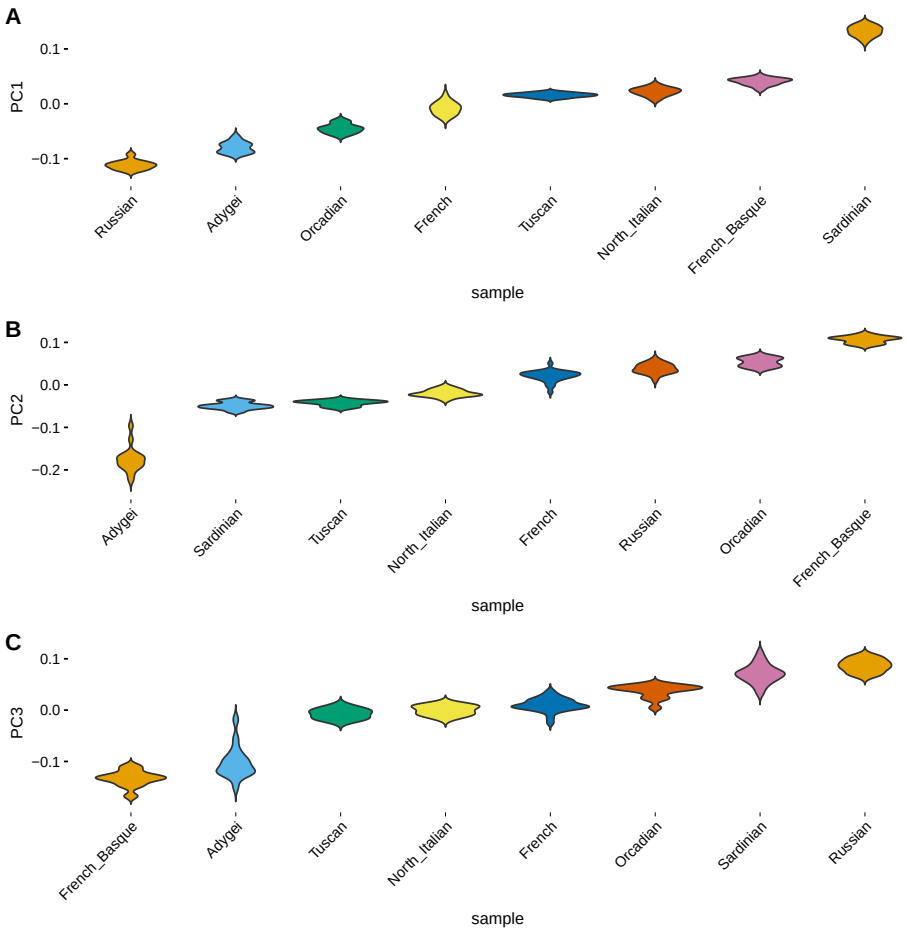

Fig. 7 Violin plots of PC values generated using the PCAviz package.

As mentioned above in the section on LD, it is useful to inspect the PC loadings to ensure that they broadly represent variation across the genome, rather than one or a small number of genomic regions (Duforet-Frebourg et al. 2016). SNPs that are selected in the same direction as genome-wide structure can show high loadings, but what is particularly pathological is if the only SNPs that show high loadings are all concentrated in a single region of the genome, as might occur if the PCA is explaining genomic structure (such as an inversion) rather than population structure.

```
for (i in 1:5) {
  plotname <- paste("plot", i, sep = "")
  plot <- pcaviz_loadingsplot(hgdp,
    pc.dim = paste0("PC", i),
```

```
    min.rank = 0.8, gap = 200, color = "chr",
    geom.point.params = list(show.legend = FALSE)
  ) +
    xlab("SNPs") + ylab(paste0("PC", i, " loading"))
  assign(plotname, plot)
}
# grep common legend
plot <- pcaviz_loadingsplot(hgdp,
  pc.dim = paste0("PC", 1),
  min.rank = 0.8, gap = 200, color = "chr"
) +
  guides(color = guide_legend(nrow = 2, byrow = TRUE)) +
  theme(legend.position = "bottom",
        legend.justification = "center")
plot_legend <- get_legend(plot)
# plot loadings
prow <- plot_grid(plot1, plot2, plot3, plot4, plot5,
                  nrow = 5, align = "vh")
plot_grid(prow, plot_legend, ncol = 1, rel_heights = c(1, .2))
```

Fig. 8 PC loading plots generated using the PCAviz package.

The proportion of total variance explained by each PC is a useful metric for understanding structure in a sample and for evaluating how many PCs one might want to include in downstream analyses. This can be computed as $\lambda_i / \sum_k \lambda_k$, with λ_i being eigenvalues in decreasing order, and is plotted below:

```
screeplot(hgdp, type = "pve") +
  ylim(0, 0.018) +
  ylab('Proportion of Variance Explained') +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  theme(axis.line = element_line(size = 1, linetype = "solid"))
```

```
## Scale for 'y' is already present. Adding another scale for 'y', which
## will replace the existing scale.
```

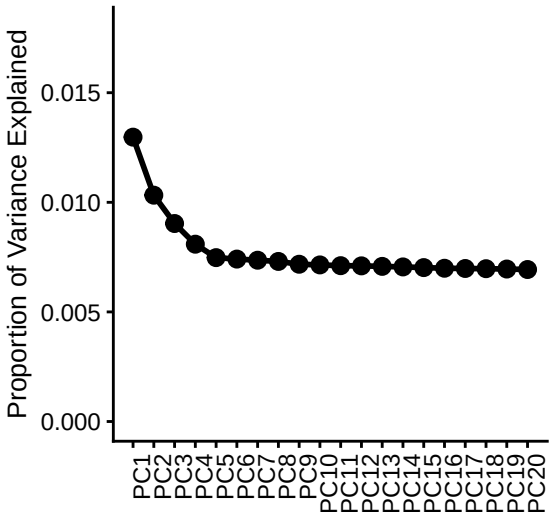

Fig. 9 Proportion of variance explained by each PC. Plot generated using the PCAviz package.

The results show that the top PCs only explain a small fraction of the variance (<1.5%) and that after about $K = 6$ the variance explained per PC becomes relatively constant; roughly in line with the visual inspection of the `admixture` results that revealed $K = 6$ may be reasonable for this dataset.

4 Discussion

Our protocol above is relatively straightforward and presents the most basic implementation of these analyses. Each analysis sofware (`ADMIXTURE` and `smartpca`) and each visualization package (`pong` and `PCAviz`) contain numerous other options that may be suitable for specific analyses and we encourage

the readers to spend time in the manuals of each. Nonetheless, what we have presented is a useful start and a standard pipeline that we use in our research.

Two broad perspectives we find helpful useful to keep in mind are: 1) How the admixture model and PCA framework are related to each other indirectly as different forms of sparse factor analysis (Engelhardt and Stephens 2010); 2) How the PCA framework in particular can be considered as a form of efficient data compression. Both of these perspectives can be helpful in interpreting the outputs of the methods and for appreciating how these approaches best serve as helpful visual exploratory tools for analyzing structure in genetic data. These methods are ultimately relatively simple statistical tools being used to summarize complex realities. They are part of the toolkit for analysis, and often are extremely useful for framing specific models of population structure that can be further investigated using more detailed and explicit approaches (such as those based on coalescent or diffusion theory, Chapters 7 on MSMC and 8 on CoalHMM).

References

Alexander, David H, John Novembre, and Kenneth Lange. 2009. "Fast model-based estimation of ancestry in unrelated individuals." *Genome Res* 19 (9): 1655–64. doi:10.1101/gr.094052.109.

Alexander, David H., and Kenneth. Lange. 2011. "Enhancements to the Admixture Algorithm for Individual Ancestry Estimation." *BMC Bioinformatics* 12 (June): 246. doi:10.1186/1471-2105-12-246.

Behr, Aaron A., Katherine Z. Liu, Gracie Liu-Fang, Priyanka Nakka, and Sohini Ramachandran. 2016. "Pong: Fast Analysis and Visualization of Latent Clusters in Population Genetic Data." *Bioinformatics* 32 (18): 2817–23. doi:10.1093/bioinformatics/btw327.

Cann, Howard M., Claudia de Toma, Lucien Cazes, Marie-Fernande Legrand, Valerie Morel, Laurence Piouffre, Julia Bodmer, et al. 2002. "A Human Genome Diversity Cell Line Panel." *Science* 296 (5566). American Association for the Advancement of Science: 261–62. doi:10.1126/science.296.5566.261b.

Cavalli-Sforza, L Luca, Paolo Menozzi, and Alberto Piazza. 1994. *The history and geography of human genes*. Princeton University Press. doi:10.2307/2058750.

Chang, Christopher C, Carson C Chow, Laurent CAM Tellier, Shashaank Vattikuti, Shaun M Purcell, and James J Lee. 2015. "Second-Generation Plink: Rising to the Challenge of Larger and Richer Datasets." *GigaScience* 4 (1): s13742–015–0047–0048. doi:10.1186/s13742-015-0047-8.

Duforet-Frebourg, Nicolas, Keurcien Luu, Guillaume Laval, Eric Bazin, and Michael G.B. Blum. 2016. "Detecting Genomic Signatures of Natural Selection with Principal Component Analysis: Application to the 1000 Genomes Data." *Molecular Biology and Evolution* 33 (4): 1082–93. doi:10.1093/molbev/msv334.

Engelhardt, Barbara E., and Matthew Stephens. 2010. "Analysis of Population Structure: A Unifying Framework and Novel Methods Based on Sparse Factor Analysis." *PLOS Genetics* 6 (9). Public Library of Science: 1–12. doi:10.1371/journal.pgen.1001117.

Falush, Daniel, Lucy van Dorp, and Daniel Lawson. 2016. "A Tutorial on How (Not) to over-Interpret Structure/Admixture Bar Plots." *bioRxiv*. Cold Spring Harbor Laboratory. doi:10.1101/066431.

Falush, Daniel, Matthew Stephens, and Jonathan K Pritchard. 2003. "Inference of Population Structure Using Multilocus Genotype Data: Linked Loci and Correlated Allele Frequencies." *Genetics* 164 (4): 471–92.

Holsinger, Kent, and Bruce Weir. 2009. "Genetics in Geographically Structured Populations: Defining, Estimating and Interpreting Fst" 10 (October): 639–50.

Hubisz, Melissa J, Daniel Falush, Matthew Stephens, and Jonathan K Pritchard. 2009. "Inferring Weak Population Structure with the Assistance

of Sample Group Information." *Molecular Ecology Resources* 9 (5): 1322–32. doi:10.1111/j.1755-0998.2009.02591.x.

Illumina, Inc. 2005. *Illumina Gencall Data Analysis Software*. `https://www.illumina.com/Documents/products/technotes/technote_gencall_data_analysis_software.pdf`.

Kopelman, Naama M., Jonathan. Mayzel, Mattias. Jakobsson, Noah A. Rosenberg, and Itay. Mayrose. 2015. "Clumpak: A Program for Identifying Clustering Modes and Packaging Population Structure Inferences Across K." *Molecular Ecology Resources* 15 (5). Wiley Online Library: 1179–91. doi:10.1111/1755-0998.12387.

Li, Jun Z., Devin M. Absher, Hua Tang, Audrey M. Southwick, Amanda M. Casto, Sohini Ramachandran, Howard M. Cann, et al. 2008. "Worldwide Human Relationships Inferred from Genome-Wide Patterns of Variation." *Science* 319 (5866). American Association for the Advancement of Science: 1100–1104. doi:10.1126/science.1153717.

McVean, Gil. 2009. "A Genealogical Interpretation of Principal Components Analysis." *PLoS Genetics* 5 (10): e1000686. doi:10.1371/journal.pgen.1000686.

Menozzi, Paolo, Alberto Piazza, and L Luca Cavalli-Sforza. 1978. "Synthetic Maps of Human Gene Frequencies in Europeans." *Science* 201 (4358): 786–92.

Novembre, John. 2014. "Variations on a Common Structure: New Algorithms for a Valuable Model." *Genetics* 197 (3): 809–11. doi:10.1534/genetics.114.166264.

———. 2016. "Pritchard, Stephens, and Donnelly on Population Structure." *Genetics* 204 (2): 391–93. doi:10.1534/genetics.116.195164.

Novembre, John, and Benjamin M Peter. 2016. "Recent Advances in the Study of Fine-Scale Population Structure in Humans." *Curr Opin Genet Dev* 41 (December): 98–105. doi:10.1016/j.gde.2016.08.007.

Novembre, John, and Matthew Stephens. 2008. "Interpreting Principal Component Analyses of Spatial Population Genetic Variation." *Nature Genetics* 40 (5): 646–49. doi:10.1038/ng.139.

Novembre, John, Toby Johnson, Katarzyna Bryc, Zoltan Kutalik, Adam R Boyko, Adam Auton, Amit Indap, et al. 2008. "Genes Mirror Geography Within Europe" 456 (December): 274.

Patterson, Nick J, Alkes L Price, and David Reich. 2006. "Population Structure and Eigenanalysis." *PLoS Genetics* 2 (12): 2074–93. doi:10.1371/journal.pgen.0020190.

Price, Alkes L, Nick J Patterson, Robert M Plenge, Michael E Weinblatt, Nancy A Shadick, and David Reich. 2006. "Principal components analysis corrects for stratification in genome-wide association studies." *Nat Genet* 38 (8): 904–9. doi:10.1038/ng1847.

Pritchard, Jonathan K, Matthew Stephens, and Peter Donnelly. 2000. "Inference of Population Structure Using Multilocus Genotype Data." *Genetics* 155 (2): 945–59.

Purcell, Shaun, Benjamin Neale, Katherine Todd-Brown, Lori Thomas, Manuel A.R. Ferreira, David Bender, Julian Maller, et al. 2007. "Plink: A Tool

Set for Whole-Genome Association and Population-Based Linkage Analyses" 81 (October): 559–75.

Raj, Anil, Matthew Stephens, and Jonathan K Pritchard. 2014. "Fast-STRUCTURE: Variational Inference of Population Structure in Large SNP Data Sets." *Genetics* 197 (2): 573–89. doi:10.1534/genetics.114.164350.

Rosenberg, Noah A, Saurabh Mahajan, Sohini Ramachandran, Chengfeng Zhao, Jonathan K Pritchard, and Marcus W Feldman. 2005. "Clines, Clusters, and the Effect of Study Design on the Inference of Human Population Structure." *PLoS Genetics* 1 (6): e70. doi:https://doi.org/10.1371/journal.pgen.0010070.

Rosenberg, Noah A. 2004. "DISTRUCT: A Program for the Graphical Display of Population Structure." *Molecular Ecology Notes* 4 (1). Wiley Online Library: 137–38. doi:10.1046/j.1471-8286.2003.00566.x.

Tian, Chao, Robert M Plenge, Michael Ransom, Annette Lee, Pablo Villoslada, Carlo Selmi, Lars Klareskog, et al. 2008. "Analysis and Application of European Genetic Substructure Using 300 K SNP Information." *PLoS Genet* 4 (1): e4. doi:10.1371/journal.pgen.0040004.

Williams, Richard, Hossein Pourreza, Yuexi Wang, Peter Carbonetto, and John Novembre. 2017. "PCAviz: Visualizing Principal Components Analysis." `http://github.com/NovembreLab/PCAviz`.