Exploring population structure with admixture models and principal components analysis

Chi-Chun Liu, Suyash Shringapure, Ken Lange, John Novembre

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 et al. 1978; Cavalli-Sforza et al. 1994; Patterson et al. 2006; Price et al. 2006) and admixture proportion inference (Pritchard et al. 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 quantitive 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 extentsively, 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 et al. 2000; Falush et al. 2003, 2016; Rosenberg et al. 2005; Patterson et al. 2006; Price et al. 2006; Novembre and Stephens 2008; Hubisz et al. 2009; McVean 2009; Alexander et al. 2009; Alexander and Lange 2011; Raj et al. 2014; Novembre 2014; 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.

Materials

The protocol we present is based on two pieces of software: 1) the `ADMIXTURE` software that our team developed (Alexander et al. 2009) for efficiently estimating admixture proportions in the "Pritchard-Stephens-Donnelly" model of admixture (Pritchard et al. 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` (Purcell et al. 2007; Chang et al. 2015) as a tool to perform some basic manipulations of the data (See Chapter X 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`, from this link: https://www.dropbox.com/sh/66p1hhbtx0rb3wt/AAA0aDXVTNhjUi5nH03___D4za?dl=0.

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).

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.

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
```

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).

Running `ADMIXTURE`

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=8 as there are 8 distinctly labelled populations in the dataset:

```
admixture out/H938_Euro.LDprune.bed 8
```

`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 likelihood 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=8, this will be a file called `H938.LDpruned.8.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.8.Q")
par(mar=c(1,4,2,2))
barplot(t(as.matrix(tbl)), col=brewer.pal(8,"Set1"),
        ylab="Admixture Proportions",border=NA,space=0)
```
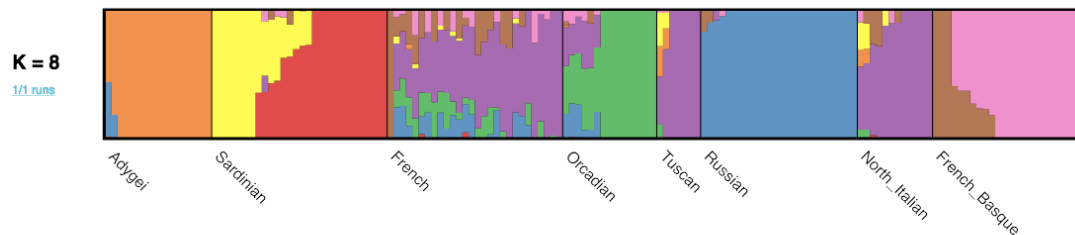
Figure 1: Example PONG output

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 (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-seperated 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=out/H938_Euro.LDPrune
K=8
awk -v K=$K -v file=$FILEPREFIX 'BEGIN{ \
printf("ExampleRun\t%d\t%s.%d.Q\n",K,file,K)
}' > $FILEPREFIX.Qfilemap
```

Note when building the `.Qfilemap` one needs to use tabs to seperate 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
```

and open a web browser to `http://localhost:4000/` to view the results.

Here's an example of what you should see:

From this visualization we can see the admixture model fits most individuals of the Adygei, Orcadian, and Russian samples as being derived each from a single source population (represented by orange, green, and blue respectively); the Sardinian and the French Basque samples are modeled as comprising individuals from two source populations each (yellow/red for Sardinia, brown/pink for French Basque); and the French, Tuscan, and North Italian samples are generally estimated to have a majority component of ancestry from a single source population ('purple') though with notable 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., Sardinia) and minimal differentiation between other samples (North Italian and Tuscan samples for instance).

We can run ADMIXTURE multiple times with -s flag specifying random seeds. This is useful determining if ADMIXTURE finds global optima. Pong summarizes modality and provides `check to highlight multimodality` checkbox whitening populations with ancestry matrices agreeing the mode. One can also click on and visualize only one cluster.

```
# run admixture multiple times
K=6
prefix=H938_Euro.LDprune

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
# 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}.multiplerun.Qfilemap
done
# run pong
pong -m out/${prefix}.multiplerun.Qfilemap -s .99 -i out/H938_Euro.ind2pop
```

Here inferred ancestry components for North Italian are different across multiple runs of ADMIXTURE.

Considering different values of K

In a typical analysis, one typically 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:
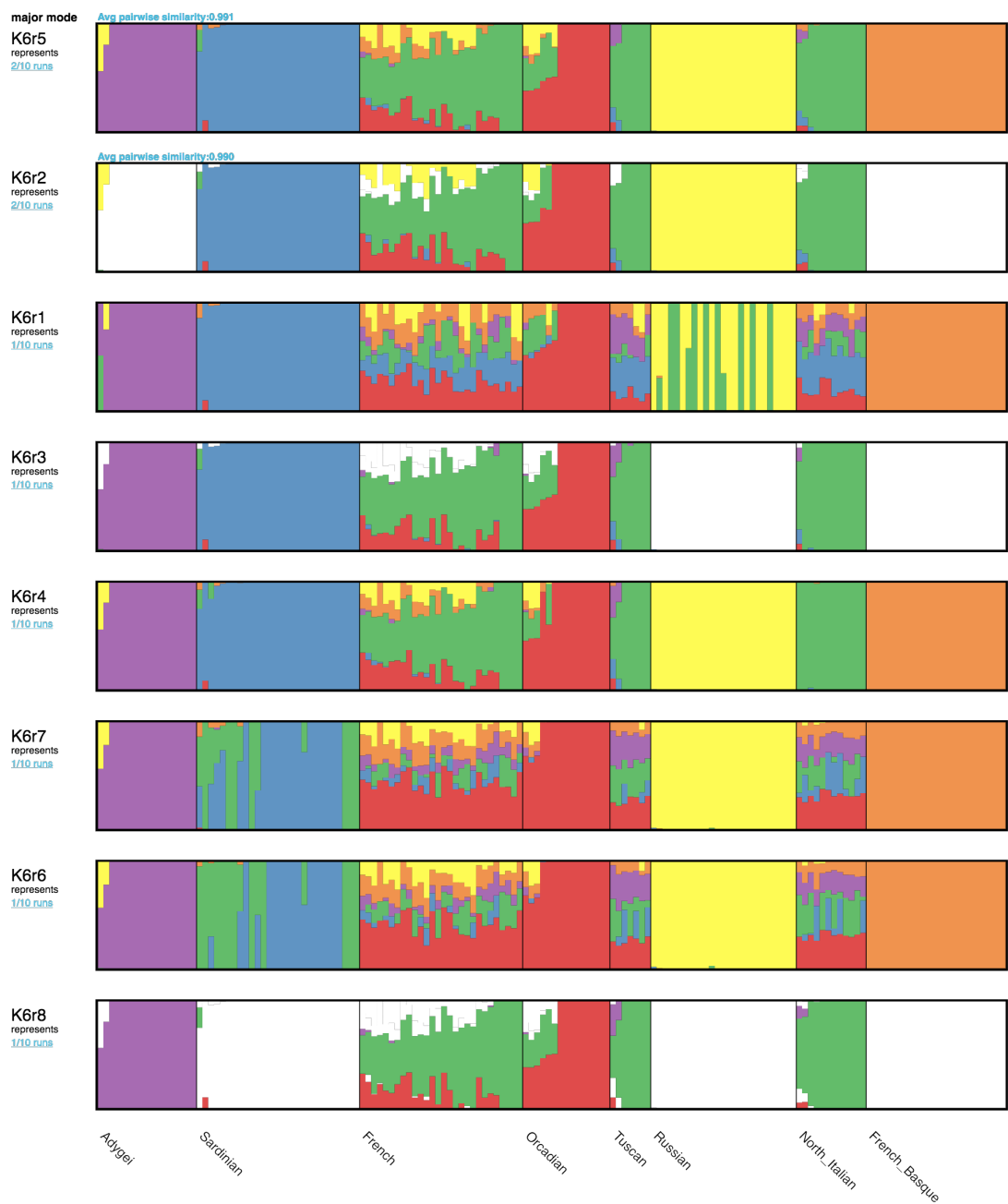
Figure 2: Visualizing multiple runs of ADMIXTURE

```
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
```

and build a Qmapfile for pong:

```
prefix=H938_Euro.LDprune
Klow=1
Khigh=12
echo '# Qfilemap' > out/$prefix.Qfilemap
for ((K=$Klow;K<=$Khigh;K++)); \
do
  awk -v K=$K -v prefix=$prefix 'BEGIN{ \
    printf("K%d\t%d\t%s.%d.Q\n",K,K,prefix,K)
  }' >> out/$prefix.Qfilemap;
done
```
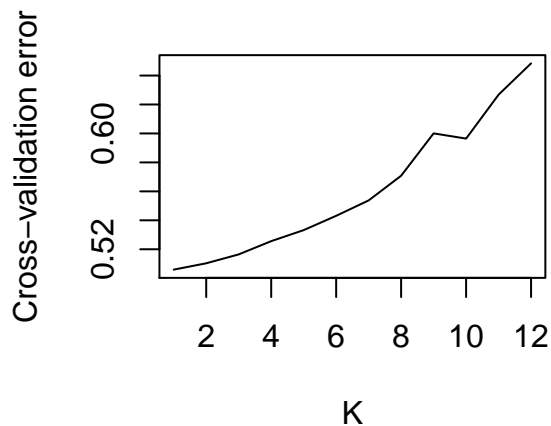
Now let's inspect the outputs. First let's make a plot of the cross-validation error as a function of K:

```
tbl = read.table("out/H938_Euro.LDprune.CV.txt")
plot(tbl$V1,tbl$V2,xlab="K",ylab="Cross-validation error",pch=16,type='l')
```


The cross-validation error suggests a single source population can model the data adequately and larger values of K lead to over-fitting. Let's now inspect the 'pong' outputs.

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

Here we see how as K increases through to $K = 6$, the Sardinian, Basque, Adygei, and Russian samples are modeled as descended from unique sources, at $K = 7$ and $K = 8$ structure within the Sardinian and Basque samples is revealed, though the Basque sub-structure is not stable at higher values of K. The values of $K = 9$ and above make increasingly finer scale divisions that are difficult to interpret.

Overall, it's worth noting that the visual inspection of the results suggests several "real" clusters in the data, though in this case 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.
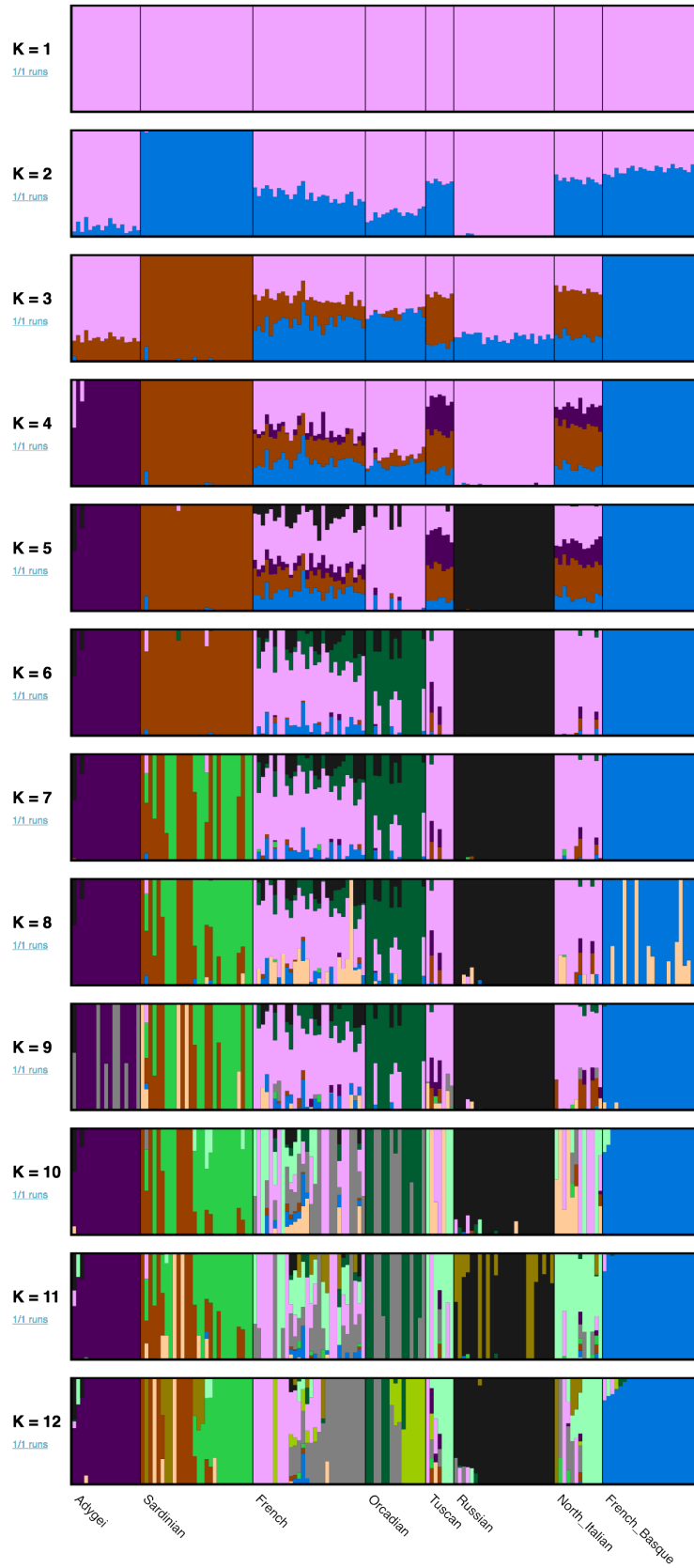
Figure 3: Example PONG output

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.

PCA with SMARTPCA

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.

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 abberviations for point characters and plotting median positions of each labelled group). To install the package use:

```r
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:

```r
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 a
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")
clst_unord=clst$V3[match(ids,clst$V2)]  # Order them to match the ids of PCA object
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)
```
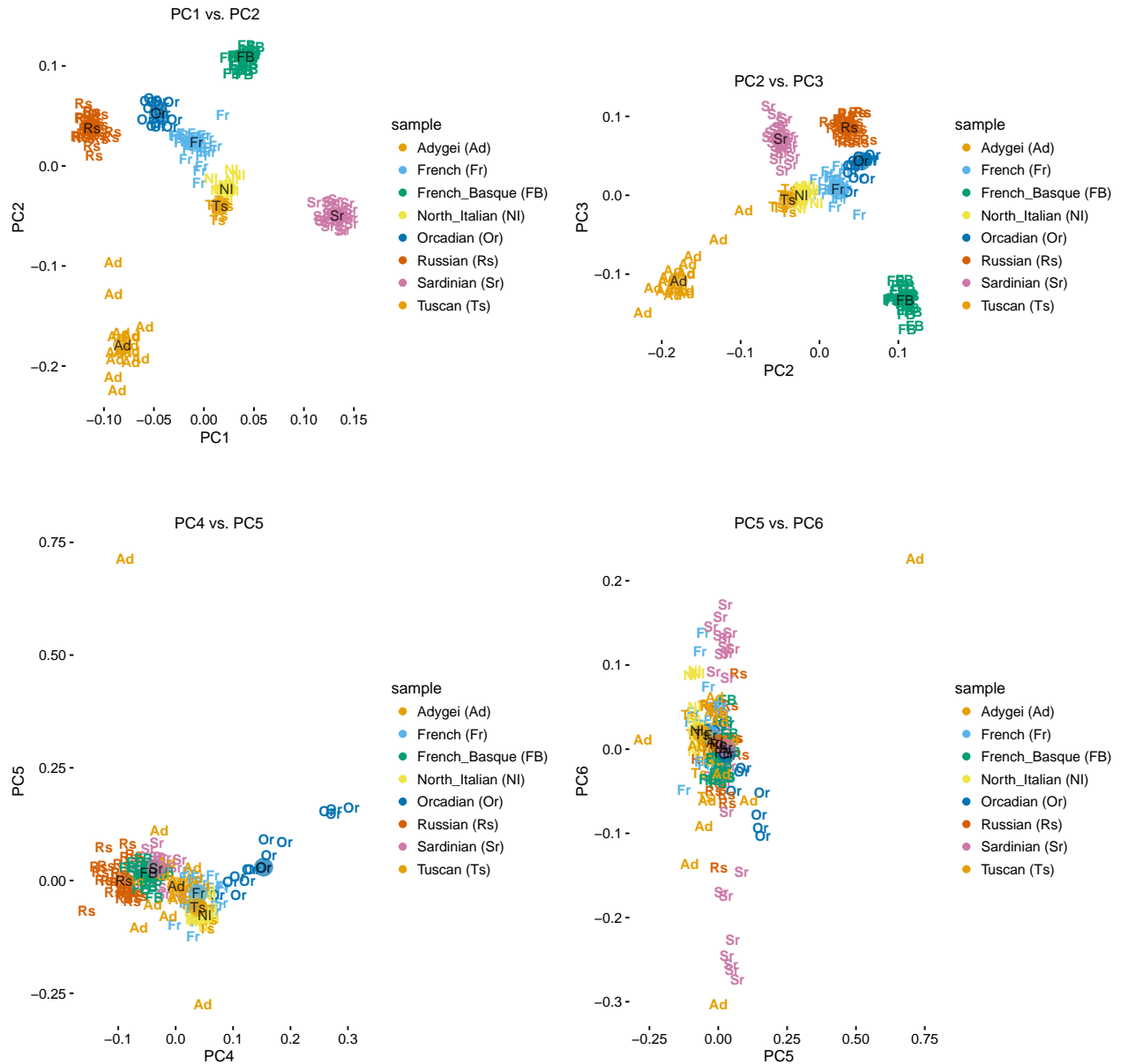
First one may notice several populations are separated with PC1 and PC2, with the more isolated populations being those that were most distinguished from the others by `ADMIXTURE`. PC4 distinguishes a subset of Orcadian individuals and PC5 distinguishes two Adygei indiviuals. PC6 corresponds to the cryptic 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))
```
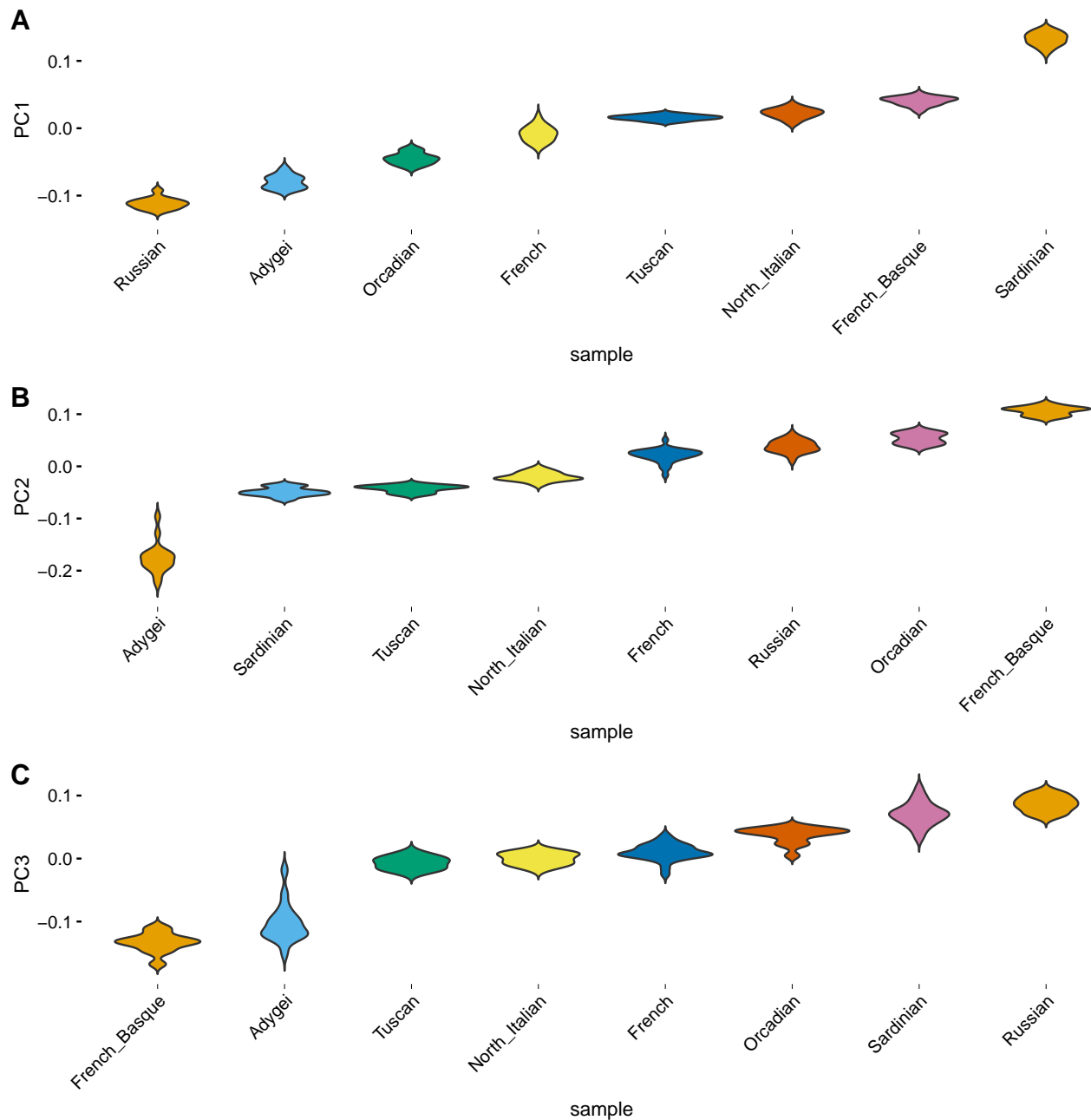
As mentioned above in the section on LD, it is useful to inspect the PC loadings to ensure that they broadly represent variation across the genome, rather than one or a small number of genomic regions (Duforet-Frebourg et al. 2016). SNPs that are selected in the same direction as genome-wide structure 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))
```


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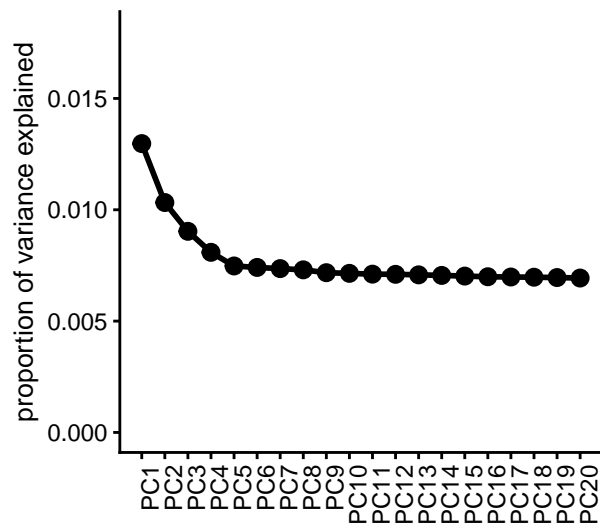
```
screeplot(hgdp,type='pve')  + ylim(0,0.018)+
      theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
      theme(axis.line = element_line(size = 1, linetype = "solid"))
```

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

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.

Discussion

Our protocol above is relatively straightfoward 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 perspecives 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 analyss (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 XX and XX).

References

Alexander DH, Lange K (2011) Enhancements to the admixture algorithm for individual ancestry estimation. BMC Bioinformatics 12:246. doi: 10.1186/1471-2105-12-246

Alexander DH, Novembre J, Lange K (2009) Fast model-based estimation of ancestry in unrelated individuals. Genome Res 19:1655–64. doi: 10.1101/gr.094052.109

Behr AA, Liu KZ, Liu-Fang G et al (2016) Pong: Fast analysis and visualization of latent clusters in population genetic data. Bioinformatics 32:2817–2823. doi: 10.1093/bioinformatics/btw327

Cann HM, Toma C de, Cazes L et al (2002) A human genome diversity cell line panel. Science 296:261–262. doi: 10.1126/science.296.5566.261b

Cavalli-Sforza LL, Menozzi P, Piazza A (1994) The history and geography of human genes. Princeton

University Press

Chang CC, Chow CC, Tellier LC et al (2015) Second-generation plink: Rising to the challenge of larger and richer datasets. GigaScience 4:s13742–015–0047–8. doi: 10.1186/s13742-015-0047-8

Duforet-Frebourg N, Luu K, Laval G et al (2016) Detecting genomic signatures of natural selection with principal component analysis: Application to the 1000 genomes data. Molecular Biology and Evolution 33:1082–1093. doi: 10.1093/molbev/msv334

Engelhardt BE, Stephens M (2010) Analysis of population structure: A unifying framework and novel methods based on sparse factor analysis. PLOS Genetics 6:1–12. doi: 10.1371/journal.pgen.1001117

Falush D, Dorp L van, Lawson D (2016) A tutorial on how (not) to over-interpret structure/admixture bar plots. bioRxiv. doi: 10.1101/066431

Falush D, Stephens M, Pritchard JK (2003) Inference of population structure using multilocus genotype data: Linked loci and correlated allele frequencies. Genetics 164:471–492

Holsinger K, Weir B (2009) Genetics in geographically structured populations: Defining, estimating and interpreting Fst. 10:639–50

Hubisz MJ, Falush D, Stephens M, Pritchard JK (2009) Inferring weak population structure with the assistance of sample group information. Molecular Ecology Resources 9:1322–1332. doi: 10.1111/j.1755-0998.2009.02591.x

Illumina I (2005) Illumina gencall data analysis software. https://www.illumina.com/Documents/products/technotes/technote_

Kopelman NM, Mayzel J, Jakobsson M et al (2015) Clumpak: A program for identifying clustering modes and packaging population structure inferences across K. Molecular Ecology Resources 15:1179–1191. doi: 10.1111/1755-0998.12387

Li JZ, Absher DM, Tang H et al (2008) Worldwide human relationships inferred from genome-wide patterns of variation. Science 319:1100–1104. doi: 10.1126/science.1153717

McVean G (2009) A genealogical interpretation of principal components analysis. PLoS Genetics 5:e1000686. doi: 10.1371/journal.pgen.1000686

Menozzi P, Piazza A, Cavalli-Sforza LL (1978) Synthetic maps of human gene frequencies in Europeans. Science 201:786–792

Novembre J (2016) Pritchard, Stephens, and Donnelly on population structure. Genetics 204:391–393. doi: 10.1534/genetics.116.195164

Novembre J (2014) Variations on a common structure: New algorithms for a valuable model. Genetics 197:809–11. doi: 10.1534/genetics.114.166264

Novembre J, Johnson T, Bryc K et al (2008) Genes mirror geography within europe. 456:274

Novembre J, Peter BM (2016) Recent advances in the study of fine-scale population structure in humans. Curr Opin Genet Dev 41:98–105. doi: 10.1016/j.gde.2016.08.007

Novembre J, Stephens M (2008) Interpreting principal component analyses of spatial population genetic variation. Nature Genetics 40:646–649. doi: 10.1038/ng.139

Patterson NJ, Price AL, Reich D (2006) Population structure and eigenanalysis. PLoS Genetics 2:2074–2093. doi: 10.1371/journal.pgen.0020190

Price AL, Patterson NJ, Plenge RM et al (2006) Principal components analysis corrects for stratification in genome-wide association studies. Nat Genet 38:904–9. doi: 10.1038/ng1847

Pritchard JK, Stephens M, Donnelly P (2000) Inference of population structure using multilocus genotype data. Genetics 155:945–959

Purcell S, Neale B, Todd-Brown K et al (2007) Plink: A tool set for whole-genome association and population-

based linkage analyses. 81:559–75

Raj A, Stephens M, Pritchard JK (2014) FastSTRUCTURE: Variational inference of population structure in large SNP data sets. Genetics 197:573–89. doi: 10.1534/genetics.114.164350

Rosenberg NA (2004) DISTRUCT: A program for the graphical display of population structure. Molecular Ecology Notes 4:137–138. doi: 10.1046/j.1471-8286.2003.00566.x

Rosenberg NA, Mahajan S, Ramachandran S et al (2005) Clines, clusters, and the effect of study design on the inference of human population structure. PLoS Genetics 1:e70. doi: https://doi.org/10.1371/journal.pgen.0010070

Tian C, Plenge RM, Ransom M et al (2008) Analysis and application of european genetic substructure using 300 K SNP information. PLoS Genet 4:e4. doi: 10.1371/journal.pgen.0040004

Williams R, Pourreza H, Wang Y et al (2017) PCAviz: Visualizing principal components analysis