Ancestry estimation based on reference samples of known ethnicities

Hannah Meyer

2020-03-11

Contents

Ancestry estimation

The identification of individuals of divergent ancestry can be achieved by combining the genotypes of the study population with genotypes of a reference dataset consisting of individuals from known ethnicities (for instance individuals from the Hapmap or 1000 genomes study [1]). Principal component analysis (PCA) on this combined genotype panel can then be used to detect population structure down to the level of the reference dataset (for Hapmap and 1000 Genomes, this is down to large-scale continental ancestry).

In the following, the workflow for combining a study dataset with the reference samples, conducting PCA and estimating ancestry is demonstrated. The study dataset consists of 200 individuals and 10,000 genetic markers and is provided with *plinkQC* in `file.path(find.package('plinkQC'),'extdata')`.

Workflow

Download reference data

A suitable reference dataset should be downloaded and if necessary, re-formated into PLINK format. Vignettes 'Processing HapMap III reference data for ancestry estimation' and 'Processing 1000Genomes reference data for ancestry estimation', show the download and processing of the HapMap phase III and 1000Genomes phase III dataset, respectively. In this example, we will use the HammapIII data as the reference dataset.

Set-up

We will first set up some bash variables and create directories needed; storing the names and directories of the reference and study will make it easy to use updated versions of the reference or new datasets in the

future. Is is also useful to keep the PLINK log-files for future reference. In order to keep the data directory tidy, we'll create a directory for the log files and move them to the log directory here after each analysis step.

```
qcdir='~/qcdir'
refdir='~/reference'
name='data'
refname='HapMapIII'

mkdir -r $qcdir/plink_log
```

Match study genotypes and reference data

In order to compute joint principal components of the reference and study population, we'll need to combine the two datasets. The plink –merge function enables this merge, but requires the variants in the datasets to be matching by chromosome, position and alleles. The following sections show how to extract the relevant data from the reference and study dataset and how to filter matching variants.

Prune study data

We will conduct principle component analysis on genetic variants that are pruned for variants in linkage disequlibrium (LD) with an $r^2 > 0.2$ in a 50kb window. The LD-pruned dataset is generated below, using plink –indep-pairwise to compute the LD-variants; additionally exclude range is used to remove genomic ranges of known high-LD structure. This file was originally provided by [6] and is available in `file.path(find.package('plinkQC'),'extdata','high-LD-regions.txt')`.

```
plink --bfile  $qcdir/$name \
      --exclude range  $refdir/$highld \
      --indep-pairwise 50 5 0.2 \
      --out $qcdir/$name
mv  $qcdir/$name.prune.log $qcdir/plink_log/$name.prune

plink --bfile  $qcdir/$name \
      --extract $qcdir/$name.prune.in \
      --make-bed \
      --out $qcdir/$name.pruned
mv  $qcdir/$name.pruned.log $qcdir/plink_log/$name.pruned
```

Filter reference data for the same SNP set as in study

We will use the list of pruned variants from the study sample to reduce the reference dataset to the size of the study samples:

```
plink --bfile  $refdir/$refname \
      --extract $qcdir/$name.prune.in \
      --make-bed \
      --out $qcdir/$refname.pruned
mv  $qcdir/$refname.pruned.log $qcdir/plink_log/$refname.pruned
```

Check and correct chromosome mismatch

The following section uses an awk-script to check that the variant IDs of the reference data have the same chromosome ID as the study data. For computing the genetic PC, the annotation is not important, however,

merging the files via PLINK will only work for variants with perfectly matching attributes. For simplicity, we update the pruned reference dataset. Note, that sex chromosomes are often encoded differently and might make the matching more difficult. Again, for simplicty and since not crucial to the final task, we will ignore XY-encoded sex chromosomes (via `sed -n '/^[XY]/!p'`).

```
awk 'BEGIN {OFS="\t"} FNR==NR {a[$2]=$1; next} \
    ($2 in a && a[$2] != $1)  {print a[$2],$2}' \
    $qcdir/$name.pruned.bim $qcdir/$refname.pruned.bim | \
    sed -n '/^[XY]/!p' > $qcdir/$refname.toUpdateChr

plink --bfile $qcdir/$refname.pruned \
      --update-chr $qcdir/$refname.toUpdateChr 1 2 \
      --make-bed \
      --out $qcdir/$refname.updateChr
mv $qcdir/$refname.updateChr.log $qcdir/plink_log/$refname.updateChr.log
```

Position mismatch

Similar to the chromosome matching, we use an awk-script to find variants with mis-matching chromosomal positions.

```
awk 'BEGIN {OFS="\t"} FNR==NR {a[$2]=$4; next} \
    ($2 in a && a[$2] != $4)  {print a[$2],$2}' \
    $qcdir/$name.pruned.bim $qcdir/$refname.pruned.bim > \
    $qcdir/${refname}.toUpdatePos
```

Possible allele flips

Unlike chromosomal and base-pair annotation, mismatching allele-annotations will not only prevent the plink –merge, but also mean that it is likely that actually a different genotype was measured. Initially, we can use the following awk-script to check if non-matching allele codes are a simple case of allele flips.

```
awk 'BEGIN {OFS="\t"} FNR==NR {a[$1$2$4]=$5$6; next} \
    ($1$2$4 in a && a[$1$2$4] != $5$6 && a[$1$2$4] != $6$5)  {print $2}' \
    $qcdir/$name.pruned.bim $qcdir/$refname.pruned.bim > \
    $qcdir/$refname.toFlip
```

Upate positions and flip alleles

We use plink to update the mismatching positions and possible allele-flips identified above.

```
plink --bfile $qcdir/$refname.updateChr \
      --update-map $qcdir/$refname.toUpdatePos 1 2 \
      --flip $qcdir/$refname.toFlip \
      --make-bed \
      --out $qcdir/$refname.flipped
mv $qcdir/$refname.flipped.log $qcdir/plink_log/$refname.flipped.log
```

Remove mismatches

Any alleles that do not match after allele flipping, are identified and removed from the reference dataset.

```
awk 'BEGIN {OFS="\t"} FNR==NR {a[$1$2$4]=$5$6; next} \
    ($1$2$4 in a && a[$1$2$4] != $5$6 && a[$1$2$4] != $6$5) {print $2}' \
    $qcdir/$name.pruned.bim $qcdir/$refname.flipped.bim > \
    $qcdir/$refname.mismatch

plink --bfile $qcdir/$refname.flipped \
      --exclude $qcdir/$refname.mismatch \
      --make-bed \
      --out $qcdir/$refname.clean
mv $qcdir/$refname.clean.log $qcdir/plink_log/$refname.clean.log
```

Merge study genotypes and reference data

The matching study and reference dataset can now be merged into a combined dataset with plink –bmerge. If all steps outlined above were conducted successfully, no mismatch errors should occur.

```
plink --bfile $qcdir/$name.pruned  \
      --bmerge $qcdir/$refname.clean.bed $qcdir/$refname.clean.bim \
         $qcdir/$refname.clean.fam  \
      --make-bed \
      --out $qcdir/$name.merge.$refname
mv $qcdir/$name.merge.$refname.log $qcdir/plink_log
```

PCA on the merged data

We can now run principal component analysis on the combined dataset using plink –pca which returns a .eigenvec file with the family and individual ID in columns 1 and 2, followed by the first 20 principal components.

```
plink --bfile $qcdir/$name.merge.$refname \
      --pca \
      --out $qcdir/$name.$reference
mv $qcdir/$name.$reference.log $qcdir/plink_log
```

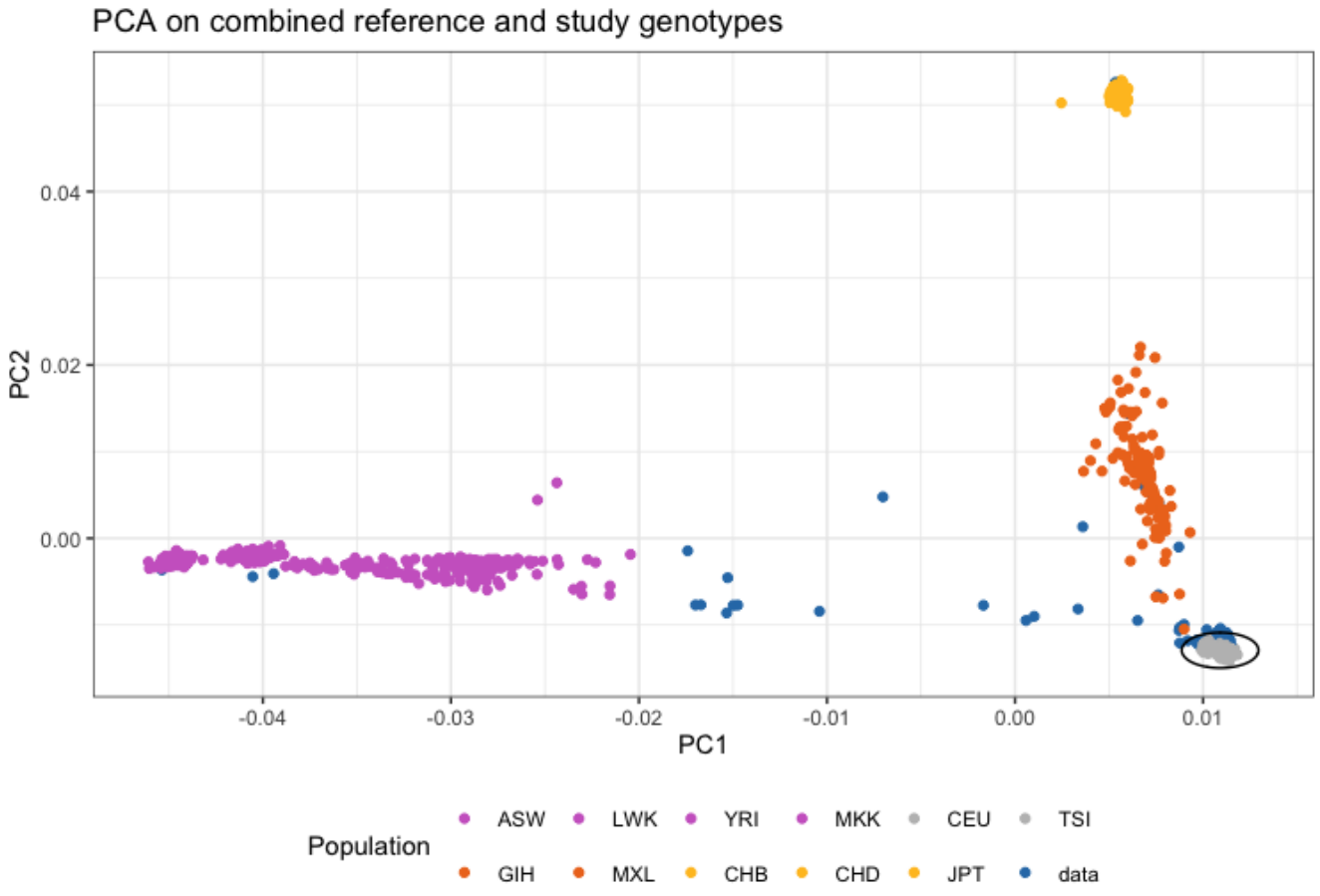
Check ancestry

We can use the .eigenvec file to estimate the ancestry of the study samples. Identifying individuals of divergent ancestry is implemented in `check_ancestry`. Currently, check ancestry only supports automatic selection of individuals of European descent. It uses principal components 1 and 2 to find the center of the known European reference samples. All study samples whose Euclidean distance from the centre falls outside the radius specified by the maximum Euclidean distance of the reference samples multiplied by the chosen `europeanTh` are considered non-European. `check_ancestry` shows the result of the ancestry analysis in a scatter plot of PC1 versus PC2 colour-coded for samples of the reference populations and the study population. From within R, run the following command to the ancestry check:

```
library(plinkQC)
indir <- system.file("extdata", package="plinkQC")
name <- 'data'
refname <- 'HapMapIII'
prefixMergedDataset <- paste(name, ".", refname, sep="")

exclude_ancestry <-
    evaluate_check_ancestry(indir=indir, name=name,
```

```
prefixMergedDataset=prefixMergedDataset,
refSamplesFile=paste(indir, "/HapMap_ID2Pop.txt",
                     sep=""),
refColorsFile=paste(indir, "/HapMap_PopColors.txt",
                    sep=""),
interactive=TRUE)
```



References

1. The International HapMap Consortium. A haplotype map of the human genome. Nature. 2005;437: 1299–320. doi:10.1038/nature04226

2. The International HapMap Consortium. A second generation human haplotype map of over 3.1 million SNPs. Nature. 2007;449: 851. doi:10.1038/nature06258

3. The International HapMap Consortium. Integrating common and rare genetic variation in diverse human populations. Nature. 2010;467. doi:10.1038/nature09298

4. 1000 Genomes Project Consortium. An integrated map of structural variation in 2,504 human genomes. Nature. 2015;526: 75–81. doi:10.1038/nature15394

5. 1000 Genomes Project Consortium. A global reference for human genetic variation. Nature. 2015;526: 75–81. doi:10.1038/nature15393

6. Anderson CA, Pettersson FH, Clarke GM, Cardon LR, Morris AP, Zondervan KT. Data quality control in genetic case-control association studies. Nature Protocols. 2010;5: 1564–73. doi:10.1038/nprot.2010.116