Preprocessing of Hispanics in 1000 Genomes data

Alejandro Ochoa and John D. Storey

2019-05-29

Introduction

This document explains the steps needed to obtain the following files that are available as part of this repository (https://github.com/StoreyLab/human-differentiation-manuscript):

```
data/hispanics.bed
data/hispanics.bim
data/hispanics.fam
data/hispanics-admix-panels.bed
data/hispanics-admix-panels.bim
data/hispanics-admix-panels.fam
data/hispanics-admix-panels.3.Q
```

All of this data originates from the 1000 Genomes project (TGP). However, extracting this data from the raw TGP data is sufficiently complicated to deserve the detailed documentation presented here.

Software requirements

These instructions assume a standard unix-like terminal, such as those of most modern Linux and MacOS.

In addition, you will need the following binaries (click for links):

- `plink2`, version 2.00 alpha.
- `admixture`, version 1.3.0.
- `zstd` package, which is available on standard Linux repositories.

These binaries will be assumed to be in your terminal's `PATH`.

Download and prepare the "plink2" version of TGP

Let's download the enormous TGP dataset in plink2 format, which is more compressed than the original data in VCF format. We rely extensivley on `plink2` to process the data, so it makes sense to start from `plink2`'s native format.

First let's switch to a `data` directory where all of these file are going. On a terminal, type:

```
# create directory if needed
mkdir data
cd data
```

You may manually download these "Merged dataset" files from the plink2-formatted raw TGP data:

- `phase3_corrected.psam` : Common sample information file, pedigree-corrected
- `all_phase3.pgen.zst` : Genotypes
- `all_phase3.pvar.zst` : Variant information

Alternatively, these files can be downloaded from a terminal using this command:

```
wget https://www.dropbox.com/s/yozrzsdrwqej63q/phase3_corrected.psam?dl=1
wget https://www.dropbox.com/s/afvvf1e15gqzsqo/all_phase3.pgen.zst?dl=1
wget https://www.dropbox.com/s/0nz9ey756xfocjm/all_phase3.pvar.zst?dl=1
```

However, the above links may change in the future, so check the plink2 website above if these fail.

Lastly, a little cleanup. Let’s rename one file for consistency:

```
mv phase3_corrected.psam all_phase3.psam
```

We must uncompress this file for `plink2` to read:

```
unzstd all_phase3.pgen.zst
```

Note that the compressed copy stays there (this is unlike default `gunzip` behavior). Also, `all_phase3.pvar.zst` can stay compressed.

Assign unique IDs to loci without them

The raw TGP data has many loci without IDs, which unfortunately get treated by `plink2` as having the same ID. This command sets missing IDs to `chr:pos` values, creating a new file `all_phase3_uniq.pvar.zst`:

```
plink2 \
    --pfile all_phase3 vzs \
    --set-missing-var-ids ’@:#’ \
    --make-just-pvar zs \
    --out all_phase3_uniq
```

Now we can replace the old file with the new one, preserving the original as `all_phase3_orig.pvar.zst`:

```
mv all_phase3.pvar.zst all_phase3_orig.pvar.zst
mv all_phase3_uniq.pvar.zst all_phase3.pvar.zst
# delete junk
rm all_phase3_uniq.log
```

Variant ascertainment and other filters

The next steps, performed jointly below, are to preserve loci that:

1. are autosomal, biallelic SNPs (model assumptions),
2. are variant in the Yoruba samples (code `YRI`), and
3. have unique IDs.

Filter 2 attempts to restrict the analysis to SNPs that were polymorphic in the last common ancestor of all humans (where `YRI` is used as a proxy for this ancestral population).

Filter 3 is only necessary due to bugs in the TGP data that were never fixed. In particular, many loci that are truly multiallelic are encoded in the raw data as multiple biallelic loci with the same `chr:pos` coordinates and same ID, with only the alternative alleles varying (we want only true biallelic loci in the analysis). This issue is documented here. In addition, some loci with different `chr:pos` coordinates also have the same ID.

We start by identifying all the sample IDs belonging to the `YRI` subpopulation (108 individuals):

```
# this command extracts the header (starts with #) and the sample rows of interest
grep -P ’\#|YRI’ all_phase3.psam > all_phase3_YRI.psam
```

Next we create a list of all loci in TGP that have unique IDs:

```
plink2 \
    --pfile all_phase3 vzs \
    --rm-dup exclude-all \
    --write-snplist zs \
    --out nodups

# cleanup
rm nodups.log
```

Lastly, we apply the filter from the previous step (the `nodups.snplist.zst` file), as well as all of our other desired filters, obtaining a locus list to filter all subsequent steps:

```
plink2 \
    --pfile all_phase3 vzs \
    --keep all_phase3_YRI.psam \   # YRI individuals only
    --extract nodups.snplist.zst \ # unique IDs only
    --autosome \
    --snps-only just-acgt \
    --max-alleles 2 \              # biallelic only
    --mac 1 \                      # polymorphic (min one count)
    --write-snplist zs \
    --out YRI

# remove these files after a successful run
rm YRI.log
rm all_phase3_YRI.psam
rm nodups.snplist.zst
```

This command is one way to count the number of loci in this filter file:

```
zstdcat YRI.snplist.zst|wc -l
# 20417484
```

Hispanics genotypes

We can finally obtain genotypes for the individuals of interest, with reasonable locus filters.

First create a list of individuals of interest. Like the earlier example, we want the header line and one line for every individual belonging to the subpopulation `AMR`:

```
grep -P ’\#|AMR’ all_phase3.psam > all_phase3_AMR.psam
```

This command creates the genotype files present in this repository. The output in this case is in BED (plink1) format, which is very widely supported:

```
plink2 \
    --pfile all_phase3 vzs \
    --keep all_phase3_AMR.psam \ # only the desired Hispanic individuals
    --extract YRI.snplist.zst \  # only consider loci in this list
    --mac 1 \                    # loci must also be polymorphic in Hispanics subset
    --make-bed \
    --out hispanics

# remove these files after a successful run
```

```
rm hispanics.log
rm all_phase3_AMR.psam
```

This simple command yields the dimensions of the genotype matrix:

```
wc -l hispanics.{bim,fam}
# 14145583 hispanics.bim # number of loci
#      347 hispanics.fam # number of individuals
```

Admixture analysis

Estimation of admixture proportions for the above Hispanic individuals is aided by reference panels of unadmixed individuals that approximate the source populations. Here we create this dataset, which is extended in the number of samples. However, as this inference scales poorly with the number of loci, and since rare variants are less informative, here we restrict loci by minor allele frequency.

First, create a list of individuals of interest. Keep the header, all individuals from subpopulation `AMR` (Hispanics), as well as individuals from `YRI` (Yoruba, representing Sub-Saharan African ancestry), `IBS` (Spanish, representing European ancestry), and `CHB` (Chinese, acting as a proxy for Native Americans, as no Native American populations are present in TGP):

```
grep -P ’\#|AMR|YRI|IBS|CHB’ all_phase3.psam > all_phase3_AMR+panels.psam
```

Create genotype file, as above except additionally requiring that the minor allele frequency (MAF) exceeds 5%:

```
plink2 \
    --pfile all_phase3 vzs \
    --keep all_phase3_AMR+panels.psam \
    --extract YRI.snplist.zst \
    --maf 0.05 \
    --make-bed \
    --out hispanics-admix-panels

# cleanup
rm hispanics-admix-panels.log
rm all_phase3_AMR+panels.psam

# data dimensions
wc -l hispanics-admix-panels.{bim,fam}
# 6216713 hispanics-admix-panels.bim
#     665 hispanics-admix-panels.fam
```

Now we can run the admixture software. NOTE: this can take 6 hours or more! The `-j6` option below specifies 6 threads to use (change as needed):

```
admixture -j6 hispanics-admix-panels.bed 3
```

From this run, only the output `hispanics-admix-panels.3.Q` (admixture proportions matrix) is provided in this repository.

Final file cleanup

These files are redundant (can be regenerated easily), but are needed until both sets of BED files are created.

```
# remove uncompressed genotype data (huge!)  Keep compressed version ending in *.zst
rm all_phase3.pgen
# variant filter
rm YRI.snplist.zst
```

Add subpopulation labels to FAM files

Although the original TGP PSAM file contains a column for the subpopulation label of every individual, the resulting FAM files in this pipeline lose that information. On the other hand, the "Family ID" column (first column) of the FAM file is trivial in these files ("0" for all individuals). It makes sense for convenience to store the subpopulation labels in this column instead.

To do this, we use the script `fam_add_pop_from_psam.R` provided in the `scripts/` directory of this repository. Note that this R script requires the packages `readr` (available on CRAN) and `genio`. The latter is available on GitHub and can be installed by running these commands in an R session:

```
install.packages("devtools") # if needed
library(devtools)
install_github("OchoaLab/genio", build_opts = c())
```

Assuming we remain in the data directory, these commands (back on the unix terminal) generate the new FAM files with the Family ID column containing the subpopulation labels:

```
Rscript ../scripts/fam_add_pop_from_psam.R \
    all_phase3.psam hispanics.fam hispanics.NEW.fam
Rscript ../scripts/fam_add_pop_from_psam.R \
    all_phase3.psam hispanics-admix-panels.fam hispanics-admix-panels.NEW.fam

# overwrite after visually inspecting for correctness
mv hispanics.NEW.fam hispanics.fam
mv hispanics-admix-panels.NEW.fam hispanics-admix-panels.fam
```