Population Genomics: background, tools and programming

Haplotype-based methods for demography

Garrett Hellenthal, University College London

For this practical, we will be applying the statistical software `GLOBETROTTER` and `RELATE` to simulated individuals in order to infer demographic history. In particular we will infer admixture events and past population sizes at various time points.

1 Inferring admixture: GLOBETROTTER

Here we will use the same dataset as in the "Clustering Algorithms" practical. I.e. we will use a subset of dataset explored in [1], again working only with chromosome 22 (6,812 SNPs) and the following populations:

Population	Country	Region	number of individuals
Balochi	Pakistan	Central South Asia	21
BantuKenya	Kenya	Africa	11
BantuSouthAfrica	South Africa	Africa	8
Burusho	Pakistan	Central South Asia	25
English	Britain	Europe	6
HanNchina	China	East Asia	10
Kalash	Pakistan	Central South Asia	23
Makrani	Pakistan	Central South Asia	22
Mandenka	Senegal	Africa	22
MbutiPygmy	Congo	Africa	13
Mongola	Mongolia	East Asia	10
NorthItalian	Italy	Europe	12
Orcadian	Britain	Europe	15
Pathan	Pakistan	Central South Asia	22
Sardinian	Italy	Europe	28
Tuscan	Italy	Europe	8
Total			256

The aim here is to see how well `GLOBETROTTER` can reconstruct an admixture event in the simulated "population" described in the "Clustering Algorithms" practical. This simulated group consists of 20 individuals descending from an admixture event occurring 30 generations ago, where 80% of the DNA was contributed from present-day Brahui individuals (from Pakistan, Central South Asia) and the remaining 20% from present-day Yoruba individuals (from Nigeria, Africa). To identify this admixture event, we will use the 16 populations above as surrogates to the admixing sources.

First we will apply `ChromoPainterv2` [2] to these data, but in a slightly different way than before in order to detect admixture. (This is described in detail in Section 8.2 of the `ChromoPainterv2` user manual, but we have already done many of these steps

in the earlier practical.) Navigate to your folder containing ChromoPainterv2 and the CHROMOPAINTER input files you used in the "Clustering Algorithms" practical, and type:

```
./ChromoPainterv2 -g example/BrahuiYorubaSimulationChrom22.haplotypes
-r example/BrahuiYorubaSimulationChrom22.recomrates
-t example/BrahuiYorubaSimulation.idfile.txt
-f BrahuiYorubaSimulation.poplistReduced.txt 0 0
-o example/BrahuiYorubaSimulationAdmixtureChrom22
```

This is very similar to the previous command used in "Clustering Algorithms", but changes the output name (specified by "-o") and, importantly, removes the "-a 0 0". Thus this uses the file BrahuiYorubaSimulation.poplistReduced.txt to determine which populations to paint and which to use as donors for this painting. Looking at BrahuiYorubaSimulation.poplistReduced.txt, we see that the population BrahuiYorubaSimulation is the only recipient population (R), while other populations are specified as donors (D). This means each BrahuiYorubaSimulation individual will be painted using all individuals from all other listed populations as donors. This is the same as yesterday's painting, except we do NOT allow the BrahuiYorubaSimulation individuals to be painted using themselves as donors. This is because we need to identify specific DNA segments inherited from admixing sources in order to generate the "coancestry curves" used to date admixture. When doing so, segments that "best match" to (i.e. are painted by) other BrahuiYorubaSimulation individuals would simply be discarded, because GLOBETROTTER does not allow a population to be an admixture source for itself, and so we would throw away information.

Next we will run GLOBETROTTER to infer admixture, using the painting samples output from ChromoPainterv2 (once it has finished!). Unzip and extract GLOBETROTTER:

```
tar -xzvf GLOBETROTTER.tar.gz
```

Next compile with:

```
R CMD SHLIB -o GLOBETROTTERCompanion.so GLOBETROTTERCompanion.c -lz
```

To run GLOBETROTTER, you have to specify three files:

(I) The parameter input file, which describes all settings, including which population to detect admixture in and which populations to use as ancestry surrogates. The file we will use here is BrahuiYorubaSimulationAdmixture.paramfile.txt.

(II) The painting samples file, which points to the ChromoPainterv2 output file(s) containing the painting samples of the putatively admixed population. The file we will use here is BrahuiYorubaSimulationAdmixture.samplesfile.txt.

(III) The recombination rate file, which points to the recombination rate file(s) used when running ChromoPainterv2. The file we will use here is BrahuiYorubaSimulationAdmixture.recomfile.txt.

File (II) will point to the `example/BrahuiYorubaSimulationAdmixtureChrom22.samples.out` file we just made using `ChromoPainterv2`, which contains 10 sampled paintings for each haplotype of each individual from our simulated admixture population. File (III) will point to the `example/BrahuiYorubaSimulationChrom22.recomrates` we used when generating these paintings.

For file (I), we have specified the input file `BrahuiYorubaSimulation.idfile.txt` we used when running `ChromoPainterv2` above, as well as the output filenames (`save.file.XX`). We have also specified the painting we made during the "Clustering Algorithms" practical, which painted the target population and each of the surrogate populations using the same set of donors. (In this case the donor populations – `copyvector.popnames` – are the same as the surrogate populations – `surrogate.popnames`.) This information is used in the linear model we discussed in the lecture. The other parameters specify ways to run `GLOBETROTTER`. Most of these will likely not be changed, except for the first 3, which specify whether to infer dates and admixture proportions (`prop.ind`) and/or bootstrap re-sample to determine uncertainty in date estimation (`bootstrap.date.ind`), and/or whether to standardize estimates by a "NULL" individual to help eliminate spurious signals of admixture (`null.ind`). (As long as you are detecting admixture in $\geq$3 individuals, I highly recommend doing the latter, as we will do below.)

To run `GLOBETROTTER` under these settings type:

```
R < GLOBETROTTER.R BrahuiYorubaSimulationAdmixture.paramfile.txt
BrahuiYorubaSimulationAdmixture.samplesfile.txt
BrahuiYorubaSimulationAdmixture.recomfile.txt --no-save > output.out
```

It will take a couple minutes to run. You can follow progress by typing:

```
pic output.out
```

Once finished, the following output files will be produced, each in the `example/` directory:

(I) `BrahuiYorubaSimulationAdmixed.globetrotter.main.txt` – this gives the results, including `GLOBETROTTER`'s "best-guess" conclusion regarding admixture and the inferred admixture dates and proportions

(II) `BrahuiYorubaSimulationAdmixed.globetrotter.main.pdf` – this gives you "coancestry curves" for every combination of surrogate populations that are inferred to have contributed >0.1% ancestry to the target population

(III) `BrahuiYorubaSimulationAdmixed.globetrotter.main_curves.txt` – this gives all of the raw data used to produce the curves in (II), in case you want to make your own plots

We will now run `GLOBETROTTER` again, but incorporating a "NULL" individual that attempts to account for any signal in LD decay that is *not* explained by genuine admixture. To do so, in `BrahuiYorubaSimulationAdmixture.paramfile.txt` change `null.ind` to "1" and change `save.file.main` to "example/BrahuiYorubaSimulationAdmixed.globetrotter.mainNULL". Then run `GLOBETROTTER`

again, exactly as above. This will make each of the output files as above, but with a `xx.mainNULLxx` in the filename.

Now answer the following questions:

1. From the `GLOBETROTTER` user manual, what do the different measures in `BrahuiYorubaSimulationAdmixed.globetrotter.main.txt` tell you?

2. What is `GLOBTROTTER`'s conclusion about admixture in this application?The inferred sources and dates give a "best-guess" conclusion of "one-date", which (according to the manual) means a simple admixture event between two sources at one time (which is correct here). Looking at the "1-DATE FIT EVIDENCE", we see the inferred date is about 26 generations (pretty close to the truth of 30). The other interesting lines, given we infer a simple admixture event, is "1-DATE FIT SOURCES, PC1:" which tells you the inferred genetic make-up of each of the two admixing sources. This concludes that 26% of the DNA comes from one source that is related to "BantuSouthAfrica" and "Mandenka" (i.e. African populations), while the remaining 74% comes from a source that looks very much like the "Balochi" from Pakistan. So a fairly accurate representation of the 20%-80% Africa-CentralSouthAsia mix we simulated.

3. How do you interpret the coancestry curves in `BrahuiYorubaSimulationAdmixed.globetrotter.main.pdf`? Do the results from `BrahuiYorubaSimulationAdmixed.globetrotter.main.txt` make sense in light of these coancestry curves? The populations with increasing curves always consist of an African ppopulation versus a non-African population. Meanwhile curves with two African populations or two non-African populations are all decreasing. This allows GLOBETROTTER to infer that the two groups that intermixed were African and non-African (mainly Balochi-like), respectively.

4. Do results change when incorporating the "NULL" individual? Results are very similar here, which is typically the case unless the target population has experienced a severe bottleneck. In such a case, inference using the "NULL" individual is more reliable, as it is designed to eliminate any patterns in the decay curves that are not due to admixture (e.g. it removes effects that are instead due to a bottleneck in the target population).

If more time, change `bootstrap.date.ind` to "1", which will provide 20 bootstrap resample estimates of the date. Then try changing the surrogates (i.e. remove some populations listed in `surrogate.popnames`). How do results change?

2 Building trees using sequencing data: RELATE

Now we will use `RELATE` [3] to infer the demographic history of populations. First we will simulate data using the program `ms` [4]. Then we will use the scripts and instructions available at https://myersgroup.github.io/relate/index.html by Leo Speidel.

First unpack `ms` and `RELATE`:

```
tar -xzvf relate_v1.0.8_x86_64_static.tgz
tar -xzvf ms.tar.gz
```

Then change to the `msdir/` directory to compile `ms`:

```
gcc -O3 -o ms ms.c streec.c rand1.c -lm
```

Navigate out of `msdir/`. We will first use `ms` to simulate some data:

```
msdir/./ms 200 1 -t 30000 -r 6000 30000000 -eN 0.01 0.1 -eN 0.06 1 -eN
0.2 0.5 -eN 1 1 -eN 2 2 -p 8 > sim1.ms
```

This specifies that you will run one simulation consisting of 200 haplotypes (i.e. 100 individuals) simulated over a 30Mb region with uniform mutation and recombination (population-scaled) rates of 1000/Mb and 200/Mb, respectively. Everything is scaled in units of $4N0$ generations, where $N0$ is the **diploid** population size at present. We'll assume $N0 = 10000$ here, so that the mutation rate per generation is $1000/(4N0) = 0.025/\text{Mb} = 2.5e^{-8}/\text{bp}$. Similarly the recombination rate per generation is $0.5e^{-8}$. The "`-eN t x`" parameters specify the population demography of this population, with `t` the time (again in units of $4N0$ generations) at which the population's size becomes `x` times that of the present-day population size. Here time goes from present to past, so that "`-eN 0.01 0.1`" means that the population shrinks to 10% of its present-day size at `0.01x4xN0` generations ago (e.g. 400 generations ago if *N0*=10K), continuing at this size until it hits the next time point specified by `-eN` (which is `0.06` in this case).

Next we will convert this `ms` output to `RELATE` input format (which is in SHAPEIT-style; see https://myersgroup.github.io/relate/input_data.html) using the perl program `MStoShapeItOutputTUTORIAL.pl`:

```
perl MStoShapeItOutputTUTORIAL.pl
```

Importantly, at each SNP "`0`" and "`1`" should refer to the ancestral and derived alleles, respectively (though `RELATE` does try to correct inconsistencies), which is the case for `ms`-simulated data.

Once finished, create a folder called (e.g.) `temp` and navigate to this folder. (As `RELATE` makes different temporary files in each directory, you should always run each new analysis in a NEW directory.) To run `RELATE` on our example, type:

```
../relate_v1.0.8_x86_64_static/bin/Relate --mode All -m 2.5e-8 -N 20000
--haps ../sim1.haps --sample ../sim1.sample --map ../sim1.map -o
sim1.relateout
```

(Note also that your '-o' (i.e. output) file MUST be in the current directory.) Insofar as I can tell, --mode has relevance if you want to e.g. parallelize. The other parameters specify our mutation rate of $2.5e^{-8}$/generation and effective (**haploid**) population size of 20000. This will output two files: sim1.relateout.anc and sim1.relateout.mut. The former contains the tree-building information, while the latter lists the branch locations and lengths (in generations) for which each mutation (SNP) falls.

RELATE provides helpful scripts to e.g. plot the tree at a particular base position:

```
../relate_v1.0.8_x86_64_static/scripts/TreeView/TreeView.sh --haps ../sim1.haps
--sample ../sim1.sample --anc sim1.relateout.anc --mut sim1.relateout.mut --poplabels
../sim1.poplabels --bp_of_interest 30000 --years_per_gen 25 -o sim1.relateout
```

Here sim1.poplabels is a RELATE-specific input file that lists the population label for each individual in the sim1.sample file. The above runs an R script treeview.R (which you can edit) that produces the output file sim1.relateout.pdf that plots the inferred tree, plus mutations, at position bp_of_interest. Time (in years) is given on the right (note that we have specified 25 years per generation), and the circles are the mutations in the region ("flipped" refers to whether the algorithm decided to flip ancestral/derived status of the allele from what is specified in the input file).

You can also infer (and produce plots for) the change in population size over time:

```
../relate_v1.0.8_x86_64_static/scripts/EstimatePopulationSize/EstimatePopulationSize.sh
-i sim1.relateout --poplabels ../sim1.poplabels -m 2.5e-8 --years_per_gen 25
--num_iter 1 -o sim1.relateout_popsize
```

To make RELATE run quicker, we have run only a single iteration (num_iter), though the authors recommend 5 iterations. The file sim1.relateout_popsize.pdf is created, which shows how the effective population size (y-axis) varies with time in years (x-axis).

Willy Rodriguez has applied PSMC [5] to a similar simulation, though consisting of only a single individual simulated for 100 independent 30Mb regions, in contrast to our simulation here consisting of 100 individuals simulated for a single 30Mb region. You can follow how he did so (and get the relevant scripts to run PSMC on this simulation if you like) at http://willyrv.github.io/tutorials/bioinformatics/ms-psmc.html. At the bottom of this webpage, you can see the results of PSMC applied to these simulated data.
Now answer the following questions:

1. How do you interpret the plots showing population sizes over time for PSMC and RELATE? Time is on the x-axis (in years) and population size is on the y-axis.

2. How does the accuracy of PSMC compare to that of RELATE? Both PSMC and RELATE do well for most of the simulation, with PSMC doing more poorly at recent dates, because two haplotypes (which is the only thing that PSMC considers) typically take a long time to coalesce. Thus there is little information on coalescent events occuring in recent times. RELATE gets around this by considering the coalesence of multiple haplotypes rather than just 2.

3. Where does the inference go wrong?

If time, try another `ms` simulation of two populations that split 4000 generations ago (100Kya, assuming 25 years/gen), after which Pop2 had a bottleneck (to the present-day) that reduced its size to 20% that of Pop1 (which again we'll assume is $N0 = 10K$ diploids):

```
msdir/./ms 200 1 -t 30000 -r 6000 30000000 -I 2 100 100 0 -ej 0.1 2 1 -en
0 2 0.2 -p 8 > sim2.ms
```

Here "`-I 2 100 100 0`" specifies that we want 2 populations with no (`0`) migration between them, sampling 100 haplotypes from each population. The "`-ej 0.1 2 1`" specifies that Pop2 merges into Pop1 at time `0.1` (in units of $4N0$ generations), and "`-en 0 2 0.2`" specifies that Pop2 has size $0.2 \times N0$ from time 0 (present-day) until this merge. Be sure to make a new folder (e.g. `temp2`) and run your `RELATE` analyses from this new folder.

1. What do you see in the `sim2.relateout.pdf` plot? You can see some evidence of the two populations splitting.

2. What do you see in the `sim2.relateout_popsize.pdf` plot? Populations 1 and 2 stay separate until the split time we simulated, after which they merge to show the same population size history. From the split until present-day, `RELATE` does a good job inferring the correct population sizes for each population. The "Pop1-Pop2" line increases rapdily during the time from present prior to the split – this is because the effective population size is equal to 1 divided by the coalescent rate, and the coalescent rate between individuals from different populations is very low until the two populations merge.

References

[1] G. Hellenthal, G.B.J. Busby, G. Band, J.F. Wilson, C. Capelli, D. Falush, and S. Myers. A genetic atlas of human admixture history. *Science*, 343:747–751, 2014.

[2] D.J. Lawson, G. Hellenthal, S. Myers, and D. Falush. Inference of population structure using dense haplotype data. *PLoS Genet*, 8(1):e1002453, 2012.

[3] L. Speidel, M. Forest, S. Shi, and S. Myers. A method for genome-wide genealogy estimation for thousands of samples. *bioRxiv*, page dx.doi.org/10.1101/550558, 2019.

[4] R.R. Hudson. Generating samples under a Wright-Fisher neutral model of genetic variation. *Bioinformatics*, 18(2):337–8, 2002.

[5] H. Li and R. Durbin. Inference of human population history from individual whole-genome sequences. *Nature*, 475:493–496, 2011.