User Manual for TreeMix v1.1

Joseph K. Pickrell, Jonathan K. Pritchard

October 1, 2012

Contents

1 Introduction

TreeMix is a program for the inference of patterns of population splitting and mixing from genome-wide allele frequency data. If given a set of allele frequencies from a number of populations, it will return the maximum likelihood tree for the set of populations, and optionally attempt to infer a number of admixture events.

2 Installation

TreeMix should run on any Unix or Unix-like (e.g., Linux or Mac OS X) system. It requires the GNU Scientific Library (`http://www.gnu.org/s/gsl/`), and the Boost Graph Library (`http://www.boost.org/`; you need version 1.42 of Boost or greater). Be sure these libraries are installed, and after downloading the source code, run the standard installation steps:

```
>tar -xvf treemix-1.0.tar.gz
>cd treemix-1.0
>./configure
>make
>make install
```

3 Input file format

3.1 SNP data

By default, TreeMix assumes biallelic sites. The input file is a gzipped file that consists of a header with a space-delimited list of the names of populations, followed by lines containing the allele counts at each SNP. It is assumed that the order of the SNPs in the file is the order of the SNPs in the genome. The line is space delimited between populations, and the two allele within the population are comma-delimited. For example:

```
pop1 pop2 pop3 pop4
5,1 1,1 4,0 0,4
3,3 0,2 2,2 0,4
1,5 0,2 2,2 1,3
```

3.2 Microsatellite data (`-micro`)

As of version 1.1, *TreeMix* can take microsatellite data as input. **If the data is of microsatellites, every command must include the `-micro` flag**. In this case the entries for each population

are the mean length, variance in length, and number of haplotypes at each microsatellite in each population, again comma-delimited. The lengths can be either physical lengths or numbers of repeats. For example:

```
pop1 pop2 pop3
240.8,47.6,18 150.6,80.1,10 270.1,34.4,30
210.0,90.2,18 130.3,45.4,10 200.9,19.9,30
56.6,10.1,18 40.4,15.5,10 79.5,31.3,30
```

3.3 Incorporating known migration (-cor_mig)

As of version 1.1, known migration events can be included in *TreeMix*. See below for the exact commands. To input known events, prepare a file with each event on a line, where the first column is the source population, the second column is the admixed population, and the third column is the admixture proportion. For example, to tell *TreeMix* that you expect the Mozabite to have 20% of their ancestry from the Yoruba, the file would be the following single line:

```
Yoruba Mozabite 0.2
```

4 Options

4.1 Build the ML tree

The default behavior of TreeMIx is to build the maximum likelihood tree of the populations in the input file under the assumption that all sites are independent. To do this, run:

```
>treemix -i input_file.gz -o out_stem
```

4.2 Choose the position of the root (-root)

Before adding migration edges to a tree, it is important to set the position of the root. To build the tree and set the position of the root, if the name of the outgroup population is Outgroup, run:

```
>treemix -i input_file.gz -root Outgroup -o out_stem
```

4.3 Group together SNPs to account for linkage disequilibrium (-k)

To account for the fact that nearby SNPs are not independent, group them together in windows of size n SNPs by using the `-k` flag. The order of SNPs in the input file is assumed to be their order

in the genome. We recommend using a value of n that far exceeds the known extent of LD in the organism in question (this will depend, of course, on the SNP density). For example, the build the ML tree using blocks of 1000 SNPs, run:

```
>treemix -i input_file.gz -k 1000 -o out_stem
```

4.4 Build the ML graph with migration (-m)

If you wish to allow for a number of migration events in the tree, use the `-m` flag, followed by the number of allowed migration events. The following command will build the ML tree and then add two migration events:

```
>treemix -i input_file.gz -m 2 -o out_stem
```

4.5 Input a previously generated tree/graph (-g)

There are two ways to input a previously generated tree/graph. The most simple is to input from *TreeMix* format using the `-g` flag, which take a file of vertices and a file of edges as input. For example:

```
>treemix -i input_file.gz -m 2 -g out_stem.vertices.gz out_stem.edges.gz -o out_stem2
```

4.6 Generate a bootstrap replicate (-bootstrap)

For judging the confidence in a given tree topology, it is often of interest to generate a bootstrap replicate. Bootstrapping is done over blocks of contiguous SNPs. The following example shows how to generate a single bootstrap replicate by resampling blocks of 500 SNPs:

```
>treemix -i input_file.gz -bootstrap -k 500 -o replicate1
```

4.7 Incorporate known migration events (-cor_mig and -climb)

As of version 1.1, *TreeMix* allows incorporation of known migration events. To include these, prepare a file describing the known events as outlined in the section on input files. To include these events, use the `-cor_mig` option. This will then build the tree accounting for those events. Additionally, we recommend performing a round of hill-climbing to optimize the exact migration edges using the `-climb` flag. To build a tree incorporating known migration events in a file called

known_events, use the following command:

```
>treemix -i input_file.gz -cor_mig known_events -climb -k 500 -o corrected_tree
```

4.8 Turn off sample size correction (-noss)

By default, when *TreeMix* calculated the covariance matrix among populations, it includes a correction for sample size effects. In some cases (e.g., with single individuals in a population), this can lead to overcorrection. If you are getting many branches with length zero, this may be the problem. To turn off the sample size correction use the `-noss` flag. For example:

```
>treemix -i input_file.gz -k 500 -noss -o uncorrected_tree
```

5 Output files

TreeMix will output a number of files. If you have used the `-o` flag to designate the output stem `outstem`, these will be:

1. `outstem.cov.gz`. The covariance matrix ($\hat{\mathbf{W}}$ in Pickrell and Pritchard [2012]) between populations estimated from the data
2. `outstem.covse.gz`. The standard errors for each entry in the covariance matrix
3. `outstem.modelcov.gz`. The fitted covariance ($\mathbf{W}$ in Pickrell and Pritchard [2012]) according to the model
4. `outstem.treeout.gz`. The fitted tree model and migration events
5. `outstem.vertices.gz`. This and the following file (`outstem.edges.gz`) contain the internal structure of the inferred graph. Modifying these files will cause issues if you try to read the graph back in, so we recommend against this.
6. `outstem.edges.gz`.

The tree inferred from the data is in `outstem.treeout.gz`. The first line of this file is the Newick format ML tree, and the remaining lines contain the migration edges. The first column for these lines is the weight on the edge, followed (optionally) by the jackknife estimate of the weight, the jackknife estimate of the standard error, and the p-values. Then come the subtree below the origin of the migration edge, and the subtree below the destination of the migration edge.

6 Visualization

6.1 Graph visualization

To visualize the graph, use the R script `plotting_funcs.R`, which you can find in the folder `src/` in the tarball of source code. The function is called `plot_tree`. To use it, from within R, run:

```
>source("src/plotting_funcs.R")
>plot_tree("outstem")
```

This will produce a figure like that displayed in Figure 1A. If there are migration edges in the tree, they will be colored according to their weight, as in Figure 2.

6.2 Residual visualization

We have found it useful to visualize the residuals from the fit of the model to the data. This helps identify populations that are not well-modeled (due to, for example, additional migration). To view the residuals, run (again within R), and after loading the plotting script:

```
>plot_resid("outstem", "poporder")
```

The file `"poporder"` is simply a list of the names of the populations in the order you would like them to be plotted. This will produce a figure like the one shown in Figure 1B.

7 Additional programs

7.1 `threepop`

The three-population test was introduced by Reich et al. [2009] as a test for treeness in three population trees. These tests are of the form $f_3(A; B, C)$, where a significantly negative value of the f_3 statistic implies that population A is admixed. See Reich et al. [2009] for details. To run this test using the implementation distributed with TreeMix, the input is the standard TreeMix input file. To run all possible f_3 statistics and get standard errors in blocks of 500 SNPs, do:

```
>threepop -i input.gz -k 500
```

This will write the f_3 statistics to standard output. The output is four columns. These are the populations used to calculate the f_3 statistic, the f_3 statistic, the standard error in the f_3 statistic, and the Z-score. For example, the following line shows the f_3 statistic where Colombian is population A, Han is population B, and Sardinian is population C:

```
Colombian;Han,Sardinian 0.0194104 0.00110139 17.6235
```

7.2 `fourpop`

The four-population test was introduced by Keinan et al. [2007] as a test for treeness in four population trees. These tests are of the form $f_4(A, B; C, D)$, where a significantly non-zero value indicates gene flow in the tree. See Reich et al. [2009] for details. To run this test using the implementation distributed with TreeMix, the input is the standard TreeMix input file. To run all possible f_4 statistics and get standard errors in blocks of 500 SNPs, do:

```
>fourpop -i input.gz -k 500
```

This will write the f_4 statistics to standard output. The output is four columns. These are the populations used to calculate the f_4 statistic, the f_4 statistic, the standard error in the f_4 statistic, and the Z-score. For example, the following line shows the f_4 statistic where Han is population A, Colombian is population B, Sardinian is population C, and Dai is population D:

```
Han,Colombian;Sardinian,Dai -0.00773721 0.000730214 -10.5958
```

8 Citing *TreeMix*

For the basic *TreeMix* functions of building trees and adding migration, please cite:
`Pickrell and Pritchard (2012). Inference of population splits and mixtures from genome-wide allele frequency data. PLoS Genetics. In Press.`

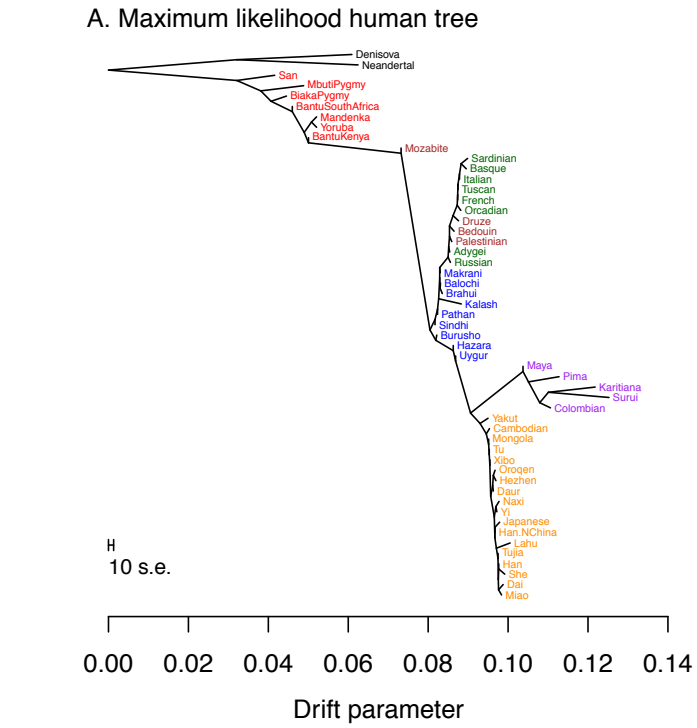
For the model incorporating known admixture (`-cor_mig`), please cite:
`Pickrell et al. (2012). The genetic prehistory of southern Africa. Nature Communications. In Press.`

References

Keinan, A., Mullikin, J. C., Patterson, N., and Reich, D., 2007. Measurement of the human allele frequency spectrum demonstrates greater genetic drift in East Asians than in Europeans. *Nat Genet*, **39**(10):1251–5.

Pickrell, J. K. and Pritchard, J. K., 2012. Inference of population splits and mixtures from genome-wide allele frequency data. *PLoS Genetics*, **In Press**.

Reich, D., Thangaraj, K., Patterson, N., Price, A. L., and Singh, L., 2009. Reconstructing Indian population history. *Nature*, **461**(7263):489–94.



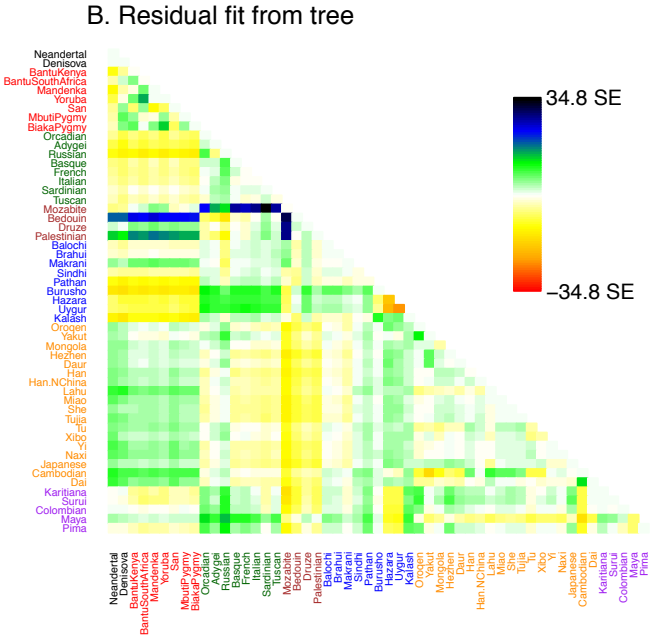


Figure 1: ML tree of 53 human populations

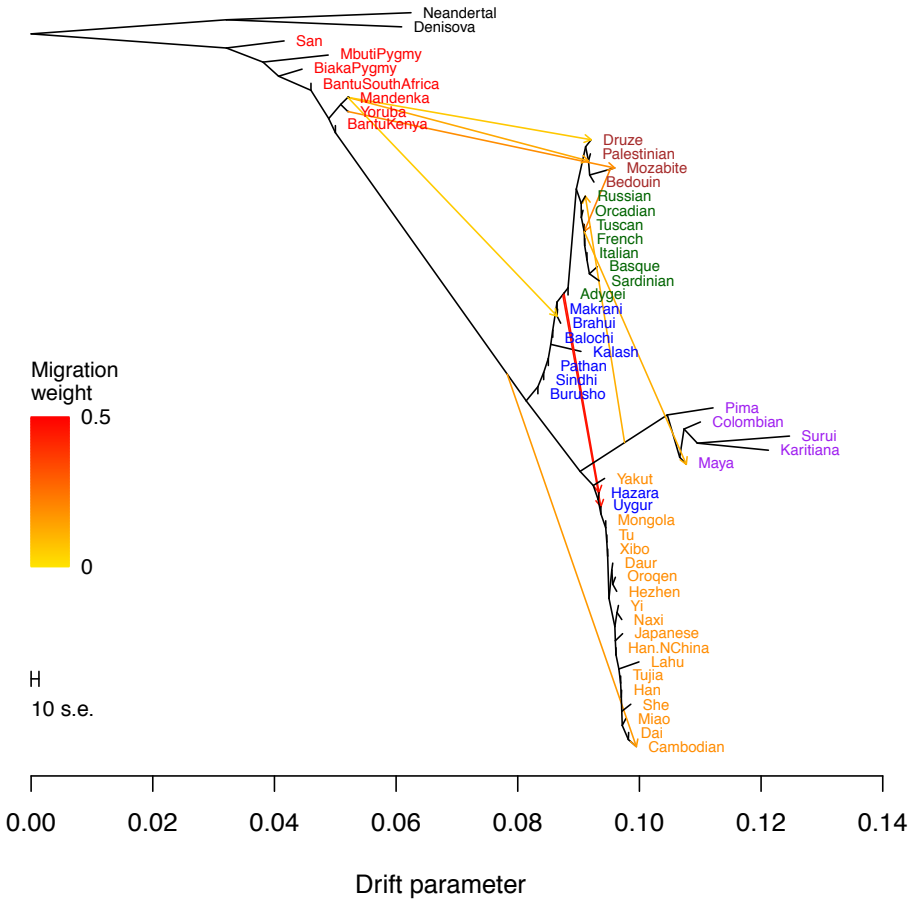


Figure 2: ML tree of 53 human populations with inferred migration edges