User Manual for TreeMix v1.0

Joseph K. Pickrell, Jonathan K. Pritchard

February 14, 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

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 pop3
5,1 1,1 4,0 0,4
3,3 0,2 2,2 0,4
1,5 0,2 2,2 1,3
```

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
```

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 et al.) 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 et al.) 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

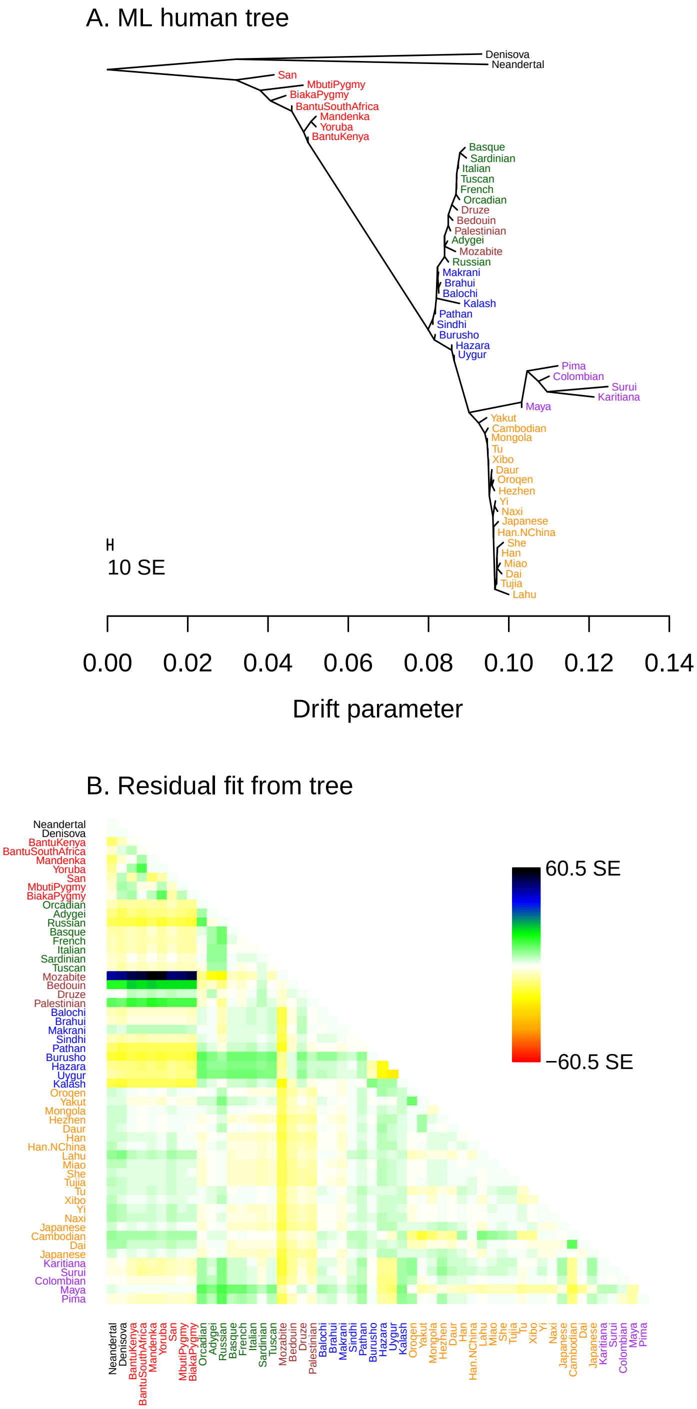
To visualize the graph, use the R script `plotting_funcs.R`, which you can find in the folder `src/` in the tarball of source code. The function is called `plot_tree`. To use it, from within R, run:

```
>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:



A. ML human tree

B. Residual fit from tree

Figure 1: ML tree of 53 human populations

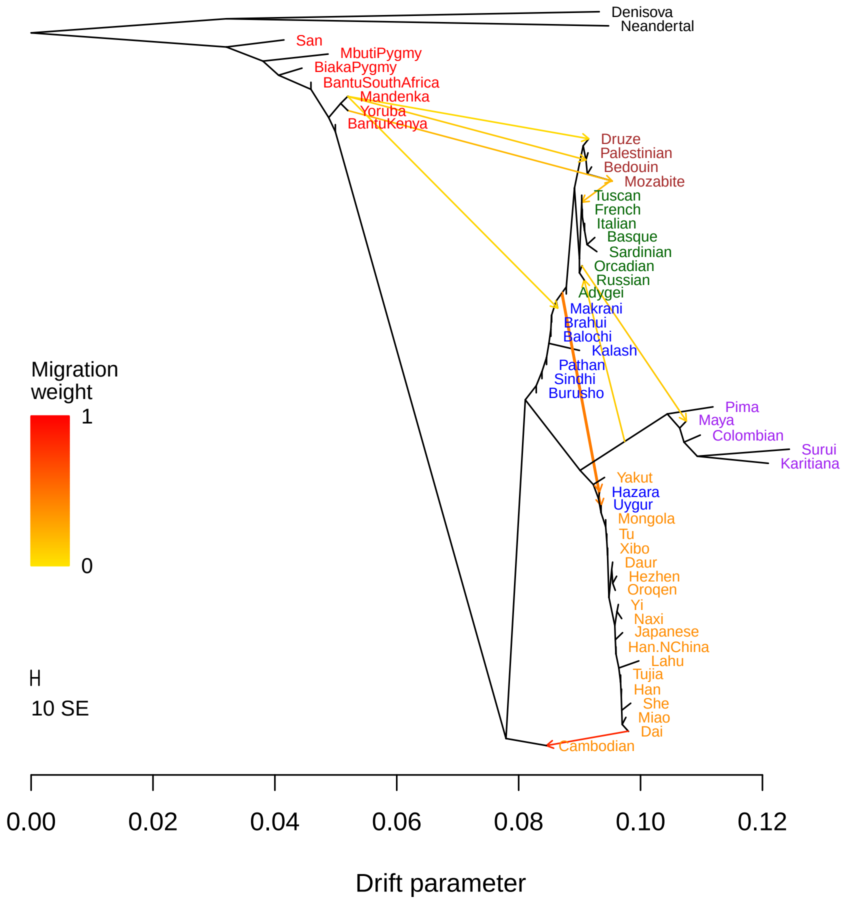


Figure 2: ML tree of 53 human populations with inferred migration edges

```
>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.