

## TUTORIAL FOR USING ANCESTRYMAP

In this document we are going to guide the user through using our software, using a set of example files provided as part of the download. We shall also go through the steps needed to run user specific data on our software.

### Download and Installation

The steps for this are as follows:

1. To download the software go to the following website:  
<http://genepath.med.harvard.edu/~reich>
2. Click on download software for [UNIX](#) or [LINUX](#)
3. Make sure to rename the downloaded file to `ancestrymap.tar.gz` in case the download process has changed the name.
4. Decompress the file `ancestrymap.tar.gz` using `gzip -d ancestrymap.tar.gz`, and you should now see `ancestrymap.tar` in your directory.
5. Unarchive `ancestrymap.tar.gz` by typing on the command line:  

```
>> tar -xvf ancestrymap.tar
```

This should give you the following directory structure under a directory called *ancestrymap/*:

- [examples/](#)
- [bin/](#)
- [src/](#)
- *README* file

In the *examples* directory we have the following files:

#### Parameter Files:

- [paramfile](#): In the format of [parameter file](#) for *ancestrymap*
- [parmono](#): In the format of [parameter file](#) for *cntmono*
- [param0](#), [param1](#): Parameter files for *ancestrymap*, discussed later in this section
- [parsim](#): Parameter file for *ancestrymap* when running simulations

#### Input Data Files:

- [indiv](#): individual input file for *ancestrymap*
- [indiv1.dat](#) : Individual file for *ancestrymap* with some samples set to Ignore
- [genodata](#): genotype input file for *ancestrymap*
- [snp.dat](#): marker input file for *ancestrymap*
- [badsnps](#): input file for *ancestrymap* with markers that need to be removed from the analysis
- [snps](#): marker input file for *cntmono* or *ancestrymap*
- [aflist](#), [eurlist](#): Ancestry files for *cntmono*

The genotype and individual files in this directory were generated by running simulations, and the marker files correspond to data reported in the [Smith et al paper](#).

#### Output Files:

- [outf](#): Output file generated by running *ancestrymap* using [paramfile](#)

- [oupt0](#): Output file generated by running *ancestrymap* using param0
- [outp1](#): Output file generated by running *ancestrymap* using param1
- *badata.dat*: Output file generated by running *admcheck* on outp1
- [outfiles/](#): This is a directory which contains the output files mentioned in Section 6.

#### Executables:

- *cntmono*
- *ancestrymap*
- *admcheck*: A perl script which is used to extract the top “bad” markers

*bin/* has the following executables:

- [cntmono](#)
- [ancestrymap](#)
- *admcheck* A perl script

*src/* has the C source code for making the *ancestrymap* and *cntmono* executables, the library *nicklib.a* and a makefile called *Makefile*.

## Running the example files

After the user has successfully downloaded all the files, the next step is to run the example files included. In the next few sections we will discuss the steps involved, where at each step we will focus on a particular parameter file and its corresponding output file.

First step makes sure that the input files are in the right format, and performs a couple of data checks. In this step we look at the parameter file [param0](#), and its corresponding output file [outp0](#). The key parameter values in this file are *numburn* = 0, *numiters* = 0, *checkit* = YES and *details* = YES.

To run this parameter file type on the command line in the [examples](#) directory:

```
>> ./ancestrymap -p param0 > out0.dat&
```

Compare the output files *out0.dat* and *outp0* (in the *examples* directory) to make sure that you can understand the output generated. Note that the use of the random number generator makes it impossible for the results to be exactly the same for two runs with the same parameter file; however they should be extremely close.

Next, look at the output file *indjunk* generated by this run. From this file one can extract a list of individuals with very small number of genotypes by sorting it by the *Num\_valid\_genotypes* column. We will set the *Status* field to *Ignore* for some of these individuals in a copy of the original individual file called *indiv1.dat* file, since a lot of missing data will cause ANCESTRYMAP to behave badly. Next look at the output file *out0.dat*, where the [checkdup](#) program has flagged a number of duplicate individuals. We shall set the *Status* field to *Ignore* for one of these pair of individuals in the *indiv1.dat* file as well.

Thus the key focus in this step is to ensure that ANCESTRYMAP can successfully process the input files, and the identification of individuals which are duplicates or have very few genotypes.

The next step involves running a lot of data checking programs. In this step we will look at the parameter file [param1](#), and its corresponding output file [outp1](#). The key parameter values in this file are *numburn* = 5, *numiters* = 5, *checkit* = YES and *details* = YES.

This corresponds to having very few burn-in or follow-on iterations and sets up ANCESTRYMAP in the mode to run the various [data checking](#) programs.

To run this parameter file type on the command line in the [examples](#) directory:

```
>>./ancestrymap -p param1 > out1.dat&
```

Compare the output files out1.dat and outp1 (in the *examples* directory) to make sure you can understand the various output sections. Note that the use of the random number generator makes it impossible for the output to be exactly the same for two runs with the same parameter file; however they should be extremely close.

Note that in the output file there are results from a large number of data checking programs. To extract the top markers that have failed the various checks run the perl script admcheck by typing on the command line:

```
>>admcheck out1.dat > ancsycheck.dat&
```

Compare the file ancsycheck.dat with the file baddata.dat in the examples directory.

For an example of the output generated by [admcheck](#) and pointers on how to extract the bad markers, go to: <http://genepath.med.harvard.edu/~reich/admcheck.htm>.

From the *ancsycheck.dat* file we will pick the markers that are outliers for the various checks, and will add them to our *badsnpname* file which will allow the software to ignore these markers for the rest of the analysis. In addition, the user must also add to this file one of the pairs of markers which are in strong linkage disequilibrium with each other. It is necessary to remove these markers since otherwise one will see spurious results. Note that since we don't really have any bad markers, the *badsnps* file in the *examples* directory is just a sample file.

Next we will look at the parameter file [paramfile](#), and its corresponding output file [outf](#). This file corresponds to having 100 burn-in and 200 follow-on iterations, with *checkit* = NO, *details* = NO and uses the *badsnps* file that we created in the previous step.

To do this type on the command line in the [examples](#) directory:

```
>> ./ancestrymap -p paramfile, and to redirect the output to the file out.dat type  
on the command line:
```

```
>> ./ancestrymap -p paramfile > out.dat&
```

Compare the output files *outf* and *out.dat* to make sure you can understand the output generated. Note that the use of the random number generator makes it impossible

for the output to be exactly the same for two runs with the same parameter file; however they should be extremely close.

The important things to focus on in this run are the  $\tau(\text{Afr})$  and  $\tau(\text{Eur})$  values, scores for the various chromosomes and the genome log factor value.

In addition to the standard output, this parameter file will also create a number of [output files](#) in the *outfiles* directory. These files are as follows, and have been discussed in detail in the [documentation](#).

- act.out
- freq.out
- snp.out
- theta.out
- lambda.out
- ethinc.out
- ind.out

## Running your own data files with ANCESTRYMAP

Before going through these steps the user should make sure they are able to follow the steps outlined below using the example files.

1. Go to the *ancestrymap/bin* directory:
2. Create the following input files:
  - snps (list of markers, using the format for [snps](#) or [snpcnts](#)). Note that if you have the marker data in the snps format, and the genotype data for the parental populations you can run the auxiliary program [cntmono](#) to obtain a file with marker data in the snpcnts format. [ An example parameter file for running cntmono is included: *ancestrymap/examples/parmono*. For a tutorial on how to run cntmono go to [http://genepath.med.harvard.edu/~reich/Cntmono\\_Tutorial.htm](http://genepath.med.harvard.edu/~reich/Cntmono_Tutorial.htm)
  - indiv (list of samples, using the format for [indiv](#))
  - genodata (genotype data for all samples and markers, using [genodata](#)'s format). Missing data can be input as -1 or not included at all. There will be a fatal error for including genotypes corresponding to samples or markers not included in their respective files.
3. Type on the command line  

```
>>./ancestrymap -p parc0
```

Use the parameter file *ancestrymap/examples/param0* as an example for *parc0*. This step ensures that the program can read the input files properly. From the file corresponding to the parameter *indoutfilename*, get a list of individuals which have very few genotypes and set their Status field to Ignore in the individual file. Next from the output file generated by this run extract the pair of duplicate individuals (if any), and set the Status field to Ignore in the individual file for one of the pair of individuals.
4. Type on the command line  

```
>>./ancestrymap -p parc1 > ancsy.out
```

Use the parameter file *ancestrymap/examples/param1* as an example. This will run the various data checking programs

5. Type on command line  
**>>./admcheck ancsy.out**

The script *admcheck* will extract the list of the top 10 markers that failed the various checks. Use the [guidelines](#) offered in the documentation to choose the bad markers.

6. Create a file called *badmarkerlist* and put the markers (look for outliers) that failed various checks in this file
7. Add to the *badmarkerlist* file one of the pair of markers that are in linkage disequilibrium with each other.
8. Create a parameter file *param* using *ancestrymap/examples/paramfile* as an example and type on the command line  
**>>./ancestrymap -p param**