SFA INSTRUCTIONS

B ENGELHARDT AND M STEPHENS

1. Background

SFA Version 1.0 released in July 2010 by B. Engelhardt and M. Stephens. This software applies sparse factor analysis to an input matrix.

Please cite: BE Engelhardt and M Stephens (2010) "Analysis of population structure: a unifying framework and novel methods based on sparse factor analysis." *PLoS Genetics* (in press).

2. Prerequisites

In order to compile this program, you need to have the GNU Scientific Library (GSL) in your path. Download and install this software from:
`http://www.gnu.org/software/gsl/`

3. Installation

To compile this software on your computer, run the following commands:

```
gunzip sfa1.0.tar.gz
tar xvf sfa1.0.tar
cd sfa
cd src
make clean
make
```

This should produce an executable called `sfa`. SFA is a command line program. In your terminal window type ./sfa -h for more help. See also these instructions.

4. Command line options

All options are case-sensitive.

4.1. File I/O related options.

- -gen(otype): specify input genotype file (n (individuals) rows, p (loci) columns) with no missing values. See examples in /input folder (**required**)
- -g: number of individuals (rows) in input matrix (**required**)
- -n: number of SNPs (columns) in input matrix (**required**)
- -k or -K: dimension of output matrices (i.e., number of factors) (**required**)
- -o or -out(put): specify the prefix of all output files, use random seed as a default value

- -r or -rand: specify random seed, use system time as default
- -iter: number of iterations of ECME, default is 20
- -t: transpose the matrix after it is read in

4.2. **Model options.**

- -mg: include a mean vector for the n rows (default is no mean vector)
- -vg: the psi variables are the n row variances (default)
- -mn: include a mean vector for the p columns (default is no mean vector)
- -vn: the psi variables are the p column variances

4.3. **Other options.** -h or -help: print this help

5. INPUT

Input to the method is a n (number of individuals) by p (number of loci) matrix with spaces or tabs in between the columns, with no header and no row or column names. If this matrix needs to be transposed, then you can input the transposed version of the matrix with the `-t` command line option. See examples in /input folder. There are no requirements of this matrix, and in particular it does not need to be genotypes.

6. OUTPUT

The basic output is a set of K sparse factor loadings of length n, and a set of K factors of length p. Including these, there are a number of output files:

- <output>_lambda.out: the file of factor loadings, which is an nxK matrix
- <output>_F.out: the file of factors, which is a Kxp matrix
- <output>_alpha.out: the file of the variance parameters, which is an n-vector when residual variances are on individuals
- <output>_eta.out: the file of the variance parameters, which is a p-vector when residual variances are on SNPs
- <output>_sigma2.out: the file of the factor loading variance parameters, which is an nxK matrix
- <output>_mug.out: the file of the mean parameters, which is an n-vector – might not exist when it is not included in model
- <output>_mun.out: the file of the mean parameters, which is a p-vector – might not exist when it is not included in model

In general, you will probably be most interested in the first or second file (the matrix of factor loadings or the matrix of factors).

You can read the lambda matrix into the software package R and plot them against each other, e.g.,

```
lambda <- read.table("<output>_lambda.out")
plot(lambda[,1])
plot(lambda[,1], lambda[,2])
```

7. Examples and intuitions

We have included a number of real and simulated genotype matrices in the `/input` file. Use these to experiment with SFA and see examples of how it works.

- HapMap example with three populations/factors
  ```
  ./sfa -gen ../input/hapmap.sfa -g 210 -k 3 -n 1859 -iter 400 -rand 284 -o test
  ```
- Square grid isolation-by-distance example with two factors, mean term
  - with residual variance terms on individuals
    ```
    ./sfa -gen ../input/habitat1.sfa -g 225 -k 2 -n 1000 -iter 20 -rand 810 -mn -o test2
    ```
  - with residual variance terms on SNPs
    ```
    ./sfa -gen ../input/habitat1.sfa -g 225 -k 2 -n 1000 -iter 20 -rand 482 -mn -vn -o test3
    ```
- Two independent square 2-D habitats
  ```
  ./sfa -gen ../input/habitats1and2.sfa -g 450 -n 1000 -k 6 -rand 115 -iter 100 -o test2hab
  ```
- Isolation-by-distance model
  - using SFA
    ```
    ./sfa -gen ../input/ibd.sfa -g 100 -k 2 -n 1000 -iter 20 -rand 234 -o testibd1
    ```
  - using SFAm
    ```
    ./sfa -gen ../input/ibd.sfa -g 100 -k 1 -n 1000 -iter 20 -rand 111 -mn -o testibd2
    ```
- Isolation-by-distance model with clustered samples
  - using SFA with two factors
    ```
    ./sfa -gen ../input/ibd-grouped.sfa -g 100 -k 2 -n 1000 -iter 20 -rand 123 -o testibdc1
    ```
  - using SFA with five factors
    ```
    ./sfa -gen ../input/ibd-grouped.sfa -g 100 -k 5 -n 1000 -iter 2000 -rand 173 -o testibdc2
    ```
  - using SFAm with one factor
    ```
    ./sfa -gen ../input/ibd-grouped.sfa -g 100 -k 1 -n 1000 -iter 20 -rand 34 -mn -o testibdc3
    ```

Other thoughts.

- You might thin your genotype data, if you are using genotype data from an array with high r^2 between the loci, since this method does not assume they are dependent. One way to do this is to use SmartPCA from the Eigenstrat software to thin the data to a certain r^2 cutoff, then use only those loci for SFA.
- As a warning, if your data has missing SNPs, the factors can be quite noisy and batch-dependent. Use some available method (IMPUTE2 seems

to work rather well for these, but others would be fine I'm sure) to impute the genotypes before running here.

- As a warning, the expected complete log likelihood is not actually the likelihood that is being maximized in EM, so it might go the wrong direction. In general, the marginal log likelihood (which takes longer to compute, and so is computed only every 10 iterations) will increase monotonically.
- More iterations tend to yield a sparser solution; fewer iterations yield a less sparse solution, but too few iterations, and it hasn't converged. Min number of iterations is around 20.
- Try a bunch of random seeds and compute the correlation of the resulting lambda and factors to see which are stable and which are not, in order to start to determine the proper number of factors.
- Try different values for K, the number of factors
- Remove individuals that appear to be outliers

8. QUESTIONS?

We (BE and MS) will continue to develop this method and integrate our new developments into this software. For questions related to the software, email Barbara Engelhardt at engelhardt-at-uchicago.edu or visit us at
`http://stephenslab.uchicago.edu/software.html`