Manual for `fsabc` version 0.2

Zach Gompert

8, May, 2020

Overview

`fsabc` implements an approximate Bayesian computation (ABC) method to detect and quantify fluctuating selection on polygenic traits from time-series data. The direction and magnitude of phenotypic selection is determined by the state of an environmental variable. The population-genomic consequences of selection are then modeled based on estimated genotype-phenotype associations. This allows inferences to be informed by patterns of change across multiple genetic loci, populations, and generations. The goal of the model is to estimate the intensity of selection on the trait and how this varies by environment, as well as the extent to which this translates into selection on different genetic loci. Key assumptions of the model are that selection is always directional, that each locus explains a small proportion of the phenotypic variance (either because the trait is highly polygenic or has a large environmental variance) and that the causal variants (or linked loci) exhibit minimal linkage disequilibrium with each other such that each experiences genetic drift independently. The model also assumes that, at least over the duration of the population-genomic time series, mutation and gene flow are negligible and can be ignored.

The program `fsabc` is written in `C++` with the Gnu Scientific Library (GSL). See Gompert & Bergland (XXXX) for a full description and evaluation of the method. Herein, we provide instructions for installing and running the program.

How to install `fsabc`

Our instructions assume a Linux operating system, such as Ubuntu, but should also work for OSX (Mac) if the appropriate utilities have been installed.

You can download (clone) the source code, manual and example files for `fsabc` from `GitHub`.

```
git clone https://github.com/zgompert/fsabc.git
```

Once you have downloaded the source code, navigate to the `fsabc` directory. You can then compile the the source code with the following command:

```
g++ -O2 -Wall -o fsabc  main.C func.C -lm -lgsl -lgslcblas
```

You must have the Gnu Scientific Library installed and in your path. Successful compilation creates an executable binary, `fsabc`. Running the program without invoking any options prints a help menu listing the command line options (try this to verify the the program was successfully compiled).

```
./fsabc
```

```
##
## ./fsabc version 0.2 -- 01 June 2020
##
## Usage: fsabc -g genefile -e envfile -f nefile -t traitfile [options]
## Use -v 1 for predictive mode, -q 1 for observed mode,
```

```
##  or leave both 0 (default) for simulation model
##
## -g     Infile with allele frequency data
## -e     Infile with environmental covariate data
## -f     Infile with varNe estimates
## -t     Infile with trait genetic arch. estimates
## -j     (optional) Infile with gens. between samples
## -z     (optional) Infile parameter posterior samples
## -v     Binary, run posterior pred. validation mode [0]
## -q     Binary, run observed summary stats. mode [0]
## -o     Outfile for simulated or obs. summary stats. [out_fsabc.txt]
## -h     Phenotypic variance [1.0]
## -n     Number of simulations [1000]
## -s     SS to print: 0 = bv, 1 = snp, 2 = both [0]
## -m     Selection model: 0 = linear, 1 = step, 2 = sigmoid [0]
## -p     Prior prob. of non-zero selection by component [0.5]
## -a     Lower bnd. on U prior for sel. function intercept [-10]
## -c     Upper bnd. on U prior for sel. function intercept [10]
## -b     Lower bnd. on U prior for sel. function slope [-10]
## -d     Upper bnd. on U prior for sel. function slope [10]
## -w     Lower bnd. on U prior for sel. function cut [-1]
## -x     Upper bnd. on U prior for sel. function cut [1]
```

Input data and file formats

`fsabc` requires four data sources.

Estimates of allele frequencies in one or more populations for multiple time points (ideally consecutive generations). The method assumes these are known with little or no error. The method requires a set of genetic markers (e.g., SNPs) known to affect the trait of interest, or as will be more commonly the case, a set of markers statistically associated with the trait (i.e., in linkage disequilibrium with the unknown causal variants). The precise format of the file depends on the run mode (see below for details).

When running the program in simulation or posterior predictive mode, the first row is a header with two values (separated by white space), the number of genetic markers and the number of populations. This is followed by one row per genetic marker. Each row gives the initial allele frequency (for one of the two alleles, e.g., the minor allele) for each population. Thus, there are as many columns as populations. See `sim_p0.txt` for an example with 100 SNPs and 10 populations (in this example all 10 populations have the same initial allele frequencies).

When running the program to compute summary statistics for the observed data, the first row is a header with two values (separated by white space), the number of genetic markers and the product of the number of populations and generations. This is followed by one row per genetic marker. Each row gives the allele frequency (for one of the two alleles, e.g., the minor allele) for each population and generation. Use the following order: Pop0_Gen0 Pop0_Gen1 ... Pop0_Gen10 Pop1_Gen0 ... Pop1_Gen10 ... Pop5_Gen10. See `sim_p0.txt` for an example with 100 SNPs and 10 populations (in this example all 10 populations have the same initial allele frequencies). See `out_example_p.txt`.

Information on the effects (associations) of the genetic markers with the trait. The method assume each marker has a probability of affecting or being associated with the trait and an effect conditional on a true association. The probability of association can be set to 1 for a set of genetic variants to indicate they are known to directly affect the trait; such confidence might be appropriate if genotype-phenotype associations have been validated by genetic manipulations.

The header row gives the number of genetic markers followed by the number of columns (always two). This is

followed by one row per genetic marker. The first column gives the probability that the marker is associated with or affects the trait (i.e., the posterior inclusions probability). If the value is exactly 0, there is no need to include the marker in this file or in the allele frequency file. In other words, only markers with non-zero probabilities of being included in the genotype-phenotype map should be included. The second column gives an estimate of the marker's phenotypic effect conditional on it having a non-zero effect (or association). See `sim_trait.txt` for an example with 100 genetic markers known to be associated with a trait of interest (i.e., all have posterior inclusion probabilities of 1.0).

Estimates of the variance effective population size. These can be estimated using either LD-based methods or change over time for a set of genetic markers not associated with the (putatively) selected trait. The proposed method accounts for uncertainty in estimates of N_e by integrating over a posterior distribution.

The first row is a header with the number of samples (set to 1 if only a point estimate is available) and number of populations. Each subsequent row contains a possible value (posterior sample) of Ne for each population. See `sim_ne.txt` for an example with 100 samples from the posterior and 10 populations.

Measurements of the environmental covariate. This covariate should be measured in each population and generation (or time step).

The first row is a header with the number of populations and number of generations. Each subsequent row gives the environmental covariate data for one population, with one column per generation. Note that the last generation (column) is not used as it would apply to the expected allele frequency in the following (not yet sampled) generation. See `sim_env.txt` for an example with 10 populations and 10 generations.

Program modes and command line arguments

At minimum `fsabc` requires the user to provide files with allele frequency data, environmental data, effective population size estimates, and a genotype-phenotype map (see above). The user should also specify the mode for the program to run in and additional command line options. The default is to run in simulation mode. This involves sampling parameters from their prior distributions, simulating evolution, and computing summary statistics. Alternative modes allow the user to compute summary statistics for the full, observed time series (`-q`) or to conduct posterior predictive simulations (`-v`) for model checking or model comparison. Here, we list all command line options and expand upon some, including those that specify optional input files.

`-g` = Infile with allele frequency data. As noted above, this file has one of two formats depending on the analysis mode.

`-e` = Infile with environmental covariate data. This file contains the environmental covariate data.

`-f` = Infile with varNe estimates. This file contains estimates of the variance effective population size for each population. The method accounts for uncertainty in Ne, and thus expects samples (values) from the posterior distribution of Ne.

`-t` = Infile with trait genetic architecture estimates. This file contains information on the genotype-phenotype map.

`-j` = (Optional) infile with the number of generations between samples. By default, the software assumes one generation spacing between samples for population allele frequencies (and corresponding environmental data). This optional file allows other spacing to be specified. If included, this file has a header row with the number of generations minus one and the number of populations. Each subsequent row gives the spacing for the samples for a population, where the value given is the number of generations (this is used for simulating evolution). The first column gives the interval between the first and second sample, hence the need for one fewer entry than the number of generations.

`z` = (Optional) infile with parameter posterior samples. This is used only for posterior predictive (validation) mode. This file contains samples from the posterior distribution for the selection differential

model parameters a, b, and c (if c is in the model). The first row is a header with the number of samples and parameters. This is followed by one row per posterior sample with the parameter values.

`-v` = Binary, run posterior pred. validation mode. Set to 1 (true) to run the program in posterior predictive mode. The default is 0 (false).

`-q` = Binary, run observed summary stats. mode. Set to 1 (true) to compute summary statistics from the observed data (rather than run simulations). The default is 0 (false).

`-o` = Outfile for simulated or obs. summary stats. Output is written to this file (see the next section for details)

`-h` = Phenotypic variance. The phenotypic variance for the trait, which is assumed to be constant across populations and generations. The default is 1.0.

`-s` = Summary statistics to print. Determines which summary statistics to print. Set to 0 (default) for breeding-value based summary statistics. This is the recommended option. Set to 1 for summary statistics computed for each genetic marker (SNP). This is not meant as a user option at this point (use with caution).

`-m` = Selection model. Specifies the mathematical function relating the environmental covariate to the selection differential. 0 = linear (default), 1 = step, 2 = sigmoid function.

`-p` = Prior prob. of non-zero selection. Denotes the prior probability that each parameter of the selection function (a, b, and c if relevant) is non-zero. See the description of the spike-and-slab prior in Gompert & Bergland for details.

`-a` Lower bound for prior on a. Sets the lower bound of the uniform component of the prior on the intercept parameter (a).

`-c` Upper bound for prior on a. Sets the upper bound of the uniform component of the prior on the intercept parameter (a).

`-b` Lower bound for prior on b. Sets the lower bound of the uniform component of the prior on the slope parameter (b).

`-d` Upper bound for prior on b. Sets the upper bound of the uniform component of the prior on the slope parameter (b).

`-w` Lower bound for prior on c. Sets the lower bound of the uniform component of the prior on selection function parameter c.

`-x` Upper bound for prior on c. Sets the upper bound of the uniform component of the prior on selection function parameter c.

Output

Results from the program are written to the outfile specific with the `-o` option. Specific results depend on the mode.

Simulation mode. When running in simulation mode, the outfile will have one row per simulation. The first two (or three) columns give the sampled values of the selection function parameters, a, b and c (the linear selection function only includes a and b). The next four columns provide four derived parameters that describe phenotypic or genetic selection: (i) the mean selection differential, (ii) the standard deviation for the selection differentials, (iii) the mean absolute intensity of selection on the genetic loci, and (iv) the standard deviation of the absolute intensity of selection on the genetic loci. Assuming breeding-value based summary statistics were specified (`-s 0`), this is then followed by the two summary statistics, the mean change in the expected breeding value (i.e., polygenic score) and the covariance between breeding value change and the environmental state.

Observed summary statistics mode. When running in observed summary statistics mode (with the breeding value level summary statistics), the outfile has only two rows. The first gives the summary statistics, that is the the mean change in the expected breeding value (i.e., polygenic score) and the covariance between breeding value change and the environmental state for the observed data. This can be used for parameter estimation. The second row gives the percentiles for the distribution of change in the mean breeding value (this is used for comparison with the posterior predictive distribution; see below).

Posterior predictive mode. When running in posterior predictive mode, the outfile will have one row per simulation. Columns denote percentiles of the distribution of change in the mean breeding value (polygenic score) across generations and populations. Values are given for the 10th through 90th percentiles in increments of 10 percentile points (nine values per simulation, in order).

Example analysis

Here we provide an example analysis. The true (in this case, simulated) allele frequency data for 10 populations sampled over 10 generations is in `sim_example_p.txt`. The files `sim_env.txt`, `sim_ne.txt` and `sim_trait.txt` contain the environmental data, estimates of effective population size (here assumed to be the same for all 10 populations) and genotype-phenotype map (100 loci, assumed to all affect the trait).

We first compute summary statistics for the observed data.

```
./fsabc -g sim_example_p.txt -e sim_env.txt -f sim_ne.txt -t sim_trait.txt \
 -n 1 -m 0 -q 1 -o obs_example.txt
```

Next, we generate 1 million simulated sets of parameters and summary statistics. Here, we assume a linear model for the selection differential. Note that `sim_p0.txt` has only the first generation allele frequencies for each of the 10 populations.

```
./fsabc -g sim_p0.txt -e sim_env.txt -f sim_ne.txt -t sim_trait.txt \
 -n 1000000 -m 0 -a -0.1 -b -0.1 -c 0.1 -d 0.1 -o sims_example.txt
```

Lastly, we summarize the posterior in `R`. This is done with the `abc` package.

```
library(abc)
```

```
## Loading required package: abc.data
## Loading required package: nnet
## Loading required package: quantreg
## Loading required package: SparseM
##
## Attaching package: 'SparseM'
## The following object is masked from 'package:base':
##
##     backsolve
## Loading required package: MASS
## Loading required package: locfit
## locfit 1.5-9.4    2020-03-24
```

```
library(scales)

## read in 1 million simulations
sims <- matrix(scan("sims_example.txt", n = 1e+06 * 8, sep = " "), nrow = 1e+06,
```

```
    ncol = 8, byrow = TRUE)

## read in obs. summary statistics
obs <- read.table("obs_example.txt", fill = TRUE)
obs <- obs[1, 1:2]  ## strip predictive dist.

## true values for a and b; these are taken from out_example.txt
a <- 0.022
b <- -0.073

## split matrixes and name columns columns 7 and 8 are the summary
## stats.
ss <- sims[, 7:8]
## these are the mean chng. in breeding value and the cov between change
## and the environment
colnames(ss) <- c("ss.mn", "ss.cov")

## columns 1:6 are the parameters and derived parameters
parm <- sims[, 1:6]
## a and b are the two selection function parameters mnS and sdS are the
## mean and sd of the selection diff. (derived params) mnsi and sdsi are
## the mean and sd of the abs. selection coef. for loci (derived params)
colnames(parm) <- c("a", "b", "mnS", "sdS", "mnsi", "sdsi")
o <- abc(target = as.matrix(obs), param = parm, sumstat = ss, method = "loclinear",
    tol = 0.001)
```

```
## Warning: All parameters are "none" transformed.
```

```
summary(o)
```

```
## Call:
## abc(target = as.matrix(obs), param = parm, sumstat = ss, tol = 0.001,
##     method = "loclinear")
## Data:
##  abc.out$adj.values (1000 posterior samples)
## Weights:
##  abc.out$weights
##
##                              a       b     mnS     sdS    mnsi    sdsi
## Min.:                  -0.0072 -0.1011 -0.0041  0.0416  0.0000  0.0001
## Weighted 2.5 % Perc.:   0.0129 -0.0969  0.0148  0.0508  0.0000  0.0001
## Weighted Median:        0.0414 -0.0773  0.0435  0.0737  0.0000  0.0001
## Weighted Mean:          0.0411 -0.0771  0.0433  0.0735  0.0000  0.0001
## Weighted Mode:          0.0388 -0.0735  0.0410  0.0709  0.0000  0.0001
## Weighted 97.5 % Perc.:  0.0636 -0.0539  0.0661  0.0925  0.0000  0.0001
## Max.:                   0.0735 -0.0439  0.0743  0.0968  0.0000  0.0001
```

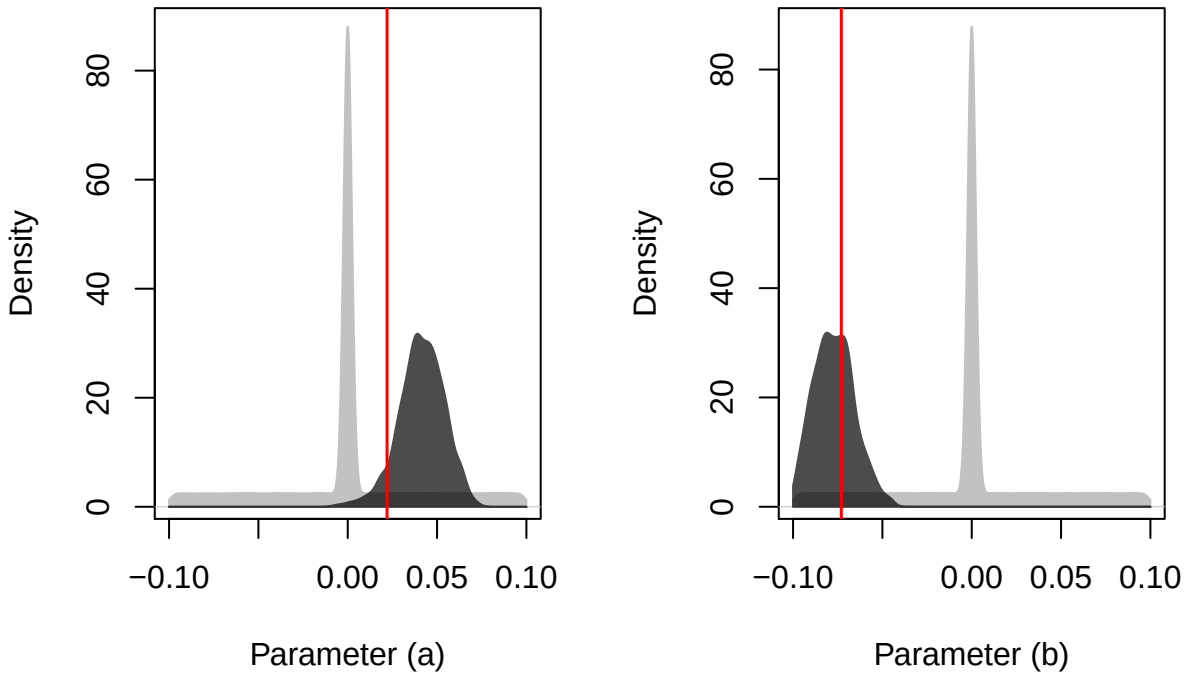
```
library(scales)
c1 <- alpha("darkgray", 0.7)
c2 <- alpha("black", 0.7)

## the prior distribution for a and b is shown in gray, the posterior is
## in black the true value is given by the red vertical line

par(mfrow = c(1, 2))
```

```r
ddpr <- density(parm[, 1], from = -0.1, to = 0.1)
ddpost <- density(o$adj.values[, 1], from = -0.1, to = 0.1)
plot(ddpr, type = "n", xlab = "Parameter (a)", main = "")
polygon(c(ddpr$x, rev(ddpr$x)), c(ddpr$y, rep(0, length(ddpr$y))), col = c1,
    border = c1)
polygon(c(ddpost$x, rev(ddpost$x)), c(ddpost$y, rep(0, length(ddpost$y))),
    col = c2, border = c2)
abline(v = a, col = "red", lwd = 1.5)

ddpr <- density(parm[, 2], from = -0.1, to = 0.1)
ddpost <- density(o$adj.values[, 2], from = -0.1, to = 0.1)
plot(ddpr, type = "n", xlab = "Parameter (b)", main = "")
polygon(c(ddpr$x, rev(ddpr$x)), c(ddpr$y, rep(0, length(ddpr$y))), col = c1,
    border = c1)
polygon(c(ddpost$x, rev(ddpost$x)), c(ddpost$y, rep(0, length(ddpost$y))),
    col = c2, border = c2)
abline(v = b, col = "red", lwd = 1.5)
```




```r
## summarize the relationship between S and the environment
x <- seq(-2, 2, 0.01)  ## approx. range of environmental variation
nx <- length(x)
y <- matrix(NA, nrow = dim(o$adj.values)[1], ncol = nx)
for (i in 1:dim(o$adj.values)[1]) {
    ## linear model, computer over the post.
    y[i, ] <- o$adj.values[i, 1] + x * o$adj.values[i, 2]
}
## compute 91% credible intervals (91 is just for fun, compute what you
## want)
est <- apply(y, 2, quantile, probs = c(0.5, 0.045, 0.955))

## in the plot below, the point estimate for the relationship between
## the environment and S is shown with a line, the CIs are shown by the
```

```r
## shaded region
par(mfrow = c(1, 1))
plot(x, est[1, ], col = "blue", type = "l", lwd = 1.8, xlab = "Environment",
    ylab = "Selection differential (S)", cex.lab = 1.2)
polygon(c(x, rev(x)), c(est[2, ], rev(est[3, ])), col = alpha("blue", 0.4),
    border = NA)
```