fastsimcoal

fastsimcoal

Firstly before proceeding I want make it clear that this tutorial was originally designed by my colleague and friend, Joana Meier and has been adapted from here for this course. The depth of this tutorial would not have been possible without her input!

`fastsimcoal2` is an extremely flexible demographic modelling software developed by Laurent Excoffier and his group at University of Bern. It uses the site frequency spectrum (SFS) to fit model parameters to the observed data by performing coalescent simulations. You can find the manual and more information here.

It is worth noting that this is a complex piece of software to run and it takes a lot of time to master. However it is really worth taking your time with it and exploring the manual. It is extremely well documented (which is rare for a program like this).

Preparing the input files

To run `fastsimcoal2` we need three input files which are all named in a consistent way. They are all just plain text files.

- observed SFS - `${PREFIX}_jointDAFpop1_0.obs`
- template file defining the demographic model - `${PREFIX}.tpl`
- estimation file defining the parameters - `${PREFIX}.est`

Observed SFS

The observed SFS can be a derived SFS (i.e. also known as DAF or an unfolded SFS) if the ancestral state is unknown or a minor allele frequency SFS (i.e. MAF or folded SFS) if you do know the ancestral state. `fastsimcoal2` will identify the kind of SFS by its name suffix and the command flag `-m` or `-d`. For a single population, it expects the name `${PREFIX}_DAFpop0.obs` or `${PREFIX}_MAFpop0.obs`.

For two populations, the file names should end with `jointDAFpop1_0.obs` or `jointMAFpop1_0.obs`. If more than two observed populations are used, pairwise MAFs or DAFs can be provided following the naming scheme above or a multidimensional SFS can be used ending with `_DSFS.obs` or `_MSFS.obs` and specifying the `-multiSFS` flag when running the program. See the fastsimcoal2 manual for further details. The `.obs` file can be generated with `Arlequin`, `angsd`, `easySFS` (as in this tutorial) or several other tools.

Template file

The template file describes the demographic model and the parameters of interest. Values that should be estimated (i.e. model parameters such as migration rate, splitting time etc) are given as different keywords. It is best practise to use capital letters for parameter names and it is important to avoid any names that are part of another parameter name or a function (e.g. `log`, `min`, `FREQ`).

As fastsimcoal2 is a coalescent simulator, all models need to be specified **backwards in time**. This means that the most recent event is specified first. The template file is best produced by modifying an example `tpl` file in a text editor.

In this tutorial, we will first write a model of two species that diverged in the face of gene flow and then evolved complete reproductive isolation. Unfortunately, we do not have a good estimate for the mutation rate, but we have an estimate for the (maximum possible) splitting time. So here we will set the split between the populations to 6000 generations.

Question: Why do we need to fix a parameter (e.g. splitting time) if we do not have a reliable mutation rate estimate?

Estimation file

All keywords introduced in the template file need to be defined in the estimation file. For each keyword, the parameter distribution (uniform or log-uniform) and search range (min and max) are given on a single line. Each parameter can be an integer or a float, as specified by a first indicator variable.

Lastly, complex parameters can be used to compute parameter values with simple operations such as computing the ratio between two simple parameters or specifying a parameter as the minimum of two parameters. In addition, for each simple or complex parameter, you need to specify if the parameter value should be written into an output file or not.

As with the template file, the estimation file is best produced by modifying an example `est` file in a text editor.

Running fastsimcoal2

Once we have all input files ready, it is time to run `fastsimcoal2`. In addition to the input files, we need to specify how many simulations and iterations `fastsimcoal2` should perform, when to stop and how many threads (i.e. CPUs) can be used in parallel.

Let's run `fastsimcoal2` with our model of two populations. First we make a `fastsimcoal2` directory and within that a directory, an additional one for our model of `early_geneflow`:

```
cd ~
mkdir fastsimcoal2
cd fastsimcoal2
mkdir early_geneflow
```

Next, we will move inside this directory and copy over the files we need

```
cd ~/fastsimcoal2/early_geneflow
cp /resources/riverlandsea/exercise_data/fastsimcoal2/early_geneflow/* .
```

This should mean we have our observed SFS, our template file and our estimation file. We are now ready to run `fastsimcoal2` - we will set a `PREFIX` variable to make our command here a bit easier to write. This will help downstream too!

```
PREFIX="early_geneflow"
fastsimcoal2 -t ${PREFIX}.tpl -e ${PREFIX}.est -m -0 -C 10 -n 10000 -L 40 -s 0 -M
```

This command runs `fastsimcoal2` using a MAF (`-m`) while ignoring monomorphic sites (`-0`) and SFS entries with less than 10 SNPs (`-C`). This means that entries with less than 10 SNPs are pooled together. This option is useful when there are many entries in the observed SFS with few SNPs and with a limited number of SNPS to avoid overfitting.

`fastsimcoal2` will also perform (`-n`) 10,000 coalescent simulations to approximate the expected SFS in each cycle and will run (`-L`) 40 optimization (ECM) cycles to estimate the parameters. The number of ECM cycles should be at least 20, better between 50 and 100. The number of coalescent simulations should ideally be something between 200,000 and 1,000,000 but to make it faster, we are now only running 10,000 simulations. We also specify (`-M`) that we want to perform parameter estimation. With `-s 0`, we can tell `fastsimcoal2` to output SNPs.

Once `fastsimcoal2` is finished, we can have a look at the output files. It produced a folder called `${PREFIX}` which contains a number of new files. The most relevant files are the following:

- `${PREFIX}.bestlhoods:` a file with the Maximum likelihood estimates for each parameter specified "output" in the `est` file and the model likelihoods.
- `${PREFIX}._jointMAFpop1_0.txt`: a file with the expected SFS obtained with the parameters that maximized the likelihood during optimization. This is needed to visually check the fit of the expected SFS to the observed SFS. This file has the same suffix as the observed SFS provided.
- `${PREFIX}.simparam`: a file with an example of the settings to run the simulations. This is useful to check when you have errors. Many times errors in specification of models can be detected in this file.
- `${PREFIX}_maxL.par`: The model specification file with the best parameter estimates. It is basically the tpl file with the keywords replaced by estimated values. This file is useful if you want to simulate data under the best model using Arlequin.

Note, the bestlhoods file contains two different likelihoods: `MaxObsLhood` is the maximum possible value for the likelihood if there was a perfect fit of the expected to the observed SFS, i.e. if the expected SFS was the relative observed SFS. `MaxEstLhood` is the maximum likelihood estimated according to the model parameters. It is obtained by using the observed SFS as the expected SFS when computing the likelihood, i.e., returning the value of the likelihood if there was a perfect fit between the expected and observed SFS.

The better the fit, the smaller the difference between `MaxObsLhood` and `MaxEstLhood`.

Finding the best parameter estimates

`fastsimcoal2` should not just be run once because it might not find the global optimum of the best combination of parameter estimates right away. It is better to run it 100 times or more. Of these runs, we then go on to select the one with the highest likelihood which is the run with the best fitting parameter estimates for this model.

Due to time constraints, we will only run this model 5 times and we will do so within a `for` loop. Note, that here we have added the flag `-q` for "quiet" which reduces the amount of information `fastsimcoal2` writes to stdout. Pay attention the `cd` commands here as they ensure that the analyses are done in the correct directory.

```
for i in {1..5}
do
  mkdir run$i
  cp ${PREFIX}.tpl ${PREFIX}.est ${PREFIX}_jointMAFpop1_0.obs run$i"/"
  cd run$i
  fastsimcoal2 -t ${PREFIX}.tpl -e ${PREFIX}.est -m -0 -C 10 -n 100 -L 40 -s0 -M -q
  cd ..
done
```

To find the best run, i.e. the run with the highest likelihood, or better the smallest difference between the maximum possible likelihood (`MaxObsLhood`) and the obtained likelihood (`MaxEstLhood`), we can check the `.bestlhoods` files.

```
cat run{1..5}/${PREFIX}/${PREFIX}.bestlhoods | grep -v MaxObsLhood | awk '{print NR,$8}' |
```

Note that `NR` in awk prints out the line number which here corresponds to the run number. `$8` is the `MaxEstLhood` column and thus the likelihood we want to compare across different runs.

Joana Meier has written a script for you that automatically extracts the files of the best run and copies them into a new folder which it calls `bestrun`. A copy of that script is available in the course resources directory so we will copy it to our run folder directory and run it like so:

Just run it in the directory where all the folders run are located and run it:

```
cd ~/fastsimcoal2/early_geneflow
cp /resources/riverlandsea/exercise_data/fastsimcoal2/fsc-selectbestrun.sh .
./fsc-selectbestrun.sh
```

Now modify the `$PREFIX.tpl` and `$PREFIX.est` files to specify different models and also rename the SFS to `${PREFIX}_jointDAFpop1_0.txt`. Then run `fastsimcoal2`for all models to see which one shows the best fit to the observed SFS.

Click here to see a possible solution.

Early Geneflow (this is the model expalined in the example, gene flow right after the split and then no gene flow anymore, e.g. speciation with gene flow and then no gene flow anymore as reproductive isolation becomes strong):

- https://github.com/speciationgenomics/data/blob/master/early_geneflow.tp
- https://github.com/speciationgenomics/data/blob/master/early_geneflow.est

No geneflow:

- https://github.com/speciationgenomics/data/blob/master/no_geneflow.tpl
- https://github.com/speciationgenomics/data/blob/master/no_geneflow.est

Recent gene flow (no gene flow intially after the split but gene flow in recent times, e.g. secondary contact scenario)

- https://github.com/speciationgenomics/data/blob/master/recent_geneflow.tpl
- https://github.com/speciationgenomics/data/blob/master/recent_geneflow.est

Different gene flow matrices (higher or lower gene flow right after the split than recently, e.g. initially higher gene flow then lower gene flow as reproductive isolation accumulates):

- https://github.com/speciationgenomics/data/blob/master/diff_geneflow.tpl
- https://github.com/speciationgenomics/data/blob/master/diff_geneflow.est

Constant gene flow (same gene flow strengths since the split until now):

- https://github.com/speciationgenomics/data/blob/master/ongoing_geneflow.tpl
- https://github.com/speciationgenomics/data/blob/master/ongoing_geneflow.est

Model comparison with AIC

In order to find the best model, the likelihoods of the best run of each model should be compared. Comparing raw likelihoods is problematic, because a model with more parameters will always tend to result in a better fit to the data. Therefore, the Akaike information criterium or AIC is typically calculated to determine if the models differ in their likelihoods accounting for the number of parameters in each model. To perform, this, we can use a script that is mostly based on R code by Vitor Sousa. We will move into our `bestrun` directory and run this script

```
cd bestrun/
cp /resources/riverlandsea/exercise_data/fastsimcoal2/calculateAIC.sh .
./calculateAIC.sh early_geneflow
```

This script generates a file `${PREFIX}.AIC` which contains the delta likelihood and the AIC value for that run.

Visualize the model fit

To visualize the fit of the simulated SFS to the data, we can use an `R` script that David Marques wrote - `SFStools.r` - which you can download here.

To visualize the model with the best parameter estimates, we can use one of Joana Meier's R scripts - plotModel.r. There are also other options here. However for the ease of this practical, both are available in the resources directory. We will copy them over and then run them.

```
cp /resources/riverlandsea/exercise_data/fastsimcoal2/*.r .
Rscript SFStools.r -t print2D -i early_geneflow
Rscript plotModel.r -p early_geneflow -l NyerMak,PundMak
```

Now, we will download the PDFs genertated and take a look at them.

Model comparison with Likelihood distributions

One drawback of the composite likelihoods in model tests based on AIC is that it can overestimate the support for the most likely model if the SNPs are not independent (here they are not LD-pruned). Another way to infer if the models are really different and do not just differ because of stochasticity in the likelihood approximation, is to get likelihood distributions for each model. This is done by running each model with the best parameter values multple times (ideally about 100 times). The likelihoods will differ because `fastsimcoal2` does not compute the likelihood but rather approximates it with simulations. If the ranges of likelihoods of two models overlap, it means that they do not differ significantly, i.e. provide an equally good fit to the observed data.

Let's recompute the likelihood for the best run for the `early_geneflow` model.

```
PREFIX="early_geneflow"
cd ~/fastsimcoal/$PREFIX/bestrun

# create temporary obs file with name _maxL_MSFS.obs
cp ${PREFIX}_jointMAFpop1_0.obs ${PREFIX}_maxL_jointMAFpop1_0.obs

# Run fastsimcoal 20 times (in reality better 100 times) to get the likelihood of the obse
for i in {1..20}
do
 fastsimcoal2 -i ${PREFIX}_maxL.par -n1000000 -m -q -0
 # Fastsimcoal will generate a new folder called ${model}_maxL and write files in there

 # collect the lhood values (Note that >> appends to the file, whereas > would overwrite i
 sed -n '2,3p' ${PREFIX}_maxL/${PREFIX}_maxL.lhoods  >> ${PREFIX}.lhoods

 # delete the folder with results
 rm -r ${PREFIX}_maxL/
done
```

We would now repeat this for different models: ongoing gene flow, a model without gene flow, a model of secondary contact (recent gene flow) and a model with different amounts of gene flow in the past and in recent times.

Due to time constraints, we will give you the results of these models. They are in the same format as the folder we generated for the model with early gene flow. Load them into your own directory. Note, the `-r` flag stands for "recursive" and allows to also copy directories and their contents.

```
cd ~/fastsimcoal2/
cp -r /resources/riverlandsea/exercise_data/fastsimcoal2/extramodels
```

Now, let's access the R Studio sderver and plot the likelihoods.

```
# Read in the likelihoods
early_geneflow<-scan("early_geneflow.lhoods")
ongoing_geneflow<-scan("ongoing_geneflow.lhoods")
diff_geneflow<-scan("diff_geneflow.lhoods")
recent_geneflow<-scan("recent_geneflow.lhoods")
no_geneflow<-scan("no_geneflow.lhoods")

# Plot the likelihoods

par(mfrow=c(1,1))
boxplot(range = 0,diff_geneflow,recent_geneflow,early_geneflow,ongoing_geneflow,
        no_geneflow, ylab="Likelihood",xaxt="n")
axis(side=1,at=1:5, labels=c("early+recent","recent","early","constant","no"))
```

Similarly, we should compare the AIC values and see if AIC suggests that the second-best model is significantly less good than the best model. Given that we already used the calculateAIC.r script to compute AIC values, this can easily be done.

```
for i in */bestrun/*AIC
do
echo -e `basename $i`"\t"`tail -n $i` >> allmodels.AIC
done
```

And we're done for this tutorial! It is worth noting that linkage among SNPs can actually distort the AIC and likelihood calculations. As a result it is best practice to perform block-bootstrapping to account for this. We don't have time to implement that here but you can refer to the original version of this tutorial which does have a guide for how to achieve this.