X-ORIGIN Tutorial

Qixin He*† and Joyce Rodrigues do Prado‡

†Department of Ecology and Evolution, University of Chicago, Chicago IL, 60637,USA

‡Departamento de Ciências Biológicas, Escola Superior de Agricultura 'Luiz de Queiroz', Universidade de São Paulo, Piracicaba, Brazil

December 1, 2017

Contents

*heqixin@uchicago.edu

1 Introduction

This tutorial gives step-by-step guidance through the pipeline in estimating origin of past expansions. Before applying the pipeline, users are strongly recommended to examine whether the data contains evidence of past expansions (details explained below). If so, whether the data represents multiple expansion origins or single origin? The pipeline can be divided in three parts:

1. range expansion simulations over combination of parameter ranges;
2. conversion of summary statistics;
3. estimation and validation of expansion origins.

The tutorial will present required input files or scripts at each step.

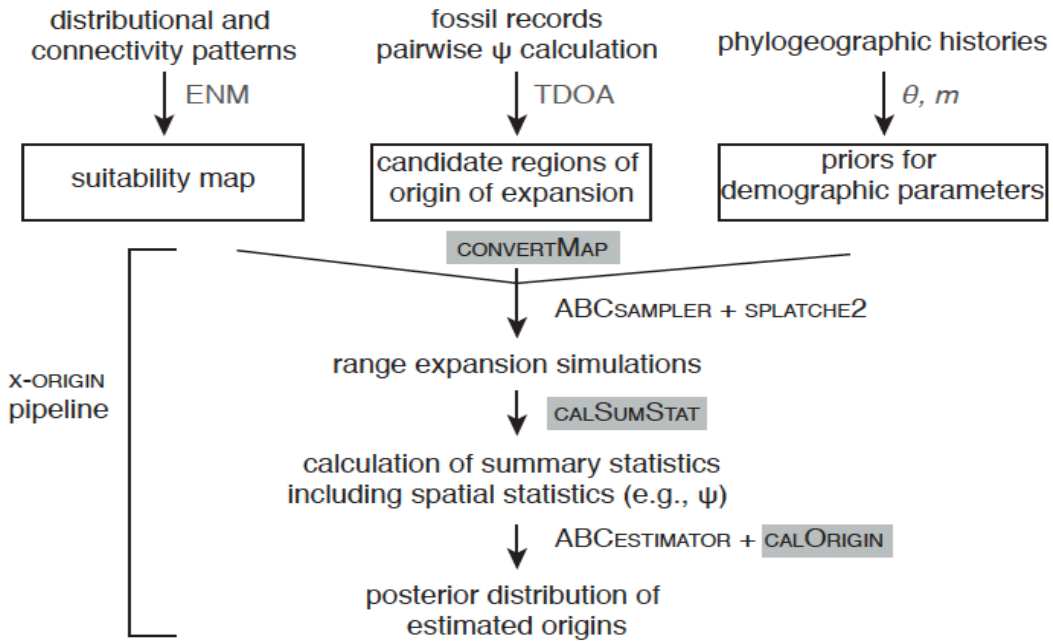

Figure 1: Flowchart of **X-ORIGIN** pipeline.

Prerequisite: linux or windows system with R and Python installed.

Before running the pipeline Users should calculate population pairwise ψ values first to inspect if there is signature of expansion left in the data. `precheck.R` calculates pairwise ψ, which requires two input files: the empirical SNP file (see `precheck/example.snapp`) and the location file (`precheck/example_loc.txt`).

The SNP file has SNP per line for each individual. Genotype follows the individual's id immediately, with 0 indicating ancestral homozygote, 1 indicating heterozygote, 2

indicating derived homozygote of the SNP, and '?' for missing genotype. Location file contains columns of id, latitude, longitude and region division. Region division is for grouping individuals based on different origins. If there is one origin, all individuals should have the same region code.

Running `precheck.R`:

```
Rscript precheck.R example.snapp example_loc.txt
example_PsiOut.txt
```

After running `precheck.R`, if all the ψ values are very close to zero, then there is no expansion signature left in the data, and users should not perform the following analysis.

2 Spatial simulations

This section explains required files for the spatial simulations of range expansions. The relevant files under `example1` folder can be categorized into the following six types:

1. setting files that specify the spatial simulation information:

   `toy_linux.input`, `splatche2input` folder: `1-settings.txt`;

2. suitability maps that describe the changing/static landscapes.

   `splatche2input` folder: `3-oriworld.asc`, `4-dynamic_K.txt` (and the files specified within);

3. location of sampling sites

   `splatche2input` folder: `5-Arrival_cell.col`, `6-GenSamples.sam`;

4. types and number of gene markers that are being generated.

   `splatche2input` folder: `7-genetic_data_SEQ.par`;

5. candidate regions of origin of expansion

   `toy.est`, `spalatche2input` folder: `2-dens_init.txt`;

6. priors for demographic parameters

   `toy.est`.

2.1 Demographic and Connectivity patterns: Modeling Habitat Suitability

`SPLATCHE2` requires a categorical map that defines where the spatial simulations take

place (in the example folders, they are called `3-oriworld.asc`. The file is in ASCII (`.asc`) raster format is commonly used in GIS-related software. It begins with a header of six lines with basic information of the map, followed by specific values for the matrix depicting the region of interest. Demes that have the same values (category) will have same values of carrying capacities (specified in `dynamic_K.txt`) in the simulation (i.e., change in the same way during temporal shifts of habitat). We illustrate here with two examples.

2.1.1 Temporal changes in habitat with barriers

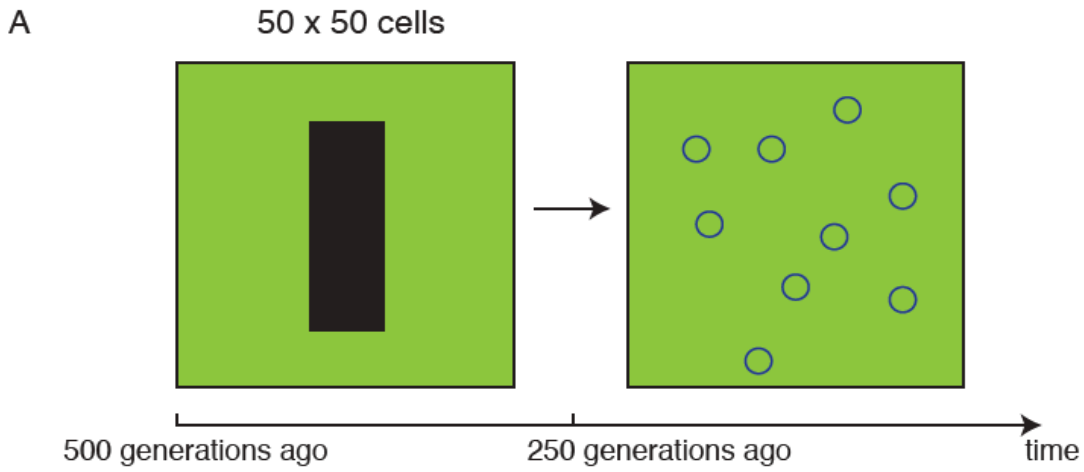

Figure 2:

If the suitability for the species is homogeneous except for the central barrier region as shown above, the demes in the landscape can be categorized into two types: demes within the barrier, which is not habitable from 500-250 generations ago, but become habitable from 250 generations ago to current; demes outside the barrier that is constantly habitable.

The input map is therefore `an.asc` file with two categories:

```
ncols           50
nrows           50
xllcorner       0
yllcorner       0
cellsize        1
NODATA_value    -9999
1 1 1 1 ····· 1 1 1 1
1 1 1 1 ····· 1 1 1 1
```

```
......
1 1 1 1 ··· 1 0 0 0 0 0 0 0 1 ··· 1 1 1 1
1 1 1 1 ··· 1 0 0 0 0 0 0 0 1 ··· 1 1 1 1
......
1 1 1 1 ····· 1 1 1 1
1 1 1 1 ····· 1 1 1 1
```

With 0 indicates barrier, and 1 indicates region outside the barrier. Therefore, the translation table (`pastK.txt`) for the category to actual carrying capacities for 500-250 generations ago is:

```
0       0       barrier
1       K       habitable
```

The first column is the category, the second column is the actual carrying capacity, and the third column is a description.

Similarly, the translation table (`presentK.txt`) for the category to actual carrying capacities for 250-0 generations ago is:

```
0       K       barrier
1       K       habitable
```

Note that these two files contain a "K" parameter inside, because the actual value will change for each spatial simulation based the sampled prior for the particular simulation. For each simulation, ABC`SAMPLER` will sample a specific value for the parameter, and generates a temp file with the actual value for each input file that contains parameter names, called `[filename]-temp.txt`.

Also note that for any translation table that is not used at the beginning of the range expansion simulation, the minimum value you can set for K is 1. Setting it to 0 will encounter problem as the population will stay static, without decreasing. This is because `SPLATCHE2` calculates population size for each generation using the following formula: $N_{t+1} = N_t(1 + r\frac{(K-N)}{K})$. Therefore, when K=0, the formula will not evaluate properly.

`4-dynamic_K.txt` sets how the carrying capacities change in time:

```
2
1       ./splatche2input/pastK-temp.txt    past
250     ./splatche2input/presentK-temp.txt    current
```

The first line depicts how many temporal layers there are, and the following lines specify which layer to use at which time. Note that the time is specified in a time-forward simulation manner.

2.1.2 Temporal changes with suitability maps inferred from Ecological Niche Modeling (ENM)

Through ENM on bioclim data, we can infer habitat suitability at different time period: current, Mid Holocene and Last Glacial Maximum (`http://worldclim.org/paleo-climate1`). These layers cannot be directly used in `SPLATCHE2` for spatial simulations. We need to find out demes that have same temporal changes during different time periods, and generate a categorical map and dynamic K mapping file.

In the example2, habitat suitability during the present and LGM was estimated from bioclimatic variables with `MAXENT v3.3.3e` (Phillips *et al.*, 2006). Full procedural details to generate the environmental niche models (ENMs) will not be presented.

To have a tractable number of cells for demographic simulations, the cell sizes of the ENMs were statistically downscaled for the current and past climatic conditions to $\sim 50\text{km}^2$ per cell, using the function *aggregate* in `ARCGIS`, with the parameter *aggregation technique* set for median. Through the *reclassify* function, the float values of suitability were converted to integers by dividing the original values into 10 classes of equal intervals using the *equal interval* clustering method. Lastly, the function *raster to ascii* was used to transform the raster map to ASCII format.

Once the two rescaled maps are in ascii format, the "`asc2dynKF.py`" script under the mapTransformation folder generates the `oriworld.asc` file that combines the demes with same trends in the LGM and current maps into a categorical map with the following command:

```
python asc2dynKF.py -a LGM.asc -c CURRENT.asc
```

The output files include the map file `oriworld.asc`, and the category number to carrying capacity files for three different time periods that are named: `veg2K_cur.txt`, `veg2K_int.txt`, `veg2K_lgm.txt`.

OBSERVATION: The above script generates veg2K files that have parameter names for converting carrying capacities for use with `ABCtoolbox` (e.g., k_1, k_9, etc.). k_1 means 10% of the maximum carrying capacity, while k_9 means 90% of the maximum carrying capacity. However for testing purposes with the GUI version of `SPLATCHE2`, user needs to include actual values of carrying capacities, instead of variable names. To do that, run the above python script with additional options:

```
python asc2dynKF.py -a LGM.asc -c CURRENT.asc -g -K [maximum K value]
```

to generate veg2K output into GUI-appropriate ones based on the maximum carrying capacities from the prior. At the end of this step you must have the following files:

- `oriworld.asc`

- veg2K_cur.txt
- veg2K_int.txt
- veg2K_lgm.txt

2.2 Origin of expansion

– dens_init.txt

This file specifies the geographic location where the demographic simulation starts. The first line indicates how many origins there are. And the second line specifies name of the origin, population size, latitude, longitude, resizing information, etc. If there is a single origin, the file looks like:

```
1
ori    N_ANCESTRAL   oriLat oriLon  0    0    0    0    0    0
```

ori is the name of the origin. N_ancestral, oriLat and oriLon are three parameters of interest. N_ancestral is the ancestral population size before expansion. oriLat and oriLon indicates the latitude and longitude of the origin. These parameters will be replaced by actual values for each iteration of the simulation. Rest of the fields indicates resizing or migration among origins, which are not used in our simulations. Please read SPLATCHE2 manual for the details of these fields (Ray *et al.*, 2010).

2.3 Sampling location files

– GenSamples.sam

This file specifies the localization of the sampled populations, as well as the number of genes sampled in each of them. The first line is the number of sampled populations. The second line can be a commented line of column names (marked by #). The following lines define a population sample with 5 fields separated by "tab" or "space" character: name, number of genes (diploid or haploid), identification of the population layer to which the sampled deme belongs (if SPLATCHE2 is not run for two competing species, then this column always 0), latitude and longitude:

```
4
#Name    #Size    #PopLayer    #Lat    #Long
1        8        0            16      5
2        14       0            17      2
3        10       0            12      20
4        10       0            4       24
```

– **Arrival_cell.col**

The file that indicates the location of the demes for which arrival times will be calculated. We keep the demes to be tracked the same as the sampling locations, because we want to make sure that by the end of the simulation all the demes are at least colonized. In the statistics calculation step, the output file of colonization times from these demes will be checked to decide if the current simulation should be included.

```
4
#Name    #PopLayer    #Lat    #Long
1        0            16      5
2        0            17      2
3        0            12      20
4        0            4       24
```

2.4 Gene markers file

– **Genetic_data_SEQ.par**

This file specifies which type of gene markers should be generated and how many of them should be generated. This is required for the backward coalescent process after the forward simulation of demographic processes.

Users can use "`writeGeneSeqFile.py`" script to generate the file specific for SNPs in the following way:

```
python writeGeneSeqFile.py [NumberofSNP] [numberOfTotalSamples]
```

The `Genetic_data_SEQ.par` declares the number of chromosomes to simulate. For this example, we want to generate 1000 independent SNPs. Therefore we specify 1000 chromosomes. The following lines specify characteristics for each of the chromosome. In this case, all the SNPs will be equivalent: with the first line indicates 1 block will be generated, and the second line indicates that the type of marker is one SNP, with per generation recombination equals 0, and the minimum frequency of the derived allele equals 1/(2*number of total diploid individuals).

```
1000 //Num chromosomes
#chromosome 1, //per Block:data type, number of SNP, per generation
recombination, Minimum frequency for the derived allele)
1
SNP    1    0    0.010000
#chromosome 2, //per Block:data type, number of SNP, per generation
recombination, Minimum frequency for the derived allele)
1
```

```
SNP    1    0    0.010000
#chromosome 3, //per Block:data type, number of SNP, per generation
recombination, Minimum frequency for the derived allele)
1
SNP    1    0    0.010000
```

2.5 Splatche2 settings file

– settings.txt

This file tells the location and names of all the required input files as well as some parameter choices. For a complete explanation about all the parameters in the Settings File see Ray *et al.*, 2010).

Here we explain several specific choices for **X-ORIGIN**.

```
#Identificator for the demographic model (MANDATORY)
ChosenDemographicModel=3
```

This means that the demographic model is a stochastic migration model with absolute number of emigrants. The number of emigrants of a deme per generation is a Poisson variable centered around $N_i m$, where N_i is the population size of the focal deme at the time.

```
#Migration rate for neighboring deme migration (MANDATORY)
MigrationRate= MRATE
```

Migration rate is set as a parameter in the simulation.

```
#Ancestral population size.  Size of the ancestral population at tau
(NOT MANDATORY)
AncestralSize= N_ANCESTRAL
```

Ancestral population size is set as a parameter in the simulation. This is relevant for the coalescent simulation.

```
#Maximum number of total generations for a simulation.  This number
corresponds to the number of generation for the demographic simulation +
the extra generations for the coalescence process prior to time 0
(MANDATORY)
MaxNumGenerations=1400000
```

Please set this number to a much larger number than $4N$ to make sure the coalescent process will be complete (if set smaller, all remaining lineages will be coalesce to one lineage at the end of the coalescent process).

2.6 Testing the demographic simulation using the GUI version

It is imperative to check if all the files are set correctly and the demographic process makes sense by running a couple of examples using the GUI version on windows. Remember that all the files in the GUI version needs to have actual values instead of parameter tags. First load the *settings* file by clicking in **Open Settings**... button on the right top of the software panel. In the graphical interface only a restricted number of parameters can be modified (e.g. migration scenario, number of generation, generation time, start time, etc.), most of the parameters can only be set in the setting files (such as carrying capacities, frictions). For a complete explanation about all the parameters in the Settings File see Ray *et al.*, 2010).

`SPLATCHE2` is divided in two parts, demographic and coalescent simulations. For the demographic simulations, besides of the *settings files*, it is required that the following files be in the same working directory:

- **dens_init.txt**
- **dynamic_K.txt**
- **veg2K_cur.txt**
- **veg2K_int.txt**
- **veg2K_lgm.txt**
- **Arrival_cell.col**
- **oriworld.asc**

For the coalescent simulation, the following files are required:

- **genetic_data_SEQ.par**
- **GenSamples.sam**

In the end of the simulation, you will get an .arp file containing the simulated genetic data from the simulations in the folder of `GeneticOutput`. You will also be able to play the demographic simulations and inspect the expansion processes.

2.7 Generating Empirical Summary Statistics

The Approximate Bayesian Computing relies on comparing simulated summary statistics to the empirical ones. Using the `precheck.R` we can calculate all the pairwise ψ. Other traditional summary statistics can be calculated by various software, for example `ARLSUMSTAT`. The following files are needed to run `ARLSUMSTAT`:

- **arlsumstat executable**
- **settings file "arl_run.ars"**

This file must be created with the windows version of `Arlequin`. In the Windows version or `Arlequin`, open a project file of the type you want to analyze, choose the computations you want to perform, close the project and copy the generated file to the directory of the `arlsumstat` executable.

- **ssdefs.txt file**

Specifies which statistics are outputted in the result file.

- **the .*arp* file**

The *.vcf* file generated from Stacks pipeline transformed in `Arlequin` format (the `PGDSpider` software can be used for this).

The following command will analyze the file *.arp* (here called `filetransformed.arp`) and write the summary statistics in the file `ObservedSumstats.obs`.

```
./arlsumstat_64bit filetransformed.arp ObservedSumstats.obs 0 1
```

3 Running spatial simulations and the calculation of summary statistics

After the initial GUI testing, we can set up an Approximate Bayesian Computing pipeline using ABC`SAMPLER` to explore the parameter spaces and iterate over many realizations of the actual demographic simulation and corresponding calculation of summary statistics. ABC`SAMPLER` aims at producing a large collection of simulations, resulting in a matrix of model parameters and their associated summary statistics.

3.1 Settings for spatial simulation

Two input files are required: a master file (example: `toy_linux.input`) that specifies the processes of each simulation and a file (example: `toy.est`) that specifies the prior ranges of the all the parameters. There are three sections in the `.est` file: parameters, rules and complex parameters. Parameters specify the ones that will vary in each simulation within a certain range. Rules specify if any pair of parameters need to follow satisfy certain relationships. For example, in the `toy.est`, I specify that the Nanc*m>30 (i.e., at least 30 individuals will migrate out of the origin from the first generation of the forward demographic simulation) to make sure that by the end of the simulation, all the demes on the map have been colonized. Complex parameters specify the ones that will be

used in the actual simulation, but is calculated from the ones specified in the [parameters] section. For example, if we specify log_migration in the [parameters] section with a uniform distribution, we can write the actual migration parameter in the [complex parameters] as m = pow10(log_migration). More details see ABCSAMPLER manual. In this example, Migration (MRATE), Ancestral Population size (N_ANCESTRAL), Carrying Capacities (K) and the original ancestral population coordinates (oriLat, oriLon) are the parameters used in the simulation. In each iteration, ABCSAMPLER will look for the parameter names in the two parameter sections in all the input files for the demographic simulation, and replace them with actual random numbers sampled from the prior ranges.

The mast input file (`toy_linux.input`) specifies all the input files and software used in the simulation processes. *estName* specifies the location of the parameter file. *obsName* is the location of the empirical observation of summary statistics file. *outName* specifies the prefix of output files. *nbSims* is the number of simulations. *SimulationProgram* specifies the location of demographic simulation software. In this case, it is the SPLATCHE2.

SimInputName specifies the input parameters for running SPLATCHE2. All the input files that need to have parameter name placeholder changed to actual values should be typed here and separated by "#" between the files.

```
simInputName ./splatche2input/1-settings.txt#./splatche2input/2-
dens_init.txt#./splatche2input/pastK.txt#./splatche2input
/presentK.txt
```

ABCSAMPLER will generate a `temp` file for each of the files specified on this line. These `[originalFileName]-temp.txt` files will be used for the current iteration with a specific set of parameter values. For example, MRATE in `1-settings.txt` file will be replaced by an actual value between 0.001 and 0.01 in the `1-settings-temp.txt` file.

3.2 Settings for summary statistics

For the summary statistics calculation programs, we will use the `calSumStat.py` script to calculate all the statistics. In this example, we have the following two lines:

```
sumStatProgram /usr/bin/python
sumStatParam calSumStat/calSumStat.py#splatche2input#5-
Arrival_cell_output.txt#1-settings-temp_6-GenSamples_1.arp#6-
GenSamples.sam#calSumStat
```

`calSumStat.py` needs 5 input parameters: splatche Running Folder (`splatche2input`); the output file from demographic simulation that tells at which generation each population was colonized (`5-Arrival_cell_output.txt`); the name of the `.arp` file (`1-settings-temp_6-GenSamples_1.arp`); the sampling locations(`6-GenSamples.sam`);

and the folder of calculated summary statistics(`calSumStat`). `calSumStat.py` checks if all the sampled populations are colonized at the end of the simulation. If not, then all the statistics will be recorded as "`-9999`". If all the sampled populations are colonized, then summary statistics are calculated with both `arlsumstat` and pairwise ψ.

3.3 Running ABCsampler

To run the ABC`SAMPLER`, type:

```
./ABCsampler toy_linux.input
```

All the summary statistics for the simulated scenario are recorded in the output file: `[outName]_sampling1.txt`.

4 Inferences of the expansion origin and validation processes

After all the simulations are done, we have summary statistics that are calculated on each simulated data. First, we need filter out all the rows with "`-9999`", because these simulations are the ones that failed to colonize sampled populations. We then need to convert and extract principal components from the original calculated statistics to reduce the curse of high dimensionality. We can then calculated the distances of simulated cases towards the empirical cases based on the PCs and leave the simulations that are closest to the empirical scenarios. We then run validations using pseudo-observation datasets (PODs). All the scripts required for these analyses are included in the folder of "`postSimulationProcessing`".

4.1 Script for converting statistics and extracting principal components:

```
Rscript find_pca.r [outName]_sampling1.txt [obsName]
```

line 15, 17 and 19, 20, 24 need to be changed to reflect the actual column number of the parameters or statistics.

4.2 estimation of the origins

The input file for `ABCestimator` needs to be specified. Important field includes the number of simulation retained to estimate the origin (numRetained). simName, obsName specifies

simulated and the empirical summary statistics file. outputPrefix specifies the prefix of the output files.

```
./ABCestimator toyEmpiricalGLM.input >toyEmpirical_marginalDensity.txt
```

The output file `toyEmpEvalBestSimsParamStats_Obs0.txt` includes the distance calculation of the all the retained simulation to the empirical scenario based on the principal components. We then use the `XOriginInfer_empirical.R` to infer the actual origin.

```
Rscript XOriginInfer_empirical.R toyEmpEvalBestSimsParamStats_Obs0.txt
```

4.3 validation runs using pseudo-observation datasets (PODs)

We can use 5000 simulations that are not included in the previous analyses to validate the overall accuracy of the estimation of the origin. Similarly, we can use previously generate EigenVector file to transform the PODs summary statistics by running `generatePods.R`. We then run the `ABCestimator` again using the `toyPODsGLM.input`. Note that the input file has an new field called trueParamName. This field records all the actual parameter values for the 5000 simulations. We then run `XOriginInfer_PODs.R` to get the evaluation of the estimated origins by calculating the Euclidean distances between the estimated origins and the actual origins.