iDiv

Making Sense of High-Throughput Sequencing (short read) Data

28-29 October 2019

Chris Barratt - Flexpool postdoc at iDiv in Evolution and Adaptation (PI: Renske Onstein) and Sustainability and Complexity in Ape Habitat (PI: Hjalmar Kuehl)

What are we doing today?

- We already did a workshop in May based on the background, sampling design and bioinformatic processing of RAD-seq data with a HPC cluster (little refresher first)

➢ Your learning outcomes from this course are how to run and interpret 6 independent downstream analyses to make sense of high throughput short-read data:

- Population structure (Admixture, DAPC)
- Evolutionary relationships (RAxML, SNAPP)
- Landscape barriers and population connectivity (EEMS)
- Historical demographic inference and hypothesis testing (∂a∂i)

Why these programs?

- The programs we will use are amongst the most common representatives for kinds of analyses which can be performed with many other pieces of published code
- These are not exactly user friendly, but they are tried and tested, and are the emergent types of analyses which have stuck around after the NGS methods explosion of the last 10 years or so
- Perhaps most importantly they are well documented and still maintained for when you run into a problem
- These analyses are very up to date as of 2019, but things change quickly and new methods are always coming

First things first…

- We need to access the UFZ cluster (terminal or puTTy)
- We need a good FTP client (I recommend CyberDuck but FileZilla also fine) and notepad style editor (TextWrangler/ Bbedit for Mac or notepad++ for windows)
- Install Rstudio, BEAST, PGDspider on your laptop/mac

- We need to build our own python and bioconda environments on the cluster for running some of the analyses (ca. 2 mins using the **create_environments.sh** script)

- We need to make sure our directory structure is fine. Easiest I suggest is to do **scp –r /public/barratt/ /public/$USER/**

> Then any submit script you simply need to change 'barratt' for your account name

Directory structure on cluster (in /public/$USER/)

- **submit_scripts** (all scripts to process the data)
- **work** (working folder)
- **virtual_env_RADseq** (virtual environment we created)

If you run this command from your terminal it will download all the files for completed analyses:

rsync --dry-run -avhP -e 'ssh -p 8022' -r barratt@localhost:/public/barratt/ /Users/chris/Desktop/Workshop_2/

Example data we are working with

Leptopelis flavomaculatus –
Yellow spotted treefrog (n=59) –
originally from 42.3 GB of data

Processed output files are just a few hundred MB

Received: 20 February 2018 | Revised: 8 August 2018 | Accepted: 29 August 2018

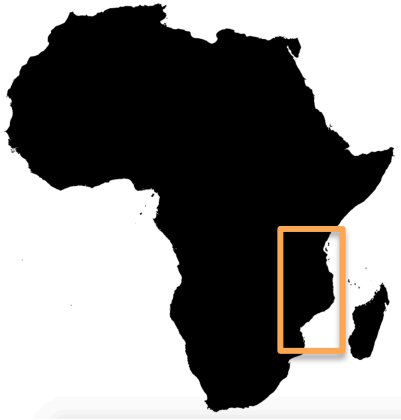
DOI: 10.1111/mec.14862

ORIGINAL ARTICLE

WILEY MOLECULAR ECOLOGY

Vanishing refuge? Testing the forest refuge hypothesis in coastal East Africa using genome wide sequence data for seven amphibians

Christopher D. Barratt[1,2] | Beryl A. Bwong[1,3] | Robert Jehle[4] | H. Christoph Liedtke[1,5] | Peter Nagel[1] | Renske E. Onstein[2] | Daniel M. Portik[6,7] | Jeffrey W. Streicher[8] | Simon P. Loader[1,8]



Input files we need to do the analyses

- Because processing RAD-seq data is so processor and time intensive I pre-prepared the data so it is ready to analyse − it has already been analysed by Stacks so we are working with the output files from there

- These datasets use only single SNPs from each RAD locus which is present in at least 50% of all samples (most samples have only a little missing data), >5x coverage and with a minor allele frequency >0.05, which gives 8598 SNPs

Population structure

- = the patterns in neutral variation that result in departure from panmixia
- Understanding population structure tells us something about past and present changes in migration regimes
- This information is a basis for understanding which evolutionary or environmental factors are acting upon species
- Essential for understanding how evolution operates at smaller scales

Population structure - Admixture

- A good tool for looking at population structure in genomic data is with Admixture (Alexander et al. 2009)

GENOME RESEARCH
CSHL Press | Journal Home | Subscriptions | eTOC Alerts | BioSupplyNet

Genome Res. 2009 Sep; 19(9): 1655–1664.
doi: 10.1101/gr.094052.109
PMCID: PMC2752134
PMID: 19648217

Fast model-based estimation of ancestry in unrelated individuals

David H. Alexander,[1,4] John Novembre,[2] and Kenneth Lange[3]

[1]Department of Biomathematics, University of California at Los Angeles, Los Angeles, California 90095, USA; [2]Department of Ecology and Evolutionary Biology, University of California at Los Angeles, Los Angeles, California 90095, USA; [3]Department of Human Genetics and Department of Statistics, University of California at Los Angeles, Los Angeles, California 90095, USA

Population stratification has long been recognized as a confounding factor in genetic association studies. Estimated ancestries, derived from multi-locus genotype data, can be used to perform a statistical correction for population stratification. One popular technique for estimation of ancestry is the model-based approach embodied by the widely applied program *structure*. Another approach, implemented in the program EIGENSTRAT, relies on Principal Component Analysis rather than model-based estimation and does not directly deliver admixture fractions. EIGENSTRAT has gained in popularity in part owing to its remarkable speed in comparison to *structure*. We present a new algorithm and a program, ADMIXTURE, for model-based estimation of ancestry in unrelated individuals. ADMIXTURE adopts the likelihood model embedded in *structure*. However, ADMIXTURE runs considerably faster, solving problems in minutes that take *structure* hours. In many of our experiments, we have found that ADMIXTURE is almost as fast as EIGENSTRAT. The runtime improvements of ADMIXTURE rely on a fast block relaxation scheme using sequential quadratic programming for block updates, coupled with a novel quasi-Newton acceleration of convergence. Our algorithm also runs faster and with greater accuracy than the implementation of an Expectation-Maximization (EM) algorithm incorporated in the program FRAPPE. Our simulations show that ADMIXTURE's maximum likelihood estimates of the underlying admixture coefficients and ancestral allele frequencies are as accurate as *structure*'s Bayesian estimates. On real-world data sets, ADMIXTURE's estimates are directly comparable to those from *structure* and EIGENSTRAT. Taken together, our results show that ADMIXTURE's computational speed opens up the possibility of using a much larger set of markers in model-based ancestry estimation and that its estimates are suitable for use in correcting for population stratification in association studies.

[Supplemental material is available online at http://www.genome.org. The ADMIXTURE program is freely available at http://www.genetics.ucla.edu/software.]

Population structure - Admixture

- ML estimation of individual ancestries from SNP datasets
- Uses same statistical model as Structure (Pritchard, Stephens & Donnelly, 2000) − the benchmark for understanding pop structure the past 20 years
- Uses a block relaxation approach for speed, which outperforms the other methods − they simply become inefficient and intractable for large SNP data

Population structure - Admixture

- Using the **run_Admixture.sh** script we run the algorithm for a range of k values on the dataset

- Admixture needs a .bed file (converted by plink), and uses cross validation scores to give you the most likely number of clusters for your data

```
####### UFZ cluster job submit #########

#!/bin/bash
#$ -N Admixture
#$ -M christopher_david.barratt@uni-leipzig.de
#$ -m beas
#$ -wd /public/barratt/
#$ -S /bin/bash
#$ -o /public/barratt/work/Lflavomaculatus/job_logs/$JOB_NAME-$JOB_ID.log
#$ -j y
#$ -pe smp -1
#$ -l h_vmem=10G
#$ -l highmem
#$ -l h_rt=1:00:00

#get slot information etc. (determine no. of slots with -pe above (can also be a range so that it takes the maximum available slots but will settle for less))
echo "Job name: $JOB_NAME"
echo "$NSLOTS slots have been dedicated to this job"
echo "On following node(s)"
cat $PE_HOSTFILE
echo "pe host file is $PE_HOSTFILE"

#load environment
module load miniconda/3
source activate conda_env_RADseq
source /public/barratt/virtual_env_RADseq/bin/activate

#load modules, start job
module load admixture/1.3.0-1

# Lflavomaculatus
cd /public/barratt/work/Lflavomaculatus/Admixture
# redo conversion using plink just to be sure...
plink --file /public/barratt/work/Lflavomaculatus/converted_files/Admixture/Lflav --make-bed --recode --out /public/barratt/work/Lflavomaculatus/converted_files/Admixture/Lflav
#run admixture
bash
for K in 1 2 3 4 5 6 7 8 9 10; \
do admixture --cv=10 ./Lflav.bed $K| tee log_Lflav.${K}.out; done
grep -h CV log_Lflavomaculatus*.out > ./RESULTS_Lflavomaculatus_admixture.txt
```

qsub shell script in more detail

- Some header information is needed to tell the cluster how to process your jobs, how much resources to use, where to send notifications etc.

- See https://wiki.ufz.de/eve/index.php/Submitting_Jobs for much more detailed information

qsub_script_meanings.sh

~/Downloads/qsub_script_meanings.sh

```
1  ####### UFZ cluster job submit #########                ###### Standard header of script
2
3  #!/bin/bash                                             ###### this line tells the UFZ cluster it is a bash script
4  #$ -N process_radtags                                   ###### the name of the job (also appears when you check qstat and the email notifications from the cluster)
5  #$ -M christopher_david.barratt@uni-leipzig.de          ###### the email address where notifications are to be sent
6  #$ -m beas                                              ###### begin end abort stop - these notifications will be sent to above email
7  #$ -wd /work/$USER                                      ###### your working directory (this can stay as it is)
8  #$ -S /bin/bash                                         ###### again, some bash information
9  #$ -o /work/$USER/Stacks/Afornasini/job_logs/$JOB_NAME-$JOB_ID.log   ###### output log file location - this is important to get right
10 #$ -j y                                                 ###### whether to do job logs? (y/n)
11 #$ -pe smp -1                                           ###### how many processor cores requested
12 #$ -l h_vmem=20G                                        ###### how much memory per core requested (max 100G)
13 #$ -l highmem                                           ###### high memory flag (useful for big memory jobs, I use this most of the time)
14 #$ -l h_rt=24:00:00                                     ###### how long job can run? (hh:mm:ss) (max 720:00:00 = 1 month)
15
16 #load environment                                       ###### here is where you load the environment if you have special software you need which is not on the main cluster
17 source ~/barratt_env/bin/activate
18
19 #load modules, start job                                ###### here is where you change directory to the directory you are working in, and load modules you need
20 cd /work/$USER/Stacks/Afornasini/
21 module load stacks/1.46-1
22 module load perl/5.16/3-1
23
24 # after this would come your main code - e.g. Stacks commands etc...
```

Population structure - Admixture

- We submit this on the cluster – so cd to your submit scripts directory and type qsub run_Admixture.sh followed by enter. This is a relatively small dataset so will only take a couple of mins to run

```
barratt@frontend1 ~ $ cd /public/barratt/submit_scripts/
barratt@frontend1 /public/barratt/submit_scripts $ qsub run_Admixture.sh
Your job 5720982 ("Admixture") has been submitted
barratt@frontend1 /public/barratt/submit_scripts $
```

- When it has finished you will receive an email notification (if you edited your scripts to change the email notifications)

Population structure - Admixture

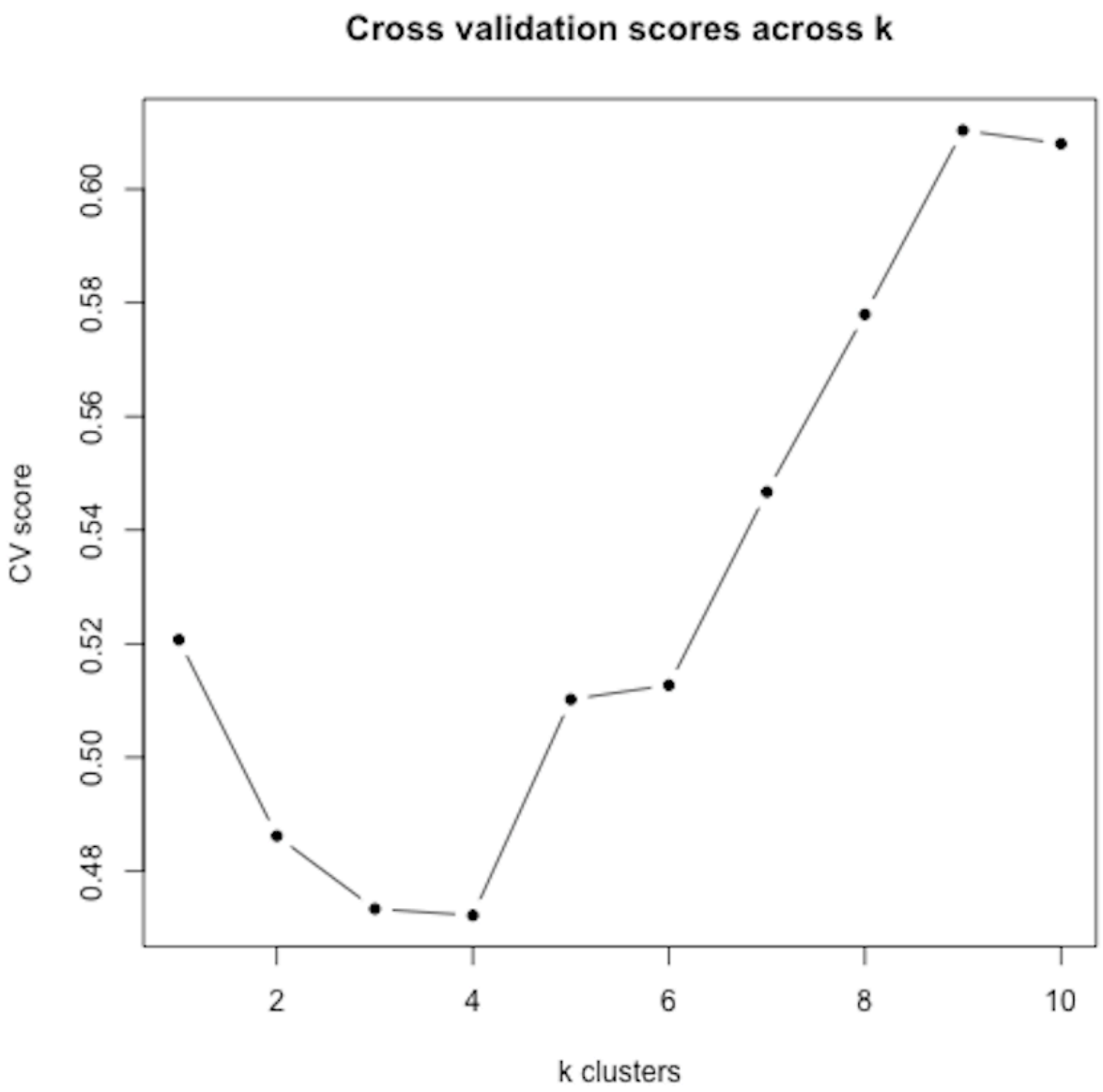
- The job log file (in /public/$USER/work/Lflavomaculatus/job_logs) contains the output from the Admixture run, have a look at it

- Within the script we already used grep on the last line to grab the CV values for each iteration of k and send them to an output file. So in /public/$USER/work/Lflavomaculatus/Admixture/ there is a RESULTS_Lflavomaculatus_admixture.txt file

- Let's visualise that in R using the **1_PLOT_ADMIXTURE.R** script. To get the file on your computer you can either drag and drop from your FTP client (Cyberduck or FileZilla) or use rsync:

- rsync --dry-run -avhP -e 'ssh -p 8022' -r barratt@localhost:/public/barratt/ /Users/chris/Desktop/Workshop_2/

Population structure - Admixture



Population structure - Admixture

Cross-validation

The choice of the number of ancestral populations K can prove difficult when the underlying population genetics of a species is poorly understood. STRUCTURE provides a means of estimating the best value of K by computing the *model evidence* for each K from a range of choices. The model evidence is defined as

$$\Pr(N|K) = \int f(N|Q, F, K)\, \pi(Q, F|K)\, dQ\, dF \qquad (2)$$

where f represents the data likelihood and π represents a prior density on the parameters. STRUCTURE approximates the integral via Monte Carlo methods. Our optimization framework is not well suited to evaluating this integral. As an alternative, we employ cross-validation. In cross-validation, we aim to identify the best K value as judged by prediction of systematically withheld data points. A similar tactic is also employed by the haplotype analysis program fastPHASE [5] and is inspired by Wold's method for cross-validating PCA models [6].

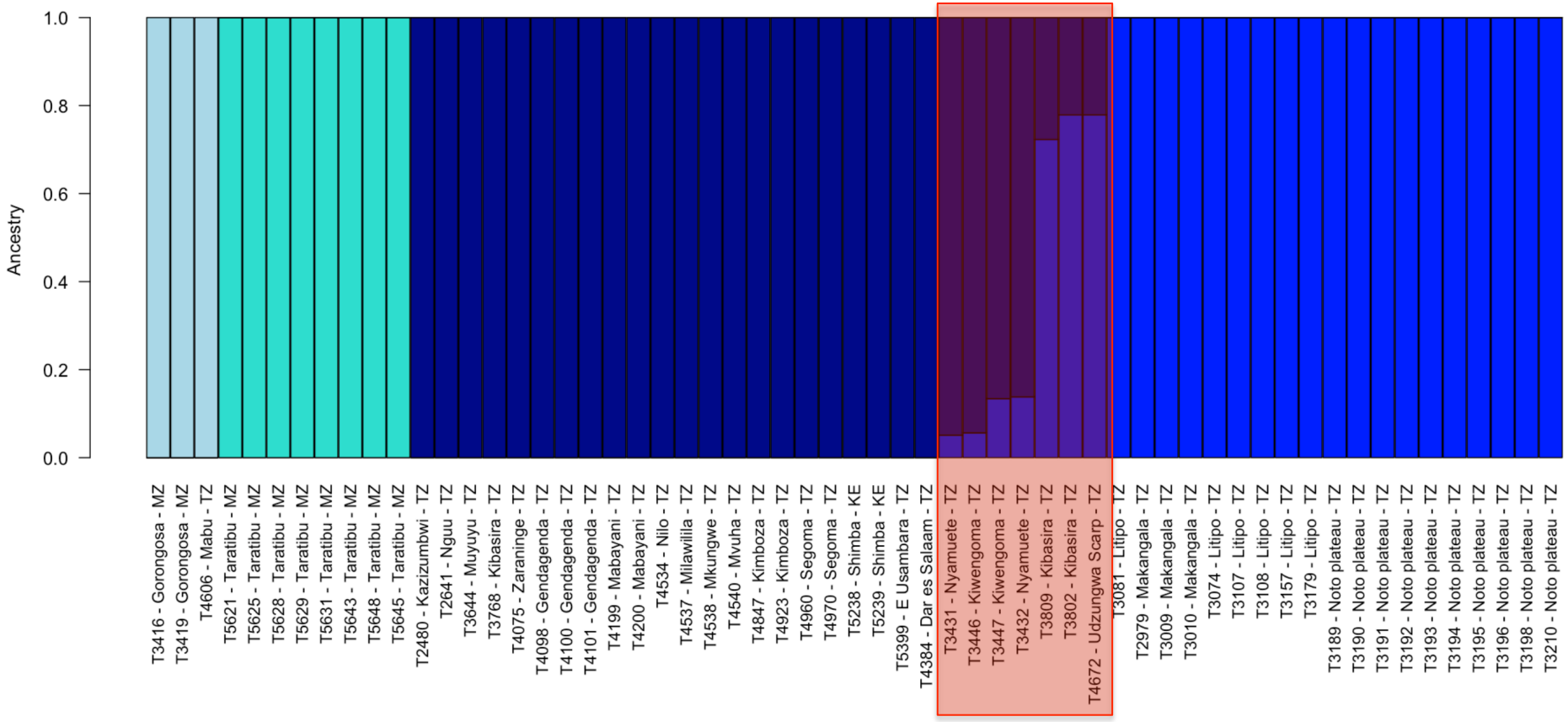
Our v-fold cross-validation procedure partitions the non-missing genotypes into v roughly equally sized subsets (*folds*). At each of v iterations, the members of one of the folds are masked (temporarily marked as missing) to yield a new data matrix $\tilde{N} = \{\tilde{n}_{ij}\}$. Analysis of the masked data matrix $\tilde{N}$ poses no new challenges. In computing the log-likelihood, score, and observed information matrix of $\tilde{N}$, we simply ignore the entries (i, j) with missing values. Maximization of the log-likelihood readily yields new estimates $\tilde{Q}$ and $\tilde{F}$ for the masked data. We then predict each masked value n_{ij} by $\hat{\mu}_{ij} = 2 \sum_k \tilde{q}_{ik} \tilde{f}_{kj}$. Prediction error is estimated by averaging the squares of the deviance residuals for the binomial model [7],

$$d(n_{ij}, \hat{\mu}_{ij}) = n_{ij} \log(n_{ij}/\hat{\mu}_{ij}) + (2 - n_{ij}) \log[(2 - n_{ij})/(2 - \hat{\mu}_{ij})], \qquad (3)$$

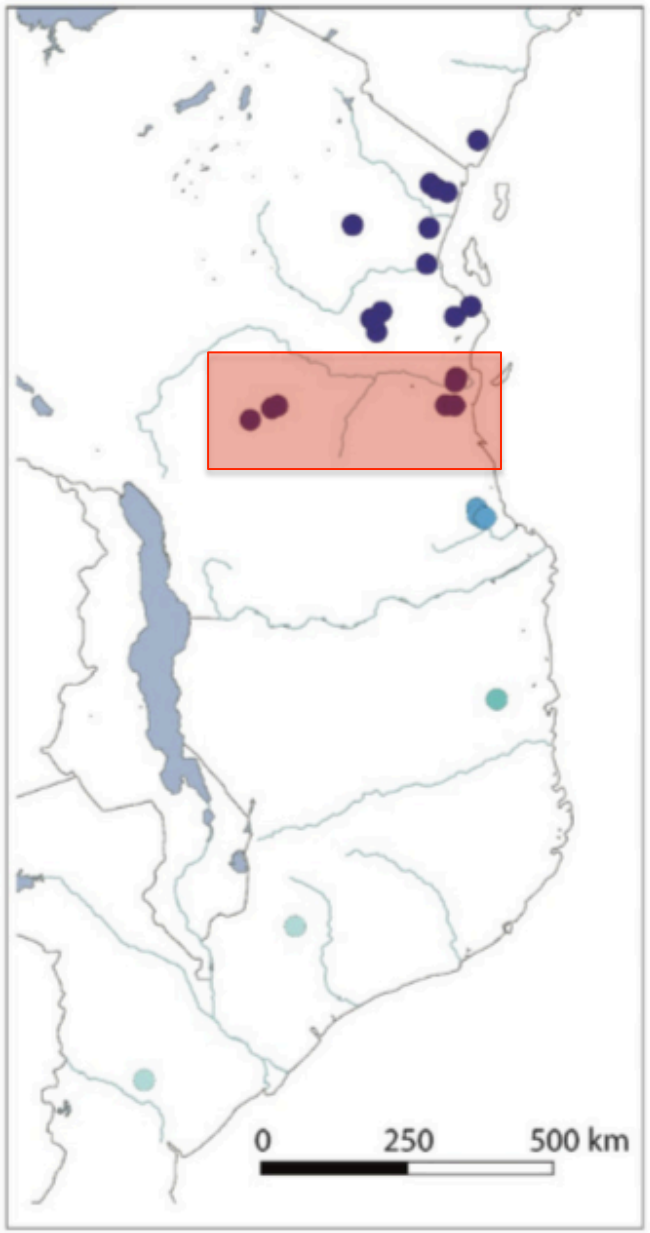
across all masked entries over all folds. Minimizing this estimated prediction error on a grid of K values then suggests the most suitable K.

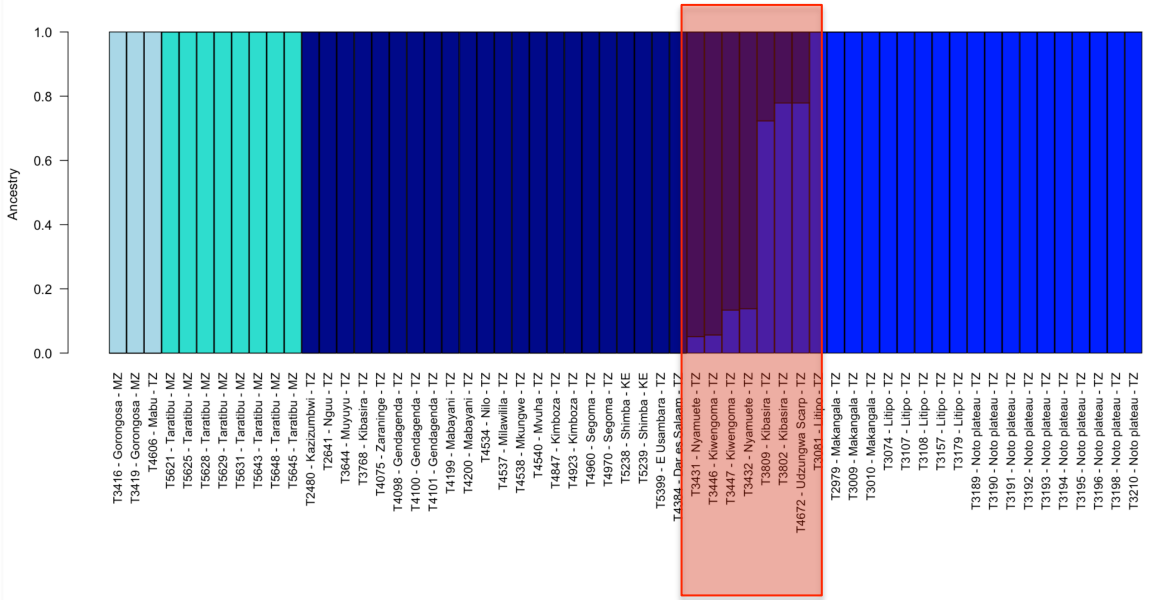
Population structure - Admixture

- So k=4 seems to be a fairly good description of the structure for this data, population structure is fairly straightforward to interpret in this case
- Note that we do have some admixture between some pops!



Population structure - DAPC





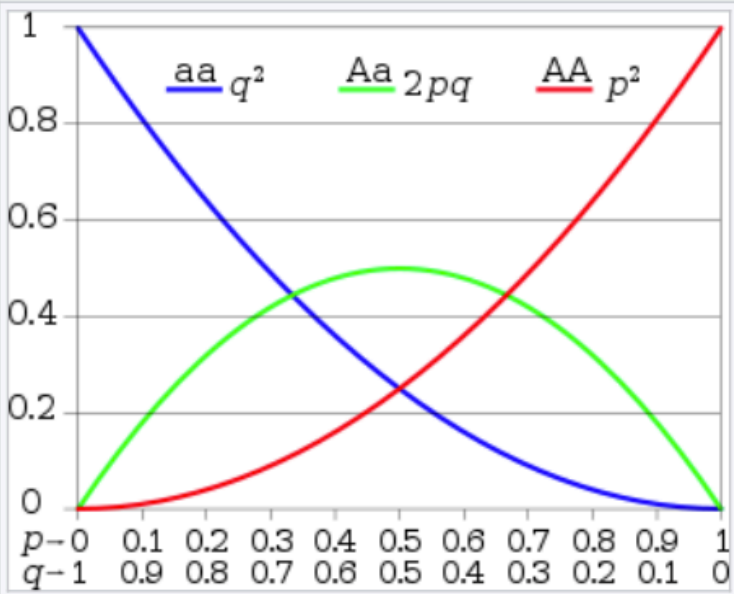
Population structure - DAPC

- What is DAPC? Discriminant function Analysis of Principal Components (Jombart et al. 2010) − a multivariate method

- Admixture is great, but it can run into problems if your loci are not in Hardy-Weinberg equilibrium, which may not be the case especially if you have small population sizes. Best to try a few other methods which can give your more confidence in your population structure analyses…

- DAPC is one such method free from HWE assumptions, and the algorithm works slightly differently so it is independent from Admixture. A good tutorial can be found here (we will do an exercise modified from this)

Population structure - DAPC



Hardy–Weinberg proportions for two alleles: the horizontal axis shows the two allele frequencies *p* and *q* and the vertical axis shows the expected genotype frequencies. Each line shows one of the three possible genotypes.

- Hardy-Weinberg principle:
- Allele and genotype frequencies will remain constant from generation to generation without external evolutionary influences
- Selection
- Mutation
- Gene flow (migration)
- Genetic drift random changes (e.g. small population sizes)

Population structure - DAPC

Jombart *et al. BMC Genetics* 2010, **11**:94
http://www.biomedcentral.com/1471-2156/11/94

BMC Genetics

METHODOLOGY ARTICLE Open Access

Discriminant analysis of principal components: a new method for the analysis of genetically structured populations

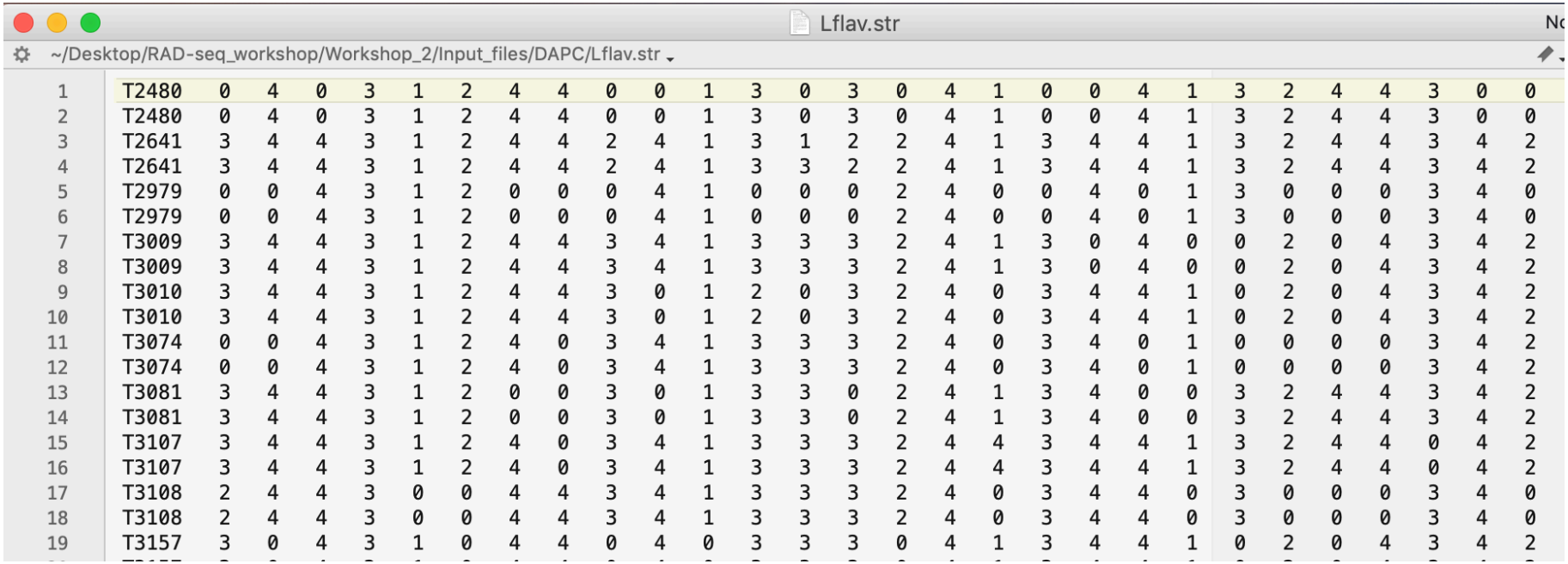
Thibaut Jombart[1*], Sébastien Devillard[2], François Balloux[1*]

Abstract

Background: The dramatic progress in sequencing technologies offers unprecedented prospects for deciphering the organization of natural populations in space and time. However, the size of the datasets generated also poses some daunting challenges. In particular, Bayesian clustering algorithms based on pre-defined population genetics models such as the STRUCTURE or BAPS software may not be able to cope with this unprecedented amount of data. Thus, there is a need for less computer-intensive approaches. Multivariate analyses seem particularly appealing as they are specifically devoted to extracting information from large datasets. Unfortunately, currently available multivariate methods still lack some essential features needed to study the genetic structure of natural populations.

Population structure - DAPC

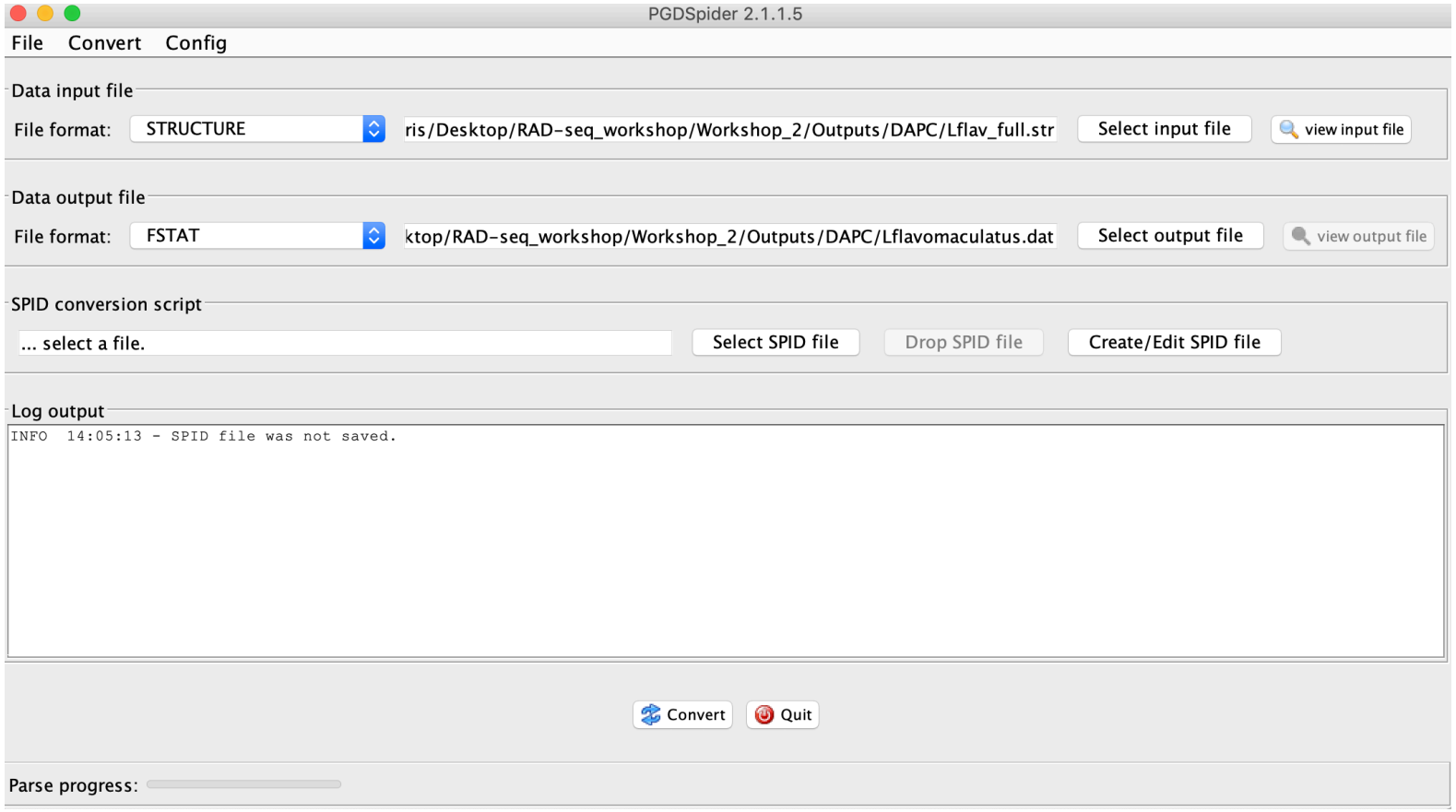
- For DAPC population stucture analyses the .str file output from Stacks contains all the info we need. However we can delete the first 2 lines output by Stacks (they are not necessary and other programs don't like it)



1	T2480	0	4	0	3	1	2	4	4	0	0	1	3	0	3	0	4	1	0	0	4	1	3	2	4	4	3	0	0
2	T2480	0	4	0	3	1	2	4	4	0	0	1	3	0	3	0	4	1	0	0	4	1	3	2	4	4	3	0	0
3	T2641	3	4	4	3	1	2	4	4	2	4	1	3	1	2	2	4	1	3	4	4	1	3	2	4	4	3	4	2
4	T2641	3	4	4	3	1	2	4	4	2	4	1	3	3	2	2	4	1	3	4	4	1	3	2	4	4	3	4	2
5	T2979	0	0	4	3	1	2	0	0	0	4	1	0	0	0	2	4	0	0	4	0	1	3	0	0	0	3	4	0
6	T2979	0	0	4	3	1	2	0	0	0	4	1	0	0	0	2	4	0	0	4	0	1	3	0	0	0	3	4	0
7	T3009	3	4	4	3	1	2	4	4	3	4	1	3	3	3	2	4	1	3	0	4	0	0	2	0	4	3	4	2
8	T3009	3	4	4	3	1	2	4	4	3	4	1	3	3	3	2	4	1	3	0	4	0	0	2	0	4	3	4	2
9	T3010	3	4	4	3	1	2	4	4	3	0	1	2	0	3	2	4	0	3	4	4	1	0	2	0	4	3	4	2
10	T3010	3	4	4	3	1	2	4	4	3	0	1	2	0	3	2	4	0	3	4	4	1	0	2	0	4	3	4	2
11	T3074	0	0	4	3	1	2	4	0	3	4	1	3	3	3	2	4	0	3	4	0	1	0	0	0	0	3	4	2
12	T3074	0	0	4	3	1	2	4	0	3	4	1	3	3	3	2	4	0	3	4	0	1	0	0	0	0	3	4	2
13	T3081	3	4	4	3	1	2	0	0	3	0	1	3	3	0	2	4	1	3	4	0	0	3	2	4	4	3	4	2
14	T3081	3	4	4	3	1	2	0	0	3	0	1	3	3	0	2	4	1	3	4	0	0	3	2	4	4	3	4	2
15	T3107	3	4	4	3	1	2	4	0	3	4	1	3	3	3	2	4	4	3	4	4	1	3	2	4	4	0	4	2
16	T3107	3	4	4	3	1	2	4	0	3	4	1	3	3	3	2	4	4	3	4	4	1	3	2	4	4	0	4	2
17	T3108	2	4	4	3	0	0	4	4	3	4	1	3	3	3	2	4	0	3	4	4	0	3	0	0	0	3	4	0
18	T3108	2	4	4	3	0	0	4	4	3	4	1	3	3	3	2	4	0	3	4	4	0	3	0	0	0	3	4	0
19	T3157	3	0	4	3	1	0	4	4	0	4	0	3	3	3	0	4	1	3	4	4	1	0	2	0	4	3	4	2

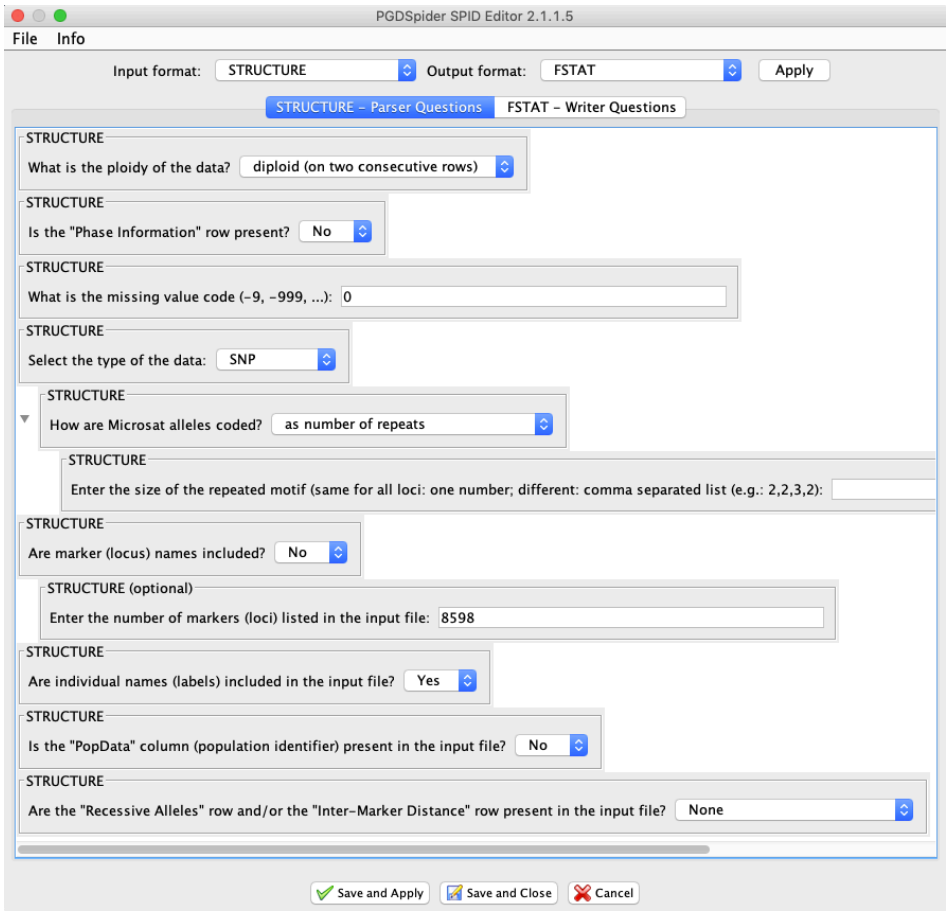
Population structure - DAPC

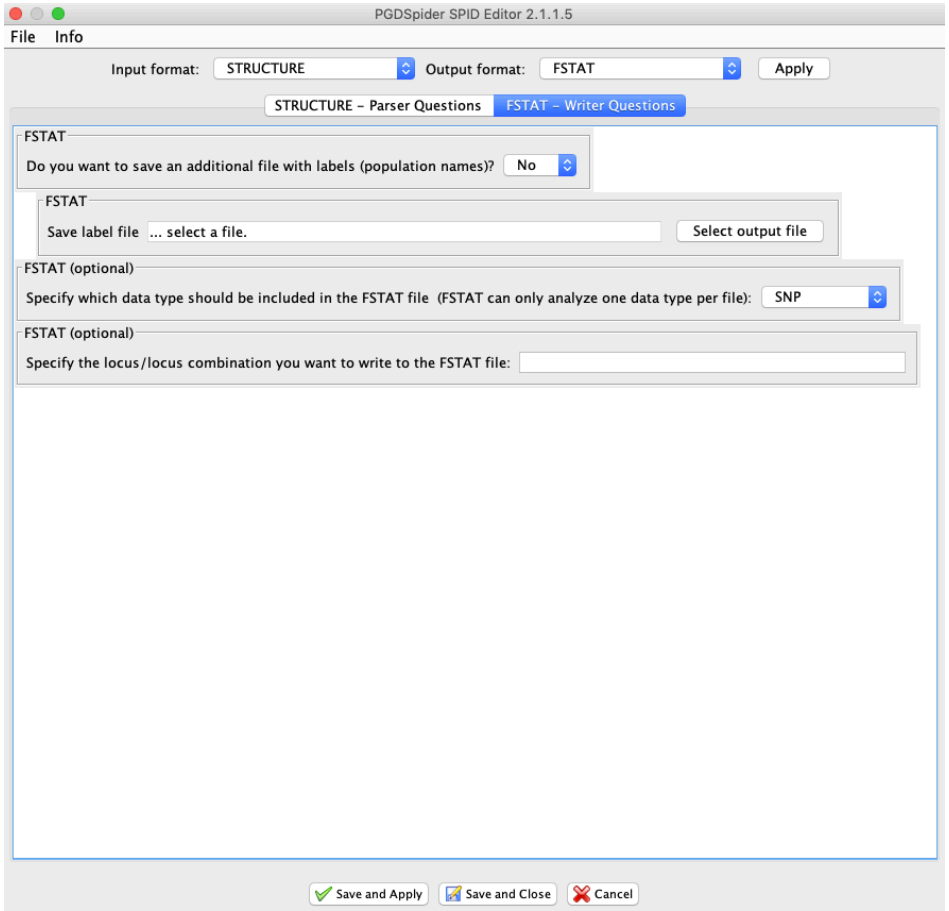
- We can use PGDspider2.jar to convert the files we need (from STRUCTURE to FSTAT for DAPC)



Population structure - DAPC

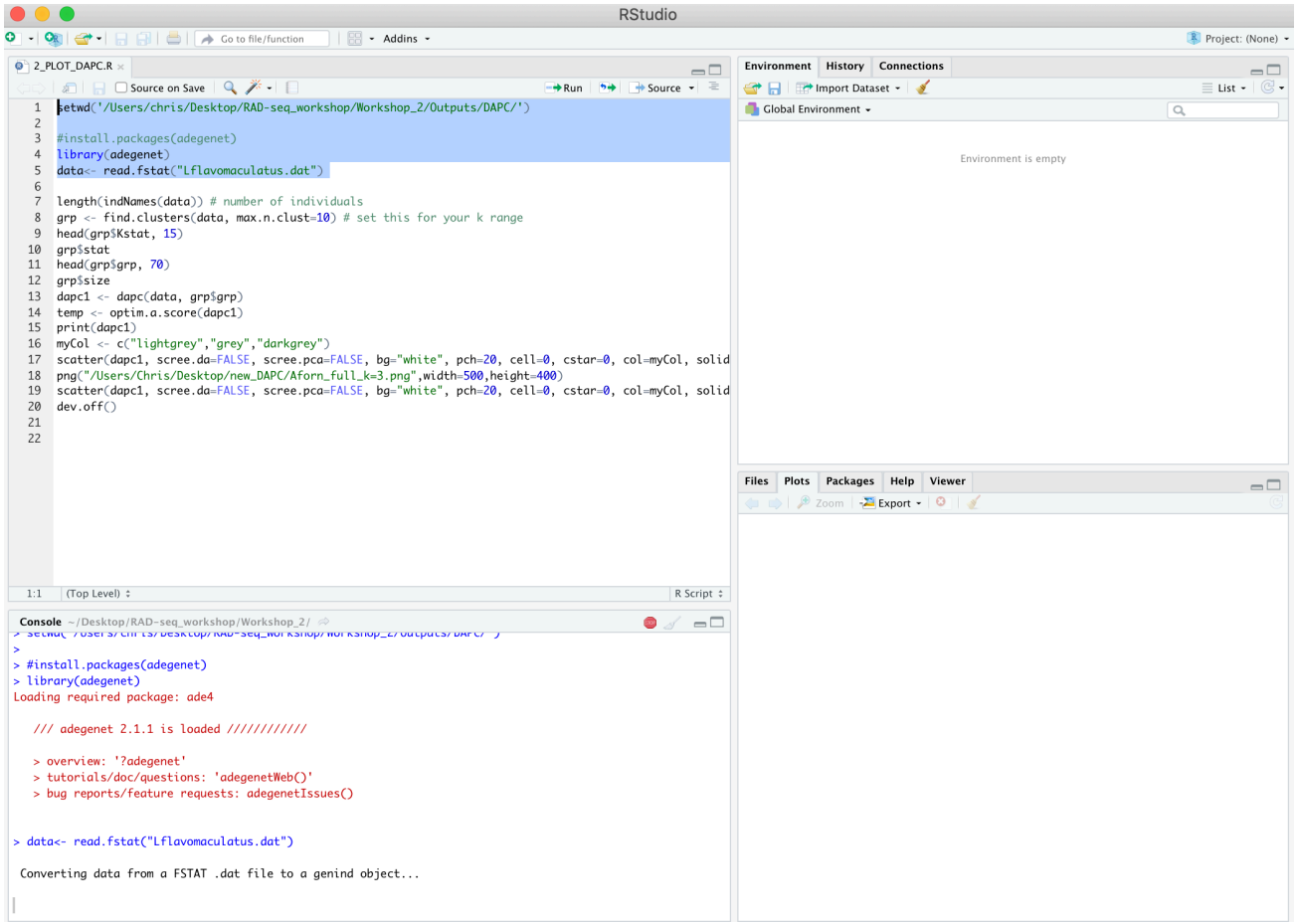
- We can use PGDspider2.jar to convert the files we need (from STRUCTURE to FSTAT for DAPC)





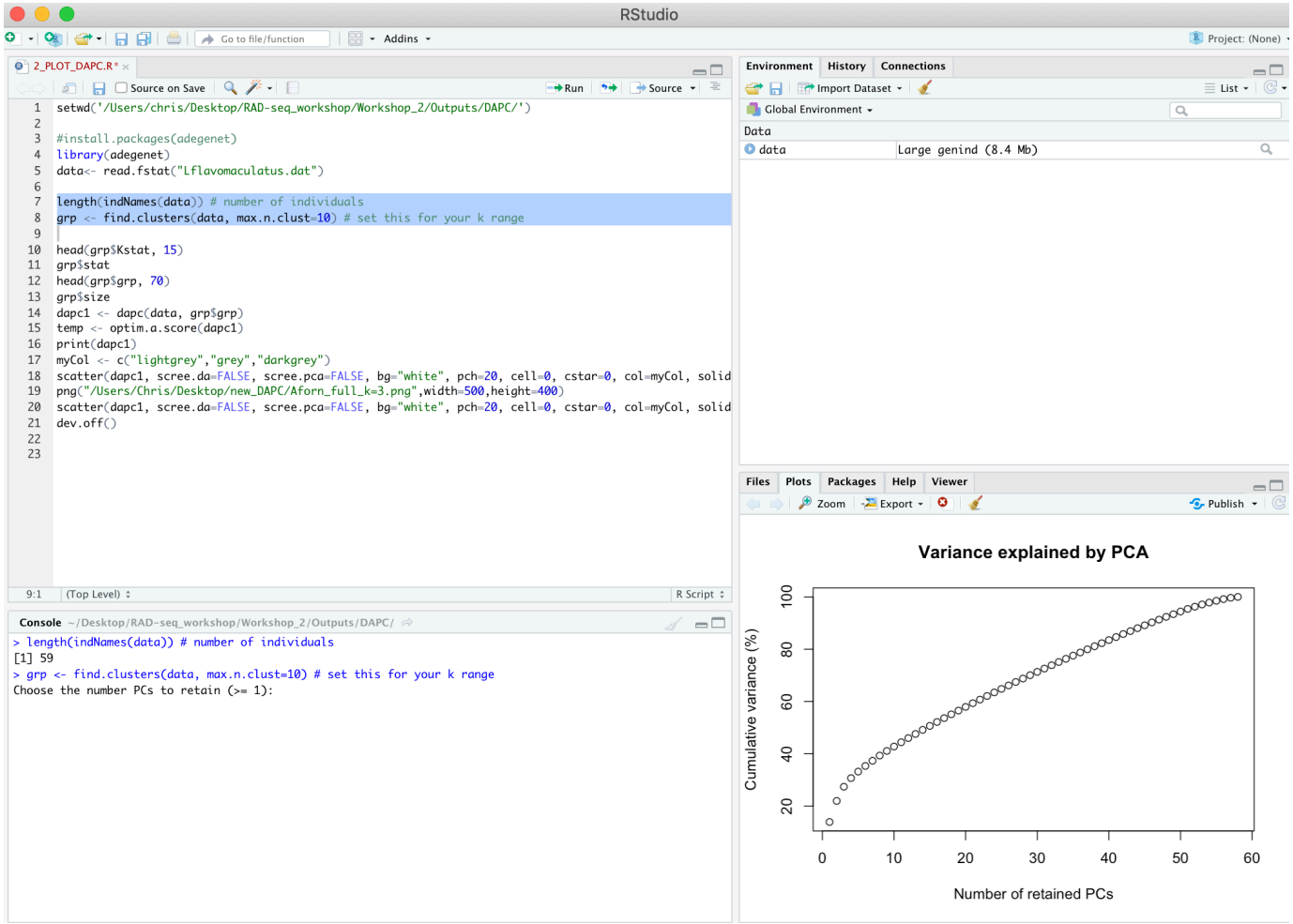
Population structure - DAPC

- Now with the resulting .dat file we can run DAPC with the script **2_RUN_AND_PLOT_DAPC.R** - Run the first 5 lines (this will take a few mins)



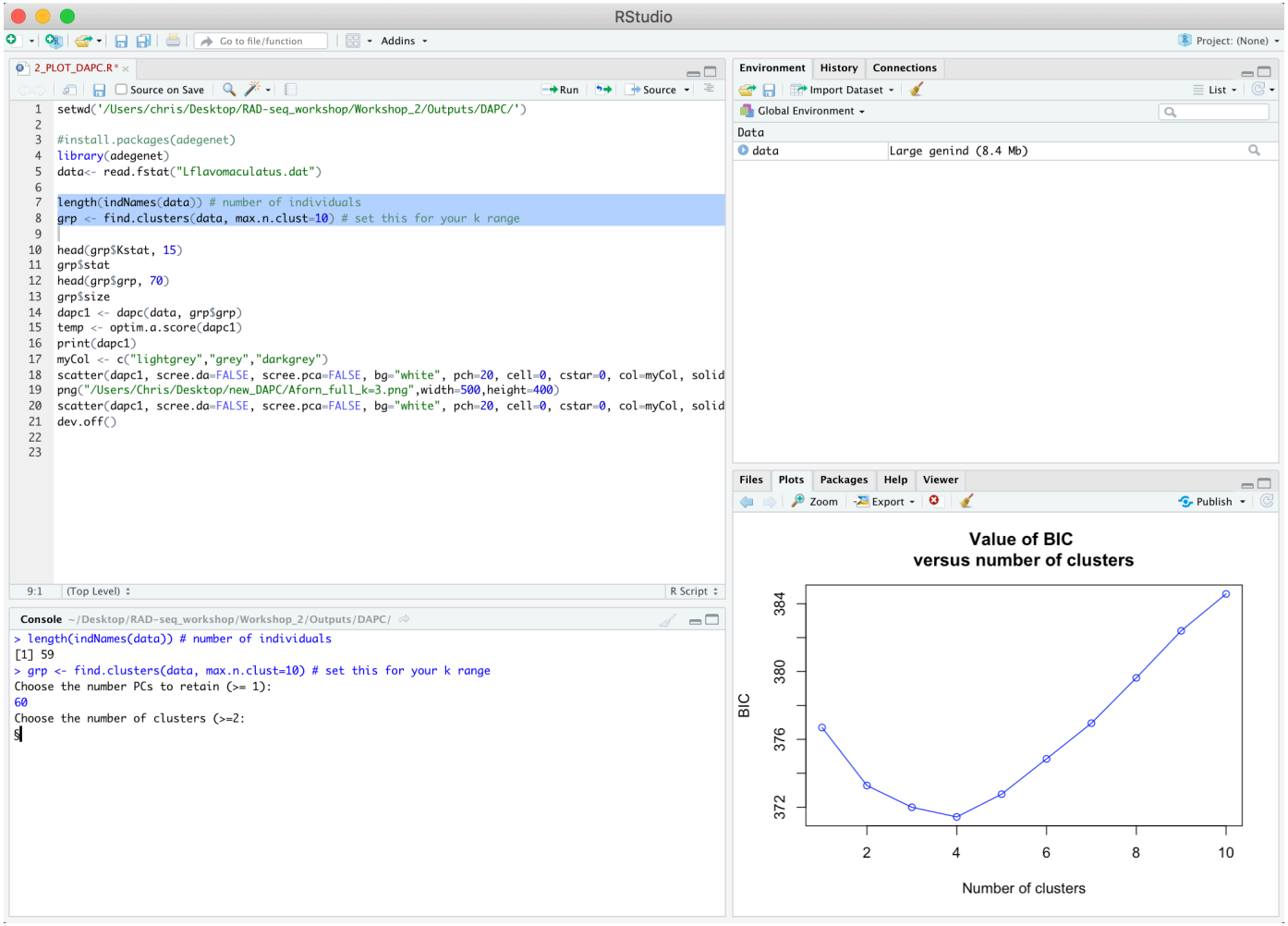
Population structure - DAPC

- Enter the value where line starts to level out (in this case at the end)



Population structure - DAPC

- Enter how many clusters are present based on the lowest BIC score (Bayesian Information Criterion)



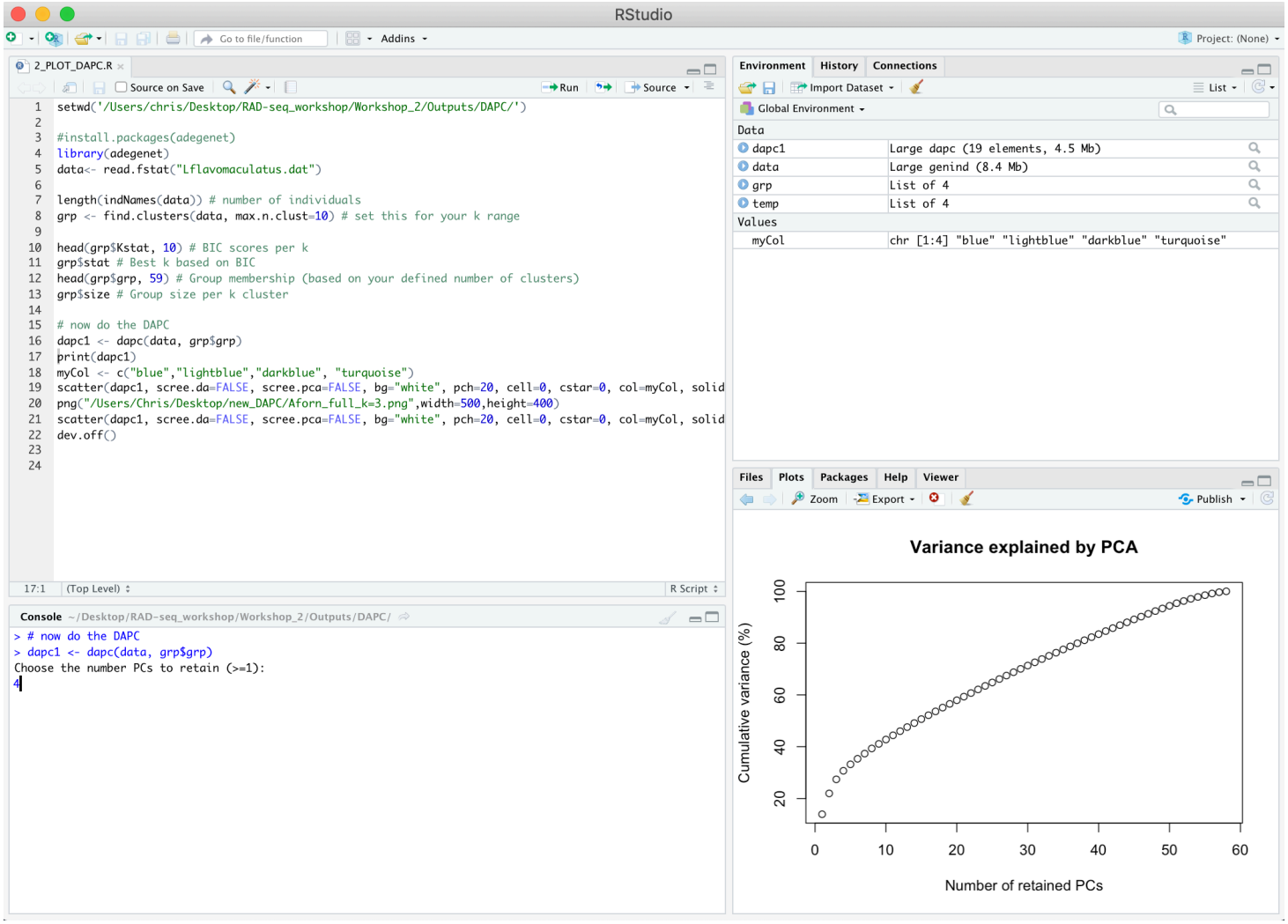
Population structure - DAPC

- Lines 10-13 will give you some info about the data including accurate BIC scores, best k chosen, cluster membership and cluster sizes

```
> head(grp$Kstat, 10) # BIC scores per k
     K=1      K=2      K=3      K=4      K=5      K=6      K=7      K=8      K=9     K=10
376.6955 373.2774 371.9943 371.4241 372.7660 374.8488 376.9537 379.6253 382.4002 384.5774
> grp$stat # Best k based on BIC
     K=4
371.4241
> head(grp$grp, 59) # Group membership (based on your defined number of clusters)
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33
 1  1  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  2  2  1  1  1  1  1  1  1  1  1  1  1
34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59
 1  1  1  1  1  1  1  1  1  2  1  1  1  1  1  1  1  1  4  4  4  4  4  4  4  4
Levels: 1 2 3 4
> grp$size # Group size per k cluster
[1] 29  3 19  8
>
```

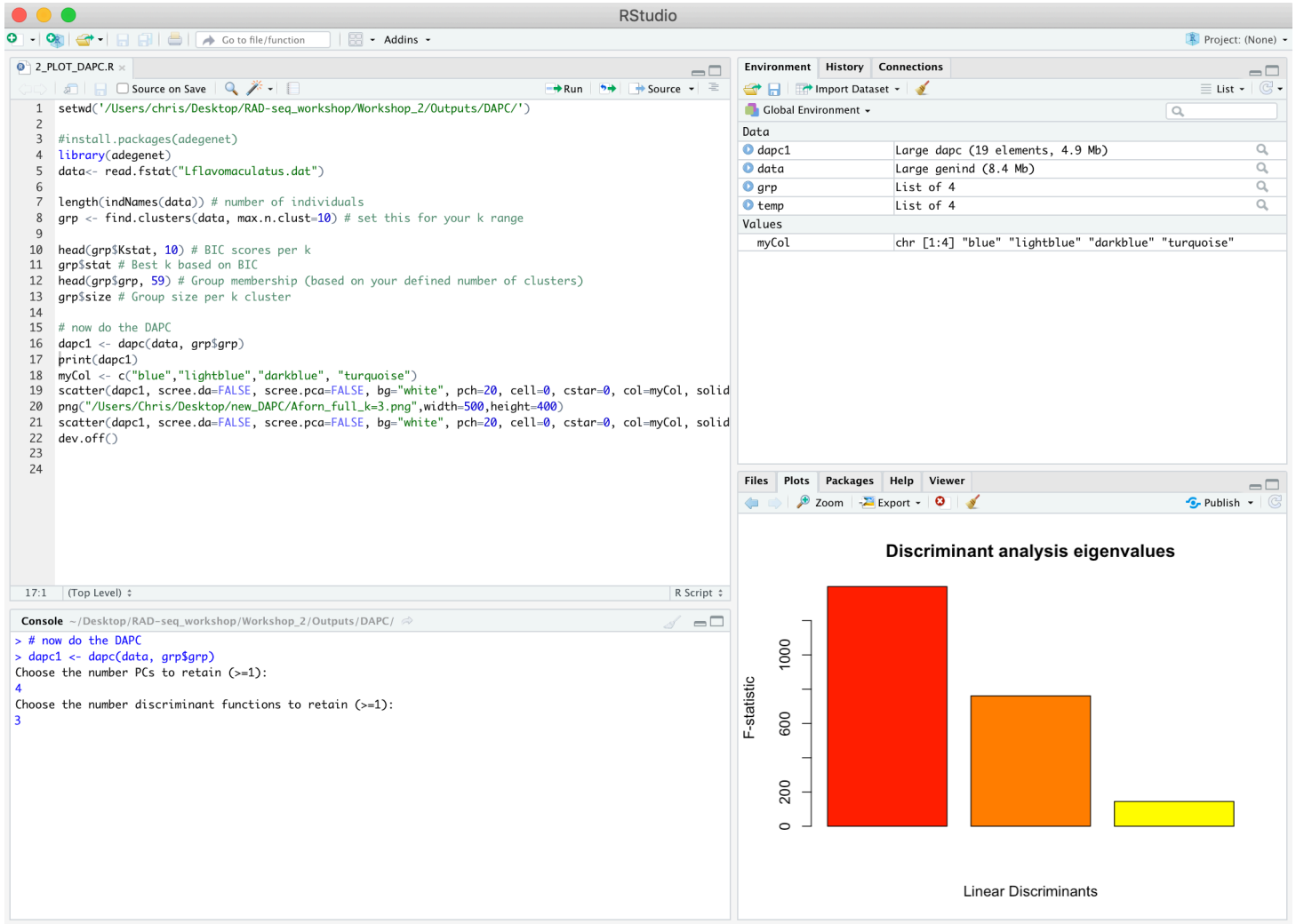
Population structure - DAPC

- Choose how many PCs to retain (top of the initial upward curve)



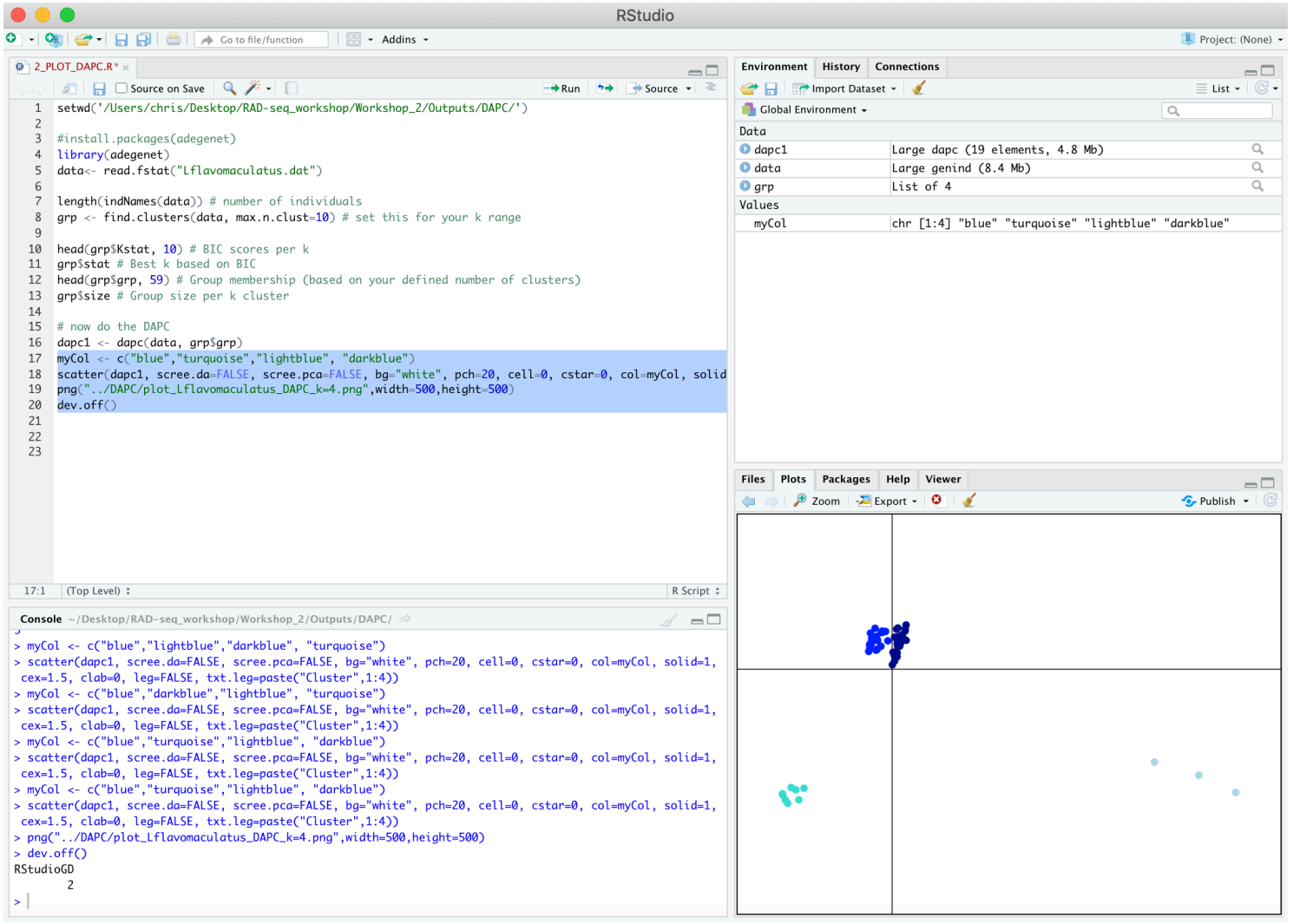
Population structure - DAPC

- Choose how many discriminant functions to retain (in this case ,3)

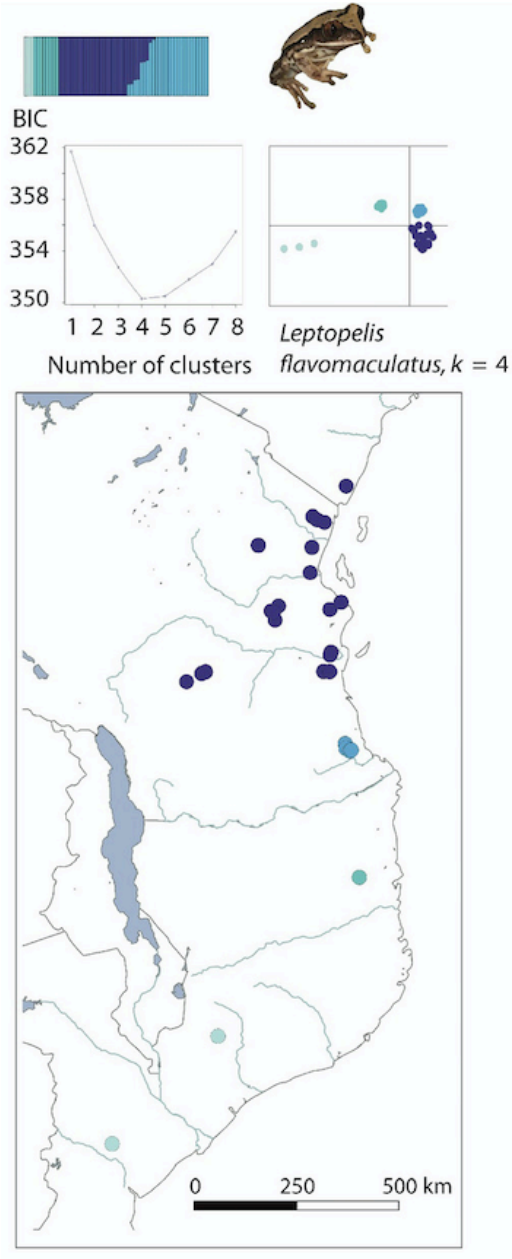


Population structure - DAPC

- Lines 17-20 will plot the resulting PCA with 4 clusters



Population structure - summary



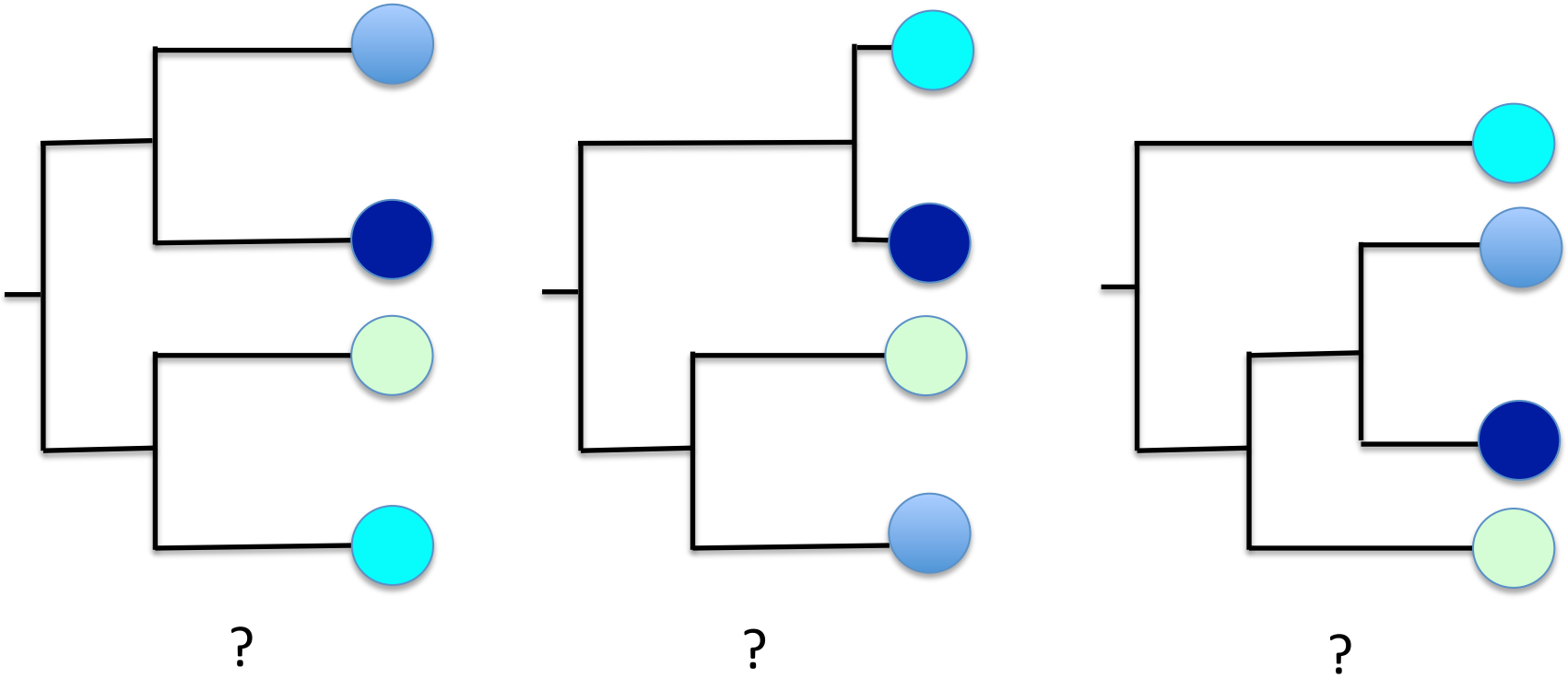
- Both of our Admixture and DAPC analyses recover the same results, k=4…

- 1 isolated cluster towards southern Mozambique
- 1 cluster in northern Mozambique
- 1 cluster in southern Tanzania
- 1 large cluster in northern Tanzania and Kenya

Phylogenetic relationships

- So now we have an idea of the structure of this data, but what about the evolutionary relationships between populations?

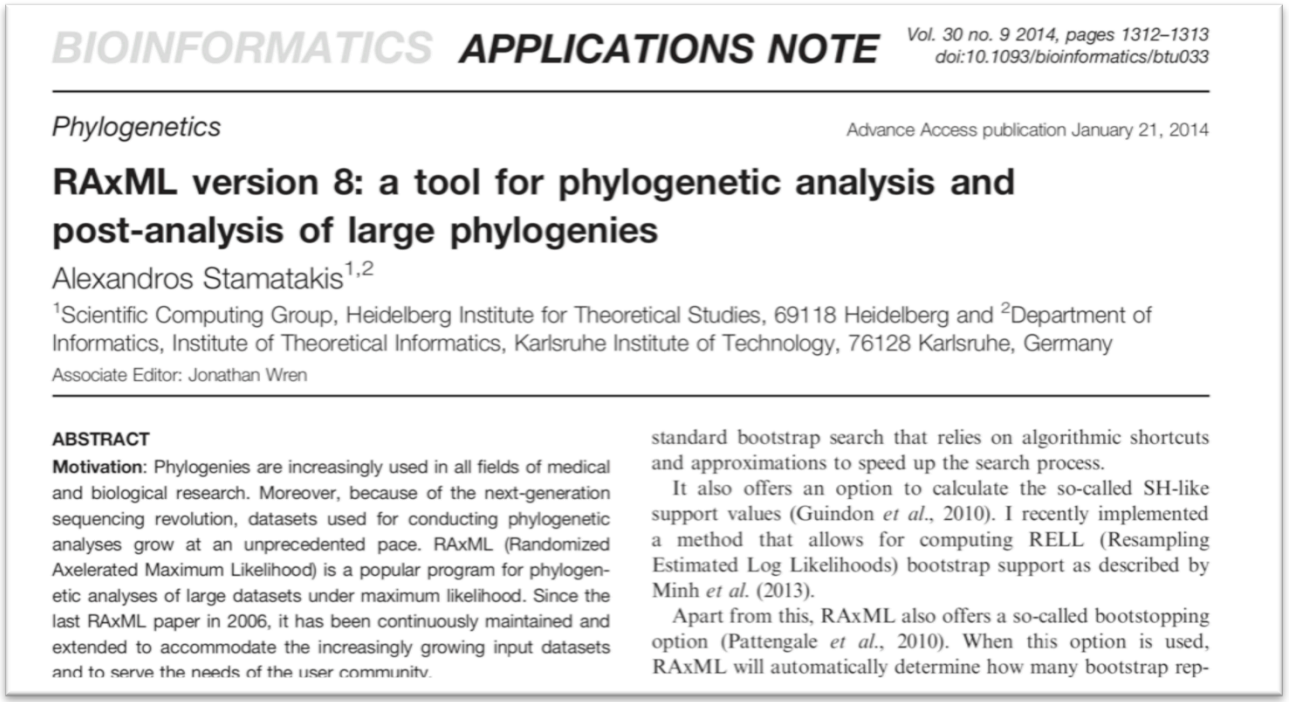
- For our later demographic modelling this is essential to know, we need to get the shape of our models correct



Phylogenetic relationships - RAxML

- A simple start to look how relationships between individuals and populations is with RAxML (Stamatakis, 2014)

- What is RAxML? A ML phylogenetic tool for visualising evolutionary relationships



BIOINFORMATICS APPLICATIONS NOTE
Vol. 30 no. 9 2014, pages 1312–1313
doi:10.1093/bioinformatics/btu033

Phylogenetics

Advance Access publication January 21, 2014

RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies

Alexandros Stamatakis[1,2]

[1]Scientific Computing Group, Heidelberg Institute for Theoretical Studies, 69118 Heidelberg and [2]Department of Informatics, Institute of Theoretical Informatics, Karlsruhe Institute of Technology, 76128 Karlsruhe, Germany

Associate Editor: Jonathan Wren

ABSTRACT

Motivation: Phylogenies are increasingly used in all fields of medical and biological research. Moreover, because of the next-generation sequencing revolution, datasets used for conducting phylogenetic analyses grow at an unprecedented pace. RAxML (Randomized Axelerated Maximum Likelihood) is a popular program for phylogenetic analyses of large datasets under maximum likelihood. Since the last RAxML paper in 2006, it has been continuously maintained and extended to accommodate the increasingly growing input datasets and to serve the needs of the user community.

standard bootstrap search that relies on algorithmic shortcuts and approximations to speed up the search process.

It also offers an option to calculate the so-called SH-like support values (Guindon *et al.*, 2010). I recently implemented a method that allows for computing RELL (Resampling Estimated Log Likelihoods) bootstrap support as described by Minh *et al.* (2013).

Apart from this, RAxML also offers a so-called bootstopping option (Pattengale *et al.*, 2010). When this option is used, RAxML will automatically determine how many bootstrap rep-

Phylogenetic relationships - RAxML

- Your input file here is a phylip file (.phy, n=53 due to missing data)
- You can submit this job to RAxML on the cluster with qsub **run_RAxML.sh**, or with the CIPRES gateway
- First the ML search, then bootstrapping to a final tree

run_RAxML.sh

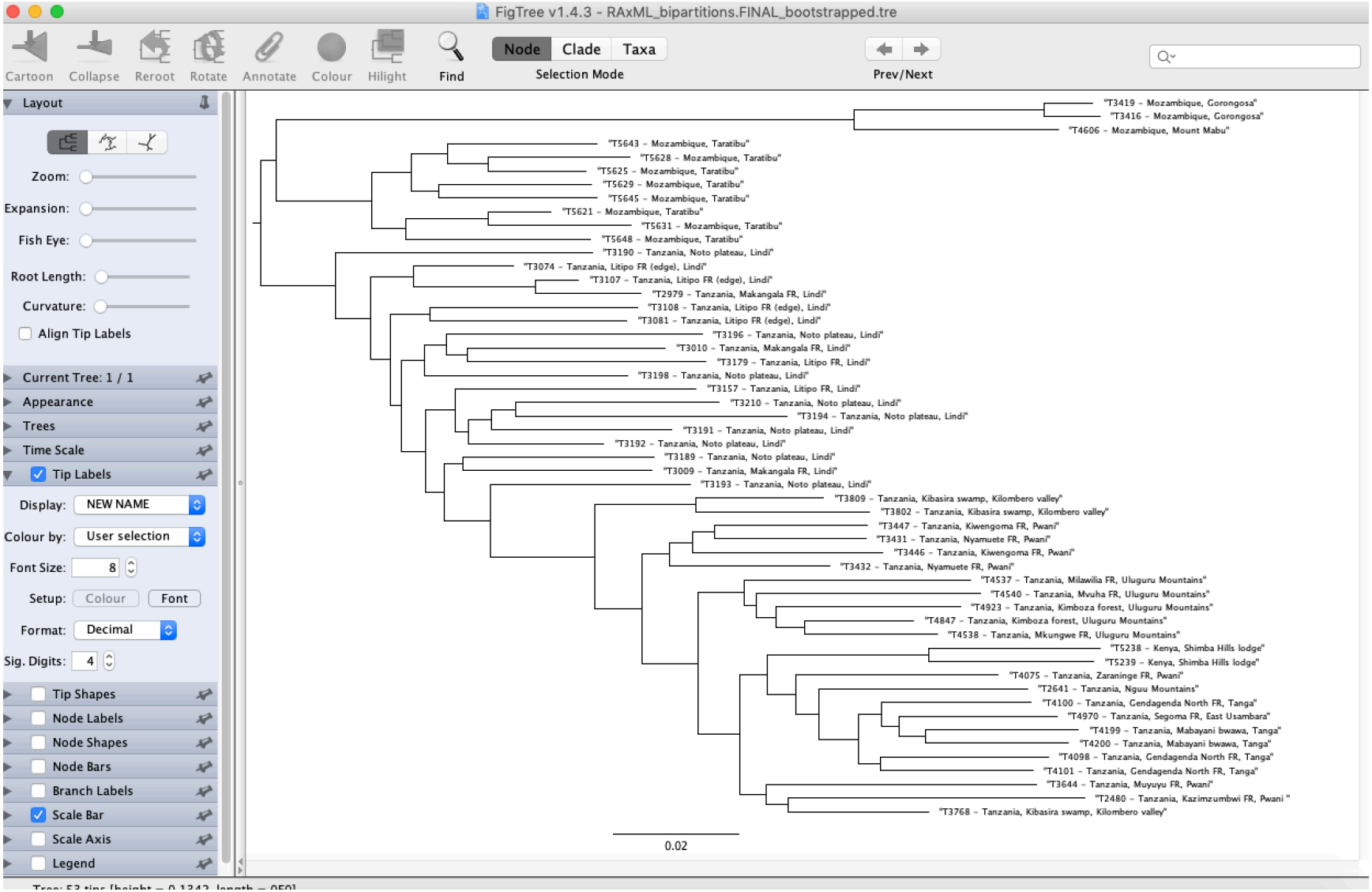
/private/var/folders/rp/_4tmp0_d3sv44byf1__k2w5w0000gn/T/0f8aa5cb-2a4a-4d58-a70f-da82442a5887/public/barratt/submit_scripts/run_RAxML.sh

```
1  ####### UFZ cluster job submit #########
2
3  #!/bin/bash
4  #$ -N RAxML
5  #$ -M christopher_david.barratt@uni-leipzig.de
6  #$ -m beas
7  Lflavomaculatus#$ -wd /public/barratt/work
8  #$ -S /bin/bash
9  #$ -o /public/barratt/work/Lflavomaculatus/job_logs/$JOB_NAME-$JOB_ID.log
10 #$ -j y
11 #$ -pe smp 1
12 #$ -l h_vmem=10G
13 #$ -l h_rt=72:00:00
14
15 #get slot information etc. (determine no. of slots with -pe above (can also be a range so that it takes the maximum available slots but will settle for less))
16 echo "Job name: $JOB_NAME"
17 echo "$NSLOTS slots have been dedicated to this job"
18 echo "On following node(s)"
19 cat $PE_HOSTFILE
20 echo "pe host file is $PE_HOSTFILE"
21
22 #load environment
23 module load miniconda/3
24 source activate PanAf_genomics_bioconda
25 module load python/2/7.13-2
26 source ~/barratt_env/bin/activate
27
28 module load raxml/8.1.17-1
29
30 cd /public/barratt/work/Lflavomaculatus/RAxML/
31
32 raxmlHPC -f a -m GTRCAT -p 12345 -x 12345 -# 100 -s /public/barratt/work/Lflavomaculatus/RAxML/Lflavomaculatus.phy --asc-corr=lewis -n bootstrapped
33 raxmlHPC -m GTRCAT -p 12345 -f b -t RAxML_bestTree.bootstrapped -z RAxML_bootstrap.bootstrapped -n FINAL_bootstrapped.tre
34
35
```

Phylogenetic relationships - RAxML

- Your output files will be several - bootstrap files and also the final tree
- Open the RAxML_bipartitons.FINAL_bootstrapped.tre in Figtree to see the results of the relationships (say ok to the label question)
- Open ‘Trees’ on the left panel, click ‘Root Tree’ and select ‘Midpoint’ (We do this because we have no outgroup in this dataset)
- If you want more informative labels click File>Import Annotations and import RAD_IDs.txt from Input_files/RAxML folder
- Then in “Tip labels” choose Display as “NEW NAME”

Phylogenetic relationships - RAxML



Phylogenetic relationships - RAxML

- Important caveat 1 − RAxML is phylogenetic software, so if your data are populations within species, or closely related species then other coalescent-based methods are much more suitable (which we will now do)

- Important caveat 2 − RAxML seems to perform better with full RAD-seq loci, not just single SNPs. Computationally this is more intensive but bootstrap scores are much better when you supply the full sequence of each RAD locus

Phylogenetic relationships - SNAPP

- What is SNAPP? A (relatively) rapid algorithm for looking at evolutionary relationships between populations using multi-locus SNP data
- Based on the backwards in time multispecies coalescent model, it integrates separate gene trees together for all of your SNPs
- Only feasible for a relatively small number of species due to the mathematical intensity of the gene tree integration (but perfect for shallow phylogenies or population structure)

Phylogenetic relationships - SNAPP

- Built as an add-on package to the popular BEAST software (Bryant et al. 2012)

Inferring Species Trees Directly from Biallelic Genetic Markers: Bypassing Gene Trees in a Full Coalescent Analysis

David Bryant,*[,1] Remco Bouckaert,[2] Joseph Felsenstein,[3] Noah A. Rosenberg,[4] and Arindam RoyChoudhury[5]

[1]Department of Mathematics and Statistics and the Allan Wilson Centre for Molecular Ecology and Evolution, University of Otago, Dunedin, New Zealand
[2]Computational Evolution Group, Department of Computer Science, University of Auckland, Auckland, New Zealand
[3]Department of Genome Sciences and Department of Biology, University of Washington
[4]Department of Biology, Stanford University
[5]Department of Biostatistics, Mailman School of Public Health, Columbia University

***Corresponding author:** E-mail: david.bryant@otago.ac.nz.

Associate editor: Rasmus Nielsen

Abstract

The multispecies coalescent provides an elegant theoretical framework for estimating species trees and species demographics from genetic markers. However, practical applications of the multispecies coalescent model are limited by the need to integrate or sample over all gene trees possible for each genetic marker. Here we describe a polynomial-time algorithm that computes the likelihood of a species tree directly from the markers under a finite-sites model of mutation effectively integrating over all possible gene trees. The method applies to independent (unlinked) biallelic markers such as well-spaced single nucleotide polymorphisms, and we have implemented it in SNAPP, a Markov chain Monte Carlo sampler for inferring species trees, divergence dates, and population sizes. We report results from simulation experiments and from an analysis of 1997 amplified fragment length polymorphism loci in 69 individuals sampled from six species of *Ourisia* (New Zealand native foxglove).

Key words: multispecies coalescent, species trees, SNP, AFLP, effective population size, SNAPP.

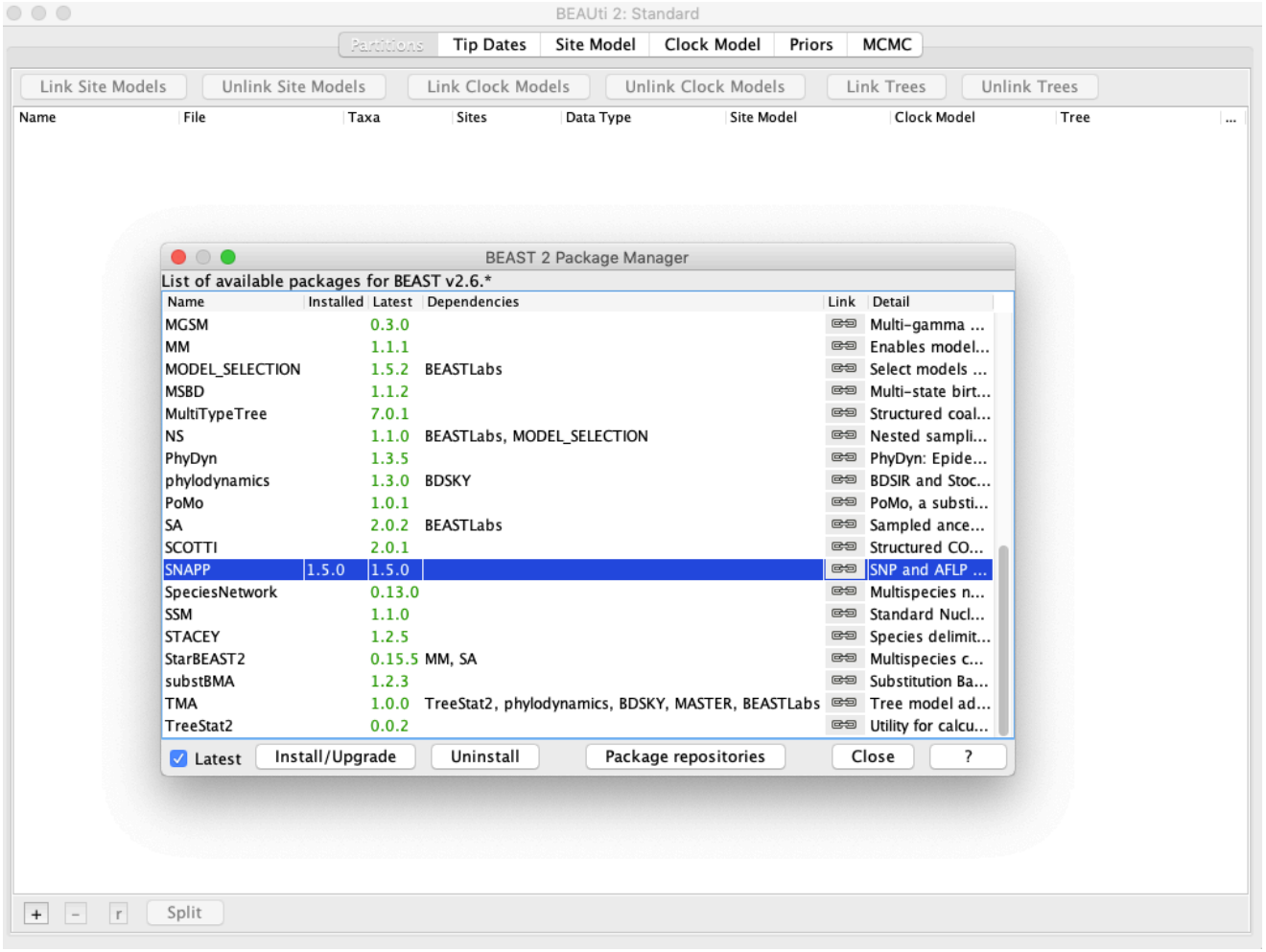
Research article

Phylogenetic relationships - SNAPP

- I thinned our nex file (Lflavomaculatus_thinned.nex) - SNAPP is very computationally intensive
- From each of the 4 populations identified earlier there are 3-4 representative individuals, each with 8598 SNPs (missing data=N)
- We will use BEAUti (part of the BEAST package) to build the xml file for this which SNAPP needs to run
- Keep in mind we only have a few thousand SNPs (~8500) because I filtered quite strictly during our processing in Stacks. If you have hundreds of thousands/millions of SNPs from other types of data then you will need to assemble your xml file bioinformatically (BEAuti will just freeze)

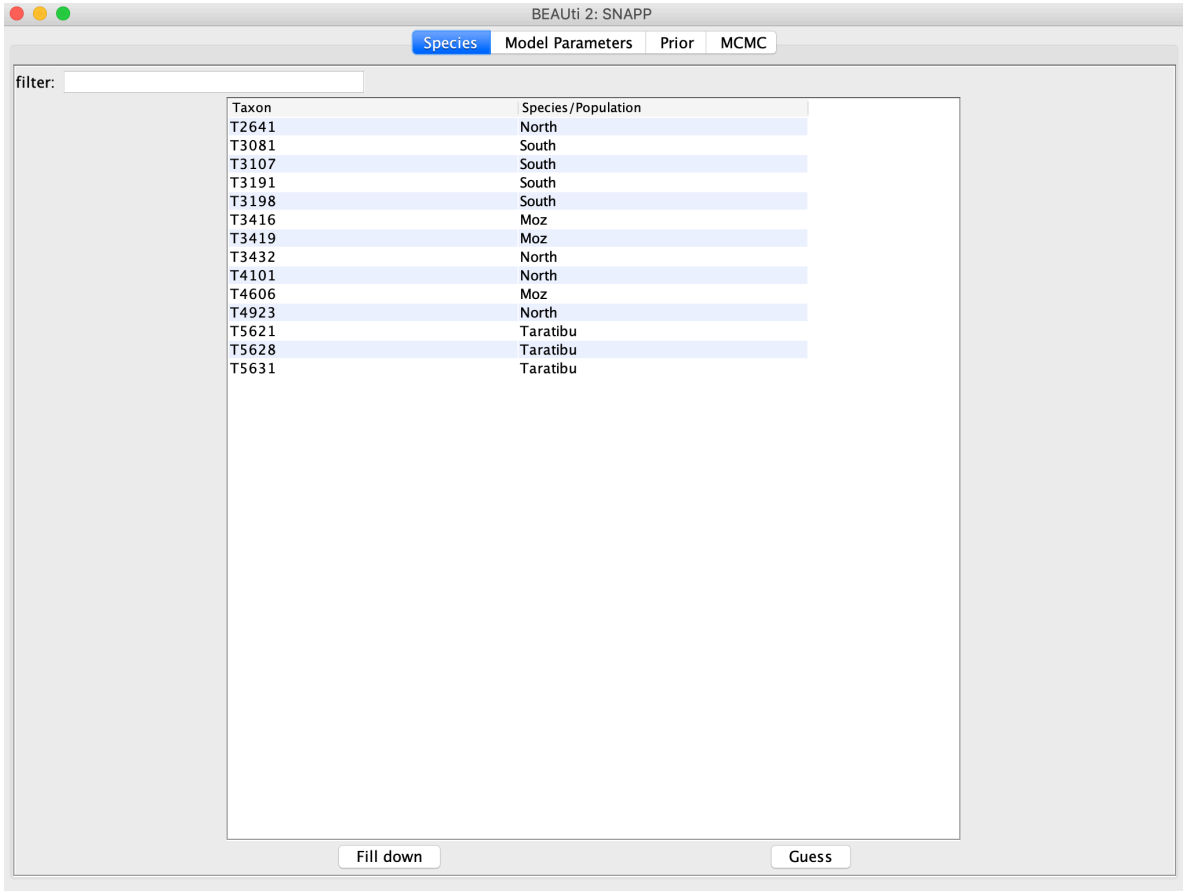
Phylogenetic relationships - SNAPP

- First, open BEAUti and go File>Manage Packages, scroll to SNAPP and install



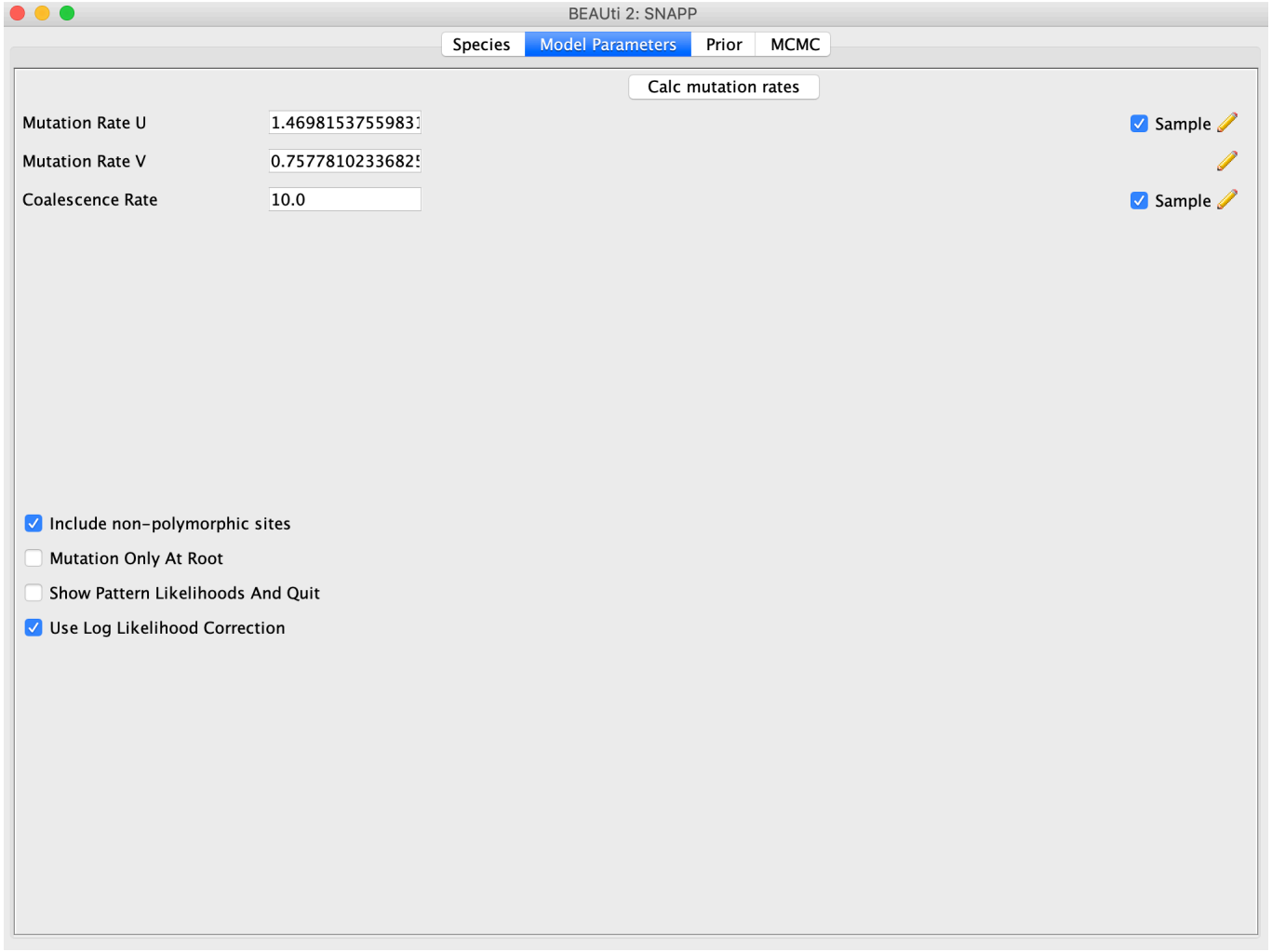
Phylogenetic relationships - SNAPP

- Go to File>Template>SNAPP
- Now go File>Import Alignment and select the thinned nexus file
- We need to tell SNAPP which individual belongs to which population, so fill in as below (don't leave any invisible spaces)



Phylogenetic relationships - SNAPP

- Click on the Model Parameters tab and calculate mutation rates button (calculates from the data)



Phylogenetic relationships - SNAPP

- Leave the Prior tab as is for now, setting priors can be complex, but you could and should explore the effects of different priors on your output trees. The defaults are probably not suitable for your data

- Click on MCMC and keep chain length at 1000000, change tracelog filename to Lflavomaculatus.log and treelog filename to Lflavomaculatus.trees. Each of these is logging every 1000 so your output files will have 1000 entries

- Save as Lflavomaculatus_thinned.xml

- You can check if it runs by opening BEAST and running this xml file. Cancel it after it begins to run. You can submit to CIPRES if you want to run yourself (setting up SNAPP on the UFZ cluster is time-consuming, we won't go through that here, but the **run_SNAPP.sh** is there for you to take a look at)

Phylogenetic relationships - SNAPP

run_SNAPP.sh

/private/var/folders/rp/_4tmp0_d3sv44byf1__k2w5w0000gn/T/0f8aa5cb-2a4a-4d58-a70f-da82442a5887/public/barratt/submit_scripts/run_SNAPP.sh

```
1  ####### UFZ cluster job submit #########
2
3  #!/bin/bash
4  #$ -N SNAPP
5  #$ -M christopher_david.barratt@uni-leipzig.de
6  #$ -m beas
7  #$ -wd /work/$USER
8  #$ -S /bin/bash
9  #$ -o /public/barratt/work/Lflavomaculatus/job_logs/$JOB_NAME-$JOB_ID.log
10 #$ -j y
11 #$ -pe smp 4
12 #$ -l h_vmem=10G
13 #$ -l h_rt=120:00:00
14
15 #get slot information etc. (determine no. of slots with -pe above (can also be a range so that it takes the maximum available slots but will settle for less))
16 echo "Job name: $JOB_NAME"
17 echo "$NSLOTS slots have been dedicated to this job"
18 echo "On following node(s)"
19 cat $PE_HOSTFILE
20 echo "pe host file is $PE_HOSTFILE"
21
22 #load modules, start job
23 source ~/barratt_env/bin/activate
24 module load beast/2.4.6-1
25 module load java
26
27 cd /public/barratt/work/Lflavomaculatus/Outputs/SNAPP/
28
29 export JVM_ARGS="-Xms100M -Xmx25G"
30 beast -overwrite -beagle_GPU -threads ${NSLOTS:-1} /public/barratt/work/Lflavomaculatus/Inputs/SNAPP/Lflavomaculatus_thinned.xml
31
```

Phylogenetic relationships - SNAPP

- This will take 12-13 hours to run because of the calculations. Ideally you would have 10,000 tree and log entries for later on, but in this case I have just given 1000. I have the output files prepared for you… . log and .trees

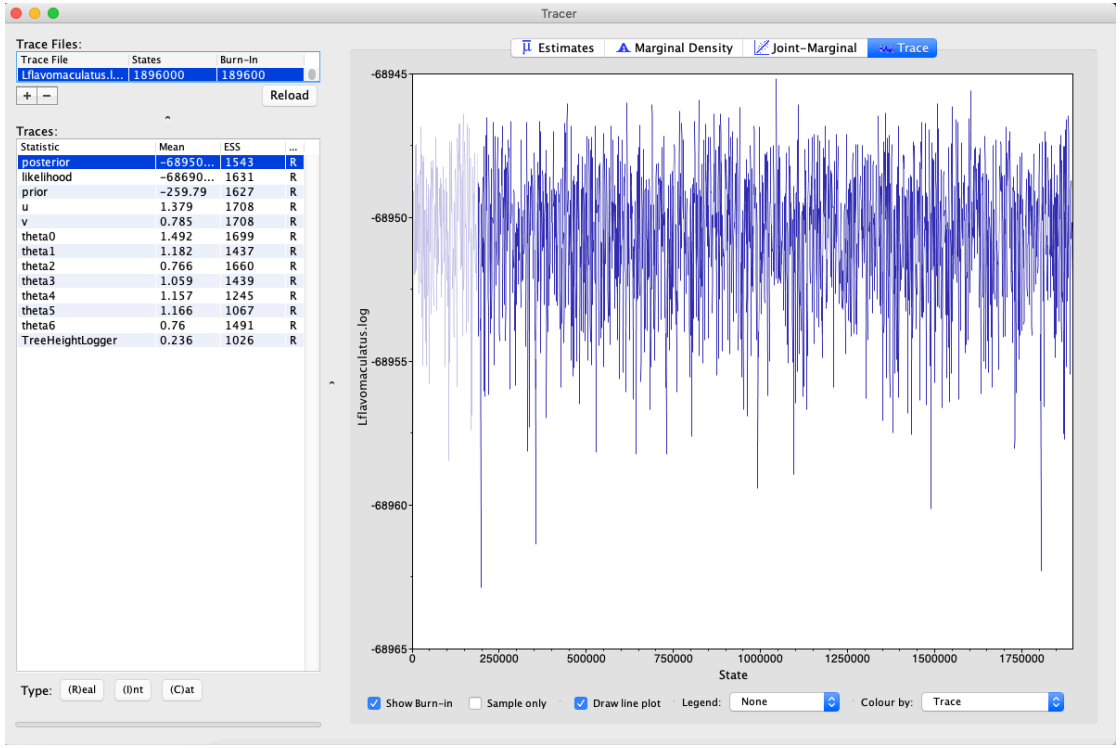
- Using Tracer you can check the log files to see how well they have converged. ESS scores (Effective Sample Sizes) above 200 are typically considered as sufficient (but are less here due to above)

- Sometimes a BEAST run can get stuck in weird parameter space so you may need to rerun analyses to check that your results hold

- Another consideration is to run the same analysis with different representative individuals (also to see if results are consistent)

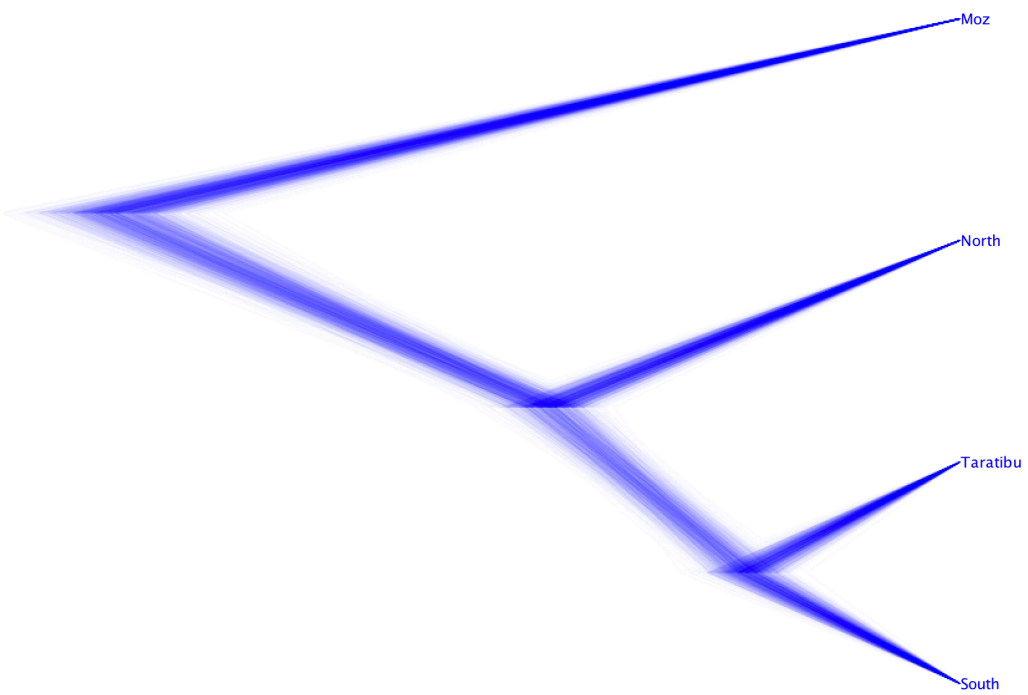
- Another more in depth tutorial can be found here

Phylogenetic relationships - SNAPP



- Load the .log files in Tracer (part of BEAST package)
- These are your 1000 logs. Typically you want your posterior ESS score to be over 200
- Look at the trace tab and see if it is in the right parameter space. A fat caterpillar is what you want to see ☺

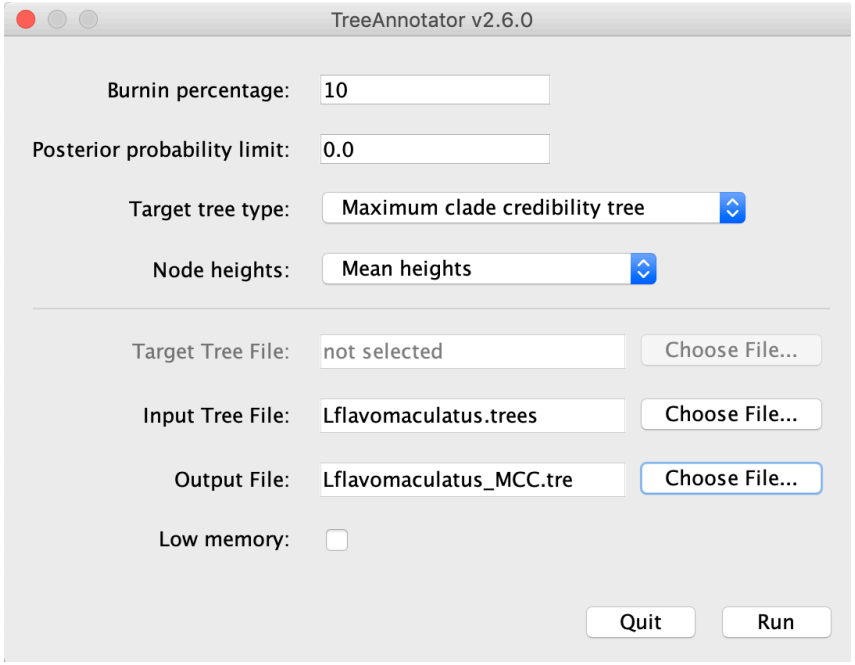
Phylogenetic relationships - SNAPP



- Load the .trees file in DensiTree (part of BEAST package)
- This is your 1000 trees and they all look consistent, which is very good!
- You can see here how the populations are related to one another, based on 8598 SNPs

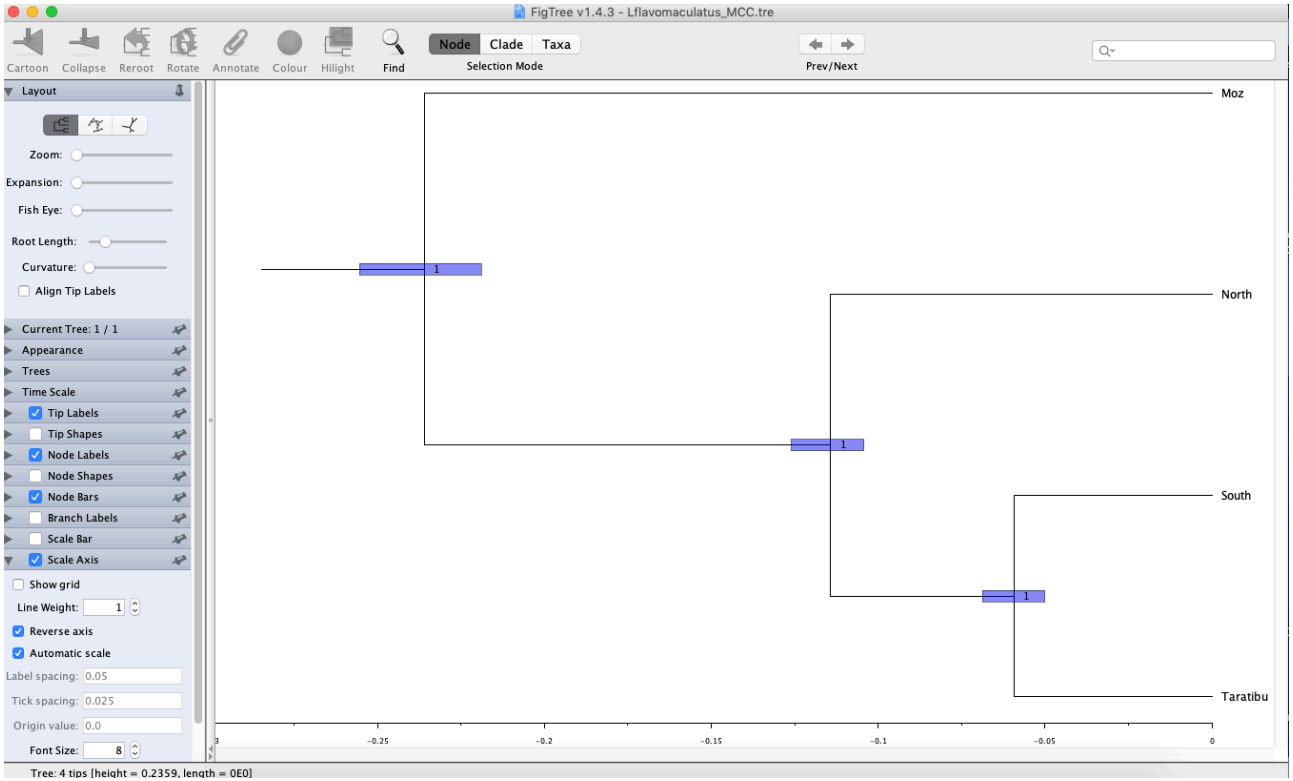
Phylogenetic relationships - SNAPP

- From all your trees you can make a single MCC (Maximum Clade Credibility) summary tree…
- Open TreeAnnotator (again part of BEAST package)
- Set your burnin % (judge this by looking at Tracer plot on logfiles), specify your input trees file and call your output a .tre file



Phylogenetic relationships - SNAPP

- Open your output tree file in FigTree or some other tree drawing visualisation software…

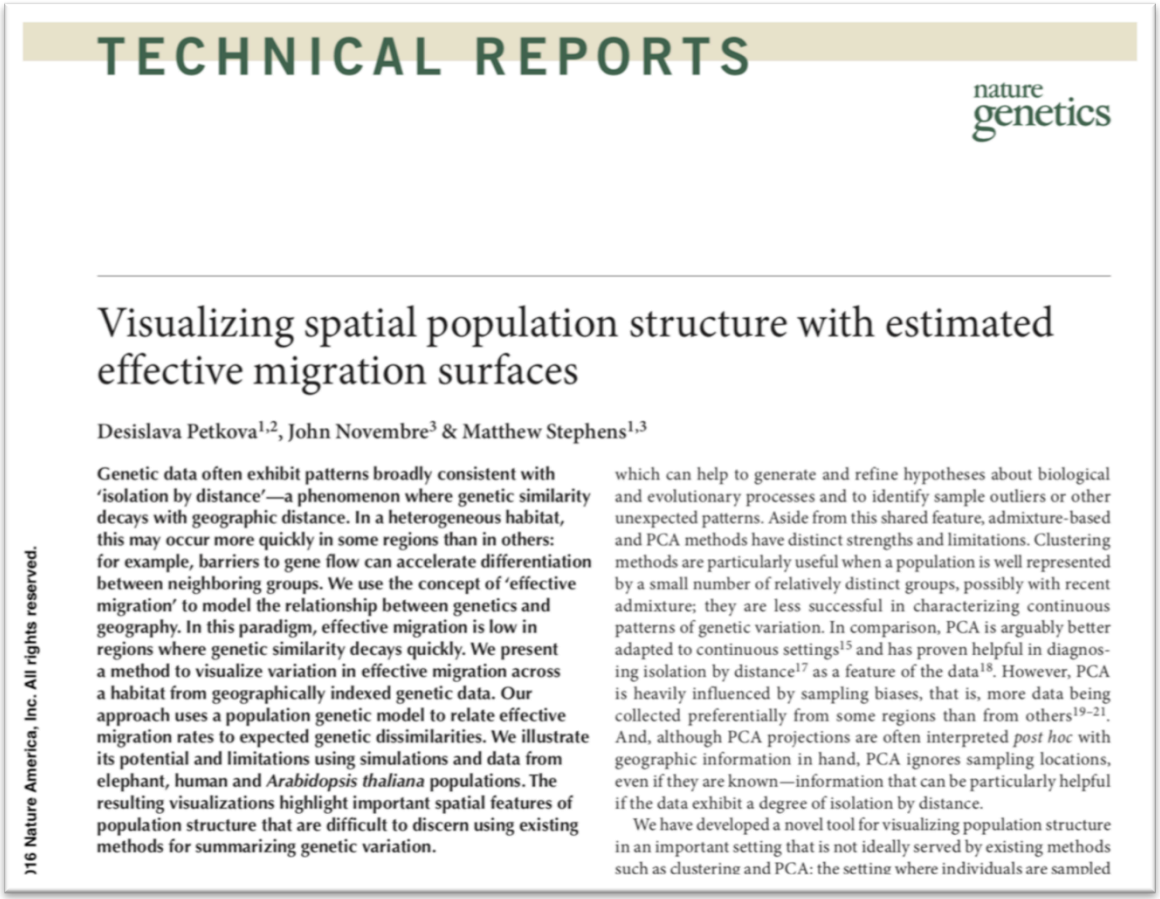


- If you have an accurate and relevant mutation rate you can convert the dates of population splits (1 x 10-8, Lynch 2010, PNAS)
- So here -0.1 would represent 1 million years (0.1 x 10-8)

That was a lot of info. Let's have a cup of coffee…

Landscape connectivity and barriers - EEMS

- What is EEMS? Estimated Effective Migration Surfaces − a way to visualise connectivity and barriers across the sampling region using FST derived metrics (Petkova et al. 2016)



TECHNICAL REPORTS

nature genetics

Visualizing spatial population structure with estimated effective migration surfaces

Desislava Petkova[1,2], John Novembre[3] & Matthew Stephens[1,3]

Genetic data often exhibit patterns broadly consistent with 'isolation by distance'—a phenomenon where genetic similarity decays with geographic distance. In a heterogeneous habitat, this may occur more quickly in some regions than in others: for example, barriers to gene flow can accelerate differentiation between neighboring groups. We use the concept of 'effective migration' to model the relationship between genetics and geography. In this paradigm, effective migration is low in regions where genetic similarity decays quickly. We present a method to visualize variation in effective migration across a habitat from geographically indexed genetic data. Our approach uses a population genetic model to relate effective migration rates to expected genetic dissimilarities. We illustrate its potential and limitations using simulations and data from elephant, human and *Arabidopsis thaliana* populations. The resulting visualizations highlight important spatial features of population structure that are difficult to discern using existing methods for summarizing genetic variation.

which can help to generate and refine hypotheses about biological and evolutionary processes and to identify sample outliers or other unexpected patterns. Aside from this shared feature, admixture-based and PCA methods have distinct strengths and limitations. Clustering methods are particularly useful when a population is well represented by a small number of relatively distinct groups, possibly with recent admixture; they are less successful in characterizing continuous patterns of genetic variation. In comparison, PCA is arguably better adapted to continuous settings[15] and has proven helpful in diagnosing isolation by distance[17] as a feature of the data[18]. However, PCA is heavily influenced by sampling biases, that is, more data being collected preferentially from some regions than from others[19–21]. And, although PCA projections are often interpreted *post hoc* with geographic information in hand, PCA ignores sampling locations, even if they are known—information that can be particularly helpful if the data exhibit a degree of isolation by distance.

We have developed a novel tool for visualizing population structure in an important setting that is not ideally served by existing methods such as clustering and PCA: the setting where individuals are sampled

Landscape connectivity and barriers - EEMS

- Assumption that genetic similarity between samples decays with geographic distance
- This is complex in space and time but the basic premise is that a barrier would increase differentiation (=reduced gene flow)
- They apply a population genetic model to relate genetic dissimilarity (FST) to effective migration rates
- Essentially what the method does is computes haplotype sharing between sites and then interpolates these across geographic space so you can visualise potential barriers and areas of high gene flow

Landscape connectivity and barriers - EEMS

- EEMS needs similar files to Admixture but with some additional extras. Firstly it needs the .count, .diffs and .order file which the first lines of the **3_RUN_EEMS.sh** script generates from the ped and bed files

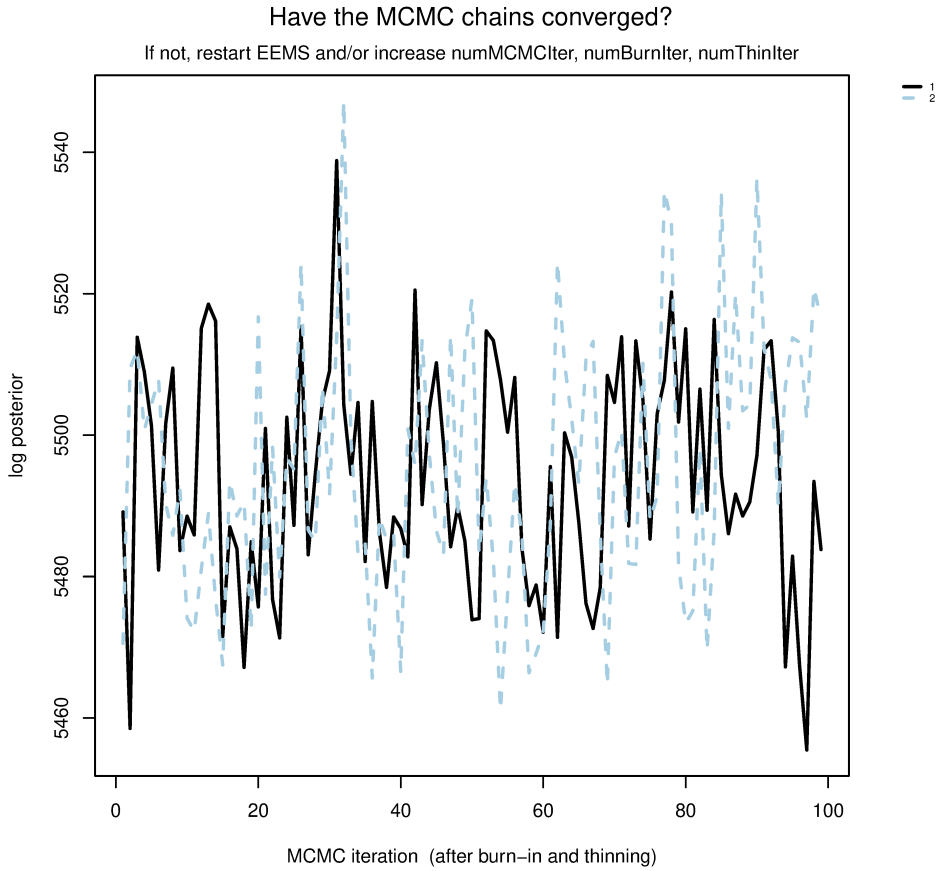
- Secondly, EEMS also requires a .coord file (location info for each individual) and an .outer file to spatially bind the analysis (generated here:http://www.birdtheme.org/useful/v3tool.html)

- The second part of the script is actually running EEMS, so you can open terminal, cd to the proper directory and and type bash **3_RUN_EEMS.sh** followed by enter

- This will take ~4 hours to run (2 x 1000000 MCMC runs, you can reduce if you want)

Landscape connectivity and barriers - EEMS

- We can plot the results of this with the **3_PLOT_EEMS.R** script – many output files will be generated, most of which are used to build the important images, so let's focus on the main 3

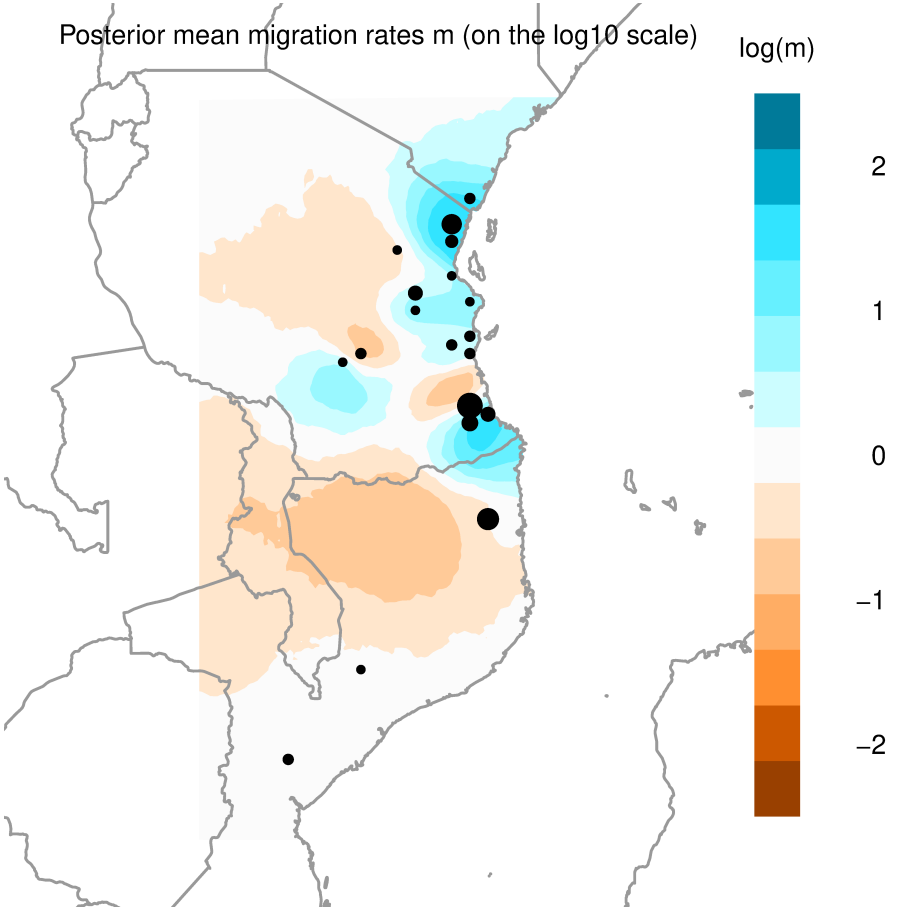


pilogl01.png

- Have the 2 MCMC runs converged?
- This looks fairly ok, not perfect but not too bad…

Landscape connectivity and barriers - EEMS

- We can plot the results of this with the **3_PLOT_EEMS.R** script – many output files will be generated, but lets focus on the main 3

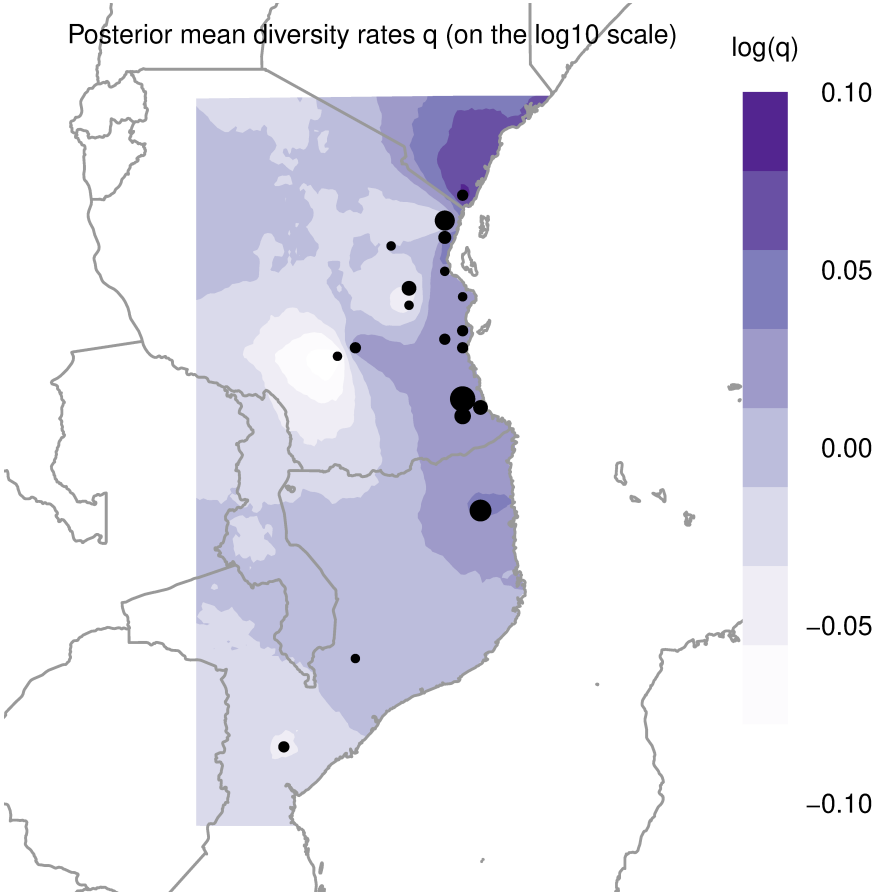


mrates01.png

- Blue = higher migration
- Orange = lower migration

Landscape connectivity and barriers - EEMS

- We can plot the results of this with the **3_PLOT_EEMS.R** script – many output files will be generated, but lets focus on the main 3



C2 - qrates01.png

- Purple = higher diversity
- White = diversity

Landscape connectivity and barriers - EEMS

- Should run EEMS for longer MCMC chains (we have done 1 million iterations here, should be at least 5 million)

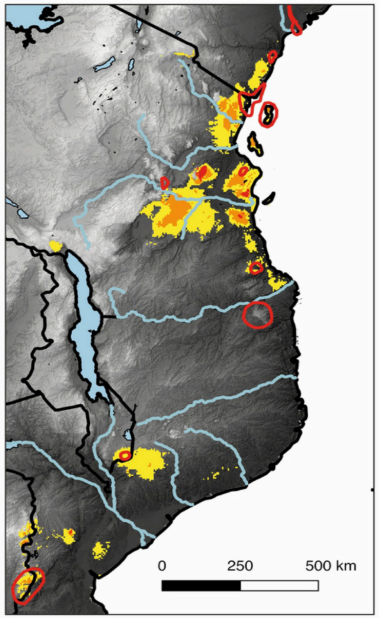
- We also ran two independent chains (2 .ini files) and take the results from both − providing they are sampling similar parameter space (have the chains converged?)

- Deme size should also be examined, we chose 700 here because I know this data. I had experimented with 100, 250, 500, 1000, 5000 to see how localities became merged in the results

Summary – all results so far

- Now we have a very good handle on population structure, admixture between pops, evolutionary relationships between pops and potential barriers to gene flow across the landscape (where FST is higher than expected due to geography)

- We may also have an idea of the historical ranges of these populations using SDMs projected into the past (not essential)



- So we can use all of this information to build sensible demographic models to be tested in ∂a∂i…

Demographic model selection - ∂a∂i

- ∂a∂i is an extremely powerful and flexible framework for demographic inference (Gutenkunst et al. 2009)

OPEN ACCESS Freely available online PLoS GENETICS

Inferring the Joint Demographic History of Multiple Populations from Multidimensional SNP Frequency Data

Ryan N. Gutenkunst[1]*, Ryan D. Hernandez[2], Scott H. Williamson[3], Carlos D. Bustamante[3]

1 Theoretical Biology and Biophysics and Center for Nonlinear Studies, Los Alamos National Laboratory, Los Alamos, New Mexico, United States of America, **2** Human Genetics, University of Chicago, Chicago, Illinois, United States of America, **3** Biological Statistics and Computational Biology, Cornell University, Ithaca, New York, United States of America

Abstract

Demographic models built from genetic data play important roles in illuminating prehistorical events and serving as null models in genome scans for selection. We introduce an inference method based on the joint frequency spectrum of genetic variants within and between populations. For candidate models we numerically compute the expected spectrum using a diffusion approximation to the one-locus, two-allele Wright-Fisher process, involving up to three simultaneous populations. Our approach is a composite likelihood scheme, since linkage between neutral loci alters the variance but not the expectation of the frequency spectrum. We thus use bootstraps incorporating linkage to estimate uncertainties for parameters and significance values for hypothesis tests. Our method can also incorporate selection on single sites, predicting the joint distribution of selected alleles among populations experiencing a bevy of evolutionary forces, including expansions, contractions, migrations, and admixture. We model human expansion out of Africa and the settlement of the New World, using 5 Mb of noncoding DNA resequenced in 68 individuals from 4 populations (YRI, CHB, CEU, and MXL) by the Environmental Genome Project. We infer divergence between West African and Eurasian populations 140 thousand years ago (95% confidence interval: 40–270 kya). This is earlier than other genetic studies, in part because we incorporate migration. We estimate the European (CEU) and East Asian (CHB) divergence time to be 23 kya (95% c.i.: 17–43 kya), long after archeological evidence places modern humans in Europe. Finally, we estimate divergence between East Asians (CHB) and Mexican-Americans (MXL) of 22 kya (95% c.i.: 16.3–26.9 kya), and our analysis yields no evidence for subsequent migration. Furthermore, combining our demographic model with a previously estimated distribution of selective effects among newly arising amino acid mutations accurately predicts the frequency spectrum of nonsynonymous variants across three continental populations (YRI, CHB, CEU).

Citation: Gutenkunst RN, Hernandez RD, Williamson SH, Bustamante CD (2009) Inferring the Joint Demographic History of Multiple Populations from Multidimensional SNP Frequency Data. PLoS Genet 5(10): e1000695. doi:10.1371/journal.pgen.1000695

Editor: Gil McVean, University of Oxford, United Kingdom

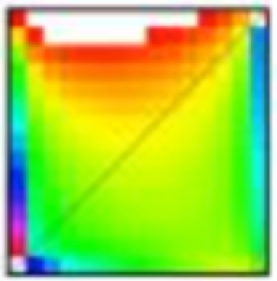
Received February 12, 2009; **Accepted** September 23, 2009; **Published** October 23, 2009

Demographic model selection - ∂a∂i

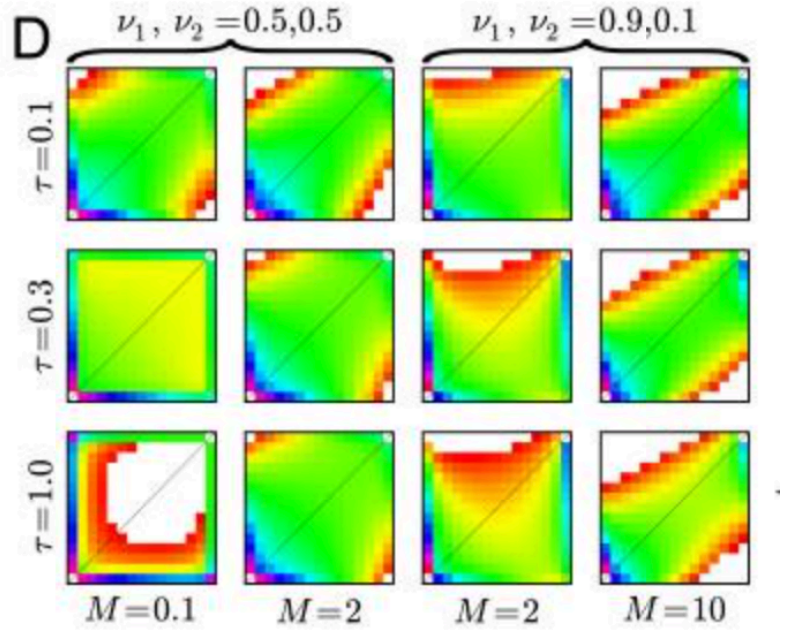
- Uses the Joint Site Frequency Spectrum (JSFS) between populations (1D, 2D or 3D)
- JSFS are multi-pop SFS; a summary of SNP allele frequency distributions between populations (Wright-Fisher models)
- Shape of the JSFS is sensitive to demography, such as population size changes, migration, and substructure
- Comparing observed empirical data to the expected JSFS under a given demographic model, we can assess the goodness of fit of that the model to the data, and use likelihood theory to estimate the best fit parameters
- Keep in mind that no model is ever perfect!

Demographic model selection - ðaði

- We build and test models to be compared against empirical data
- Each model needs to be optimized to best fit the data, then likelihoods and AIC calculated
- Best likelihood and smallest delta AIC = best explanatory model



?



Demographic model selection - ðaði

What's happening under the hood in ðaði?

1. Build SNP matrix from the SNP data
2. Calculate observed JSFS (3, 1, 0, 1) [derived from 111, 2, x, 4]
3. Calculate expected JSFS under different hypothetical models (time steps, migration, pop size changes)
4. Assess goodness of fit between observed and expected data in a composite likelihood framework

	SNP 1	SNP 2	SNP 3	SNP 4	SNP 5
Sample 1			1	1	
Sample 2		1		1	
Sample 3	1			1	1
Sample 4		1		1	
Total	1	2	1	4	1

Demographic model selection - ∂a∂i

- Running and interpreting ∂a∂i can be an total nightmare. Dan Portik (University of Arizona) wrote this stunning pipeline which can be used to automate model selection: https://github.com/dportik/dadi_pipeline

- You still need to do **a lot** of thinking and piecing together previous evidence (e.g. pop structure, evolutionary relationships etc., migration) to build suitable models to fit your study system (i.e. hypotheses testing)

- Another important consideration is that you will need unlinked SNPs to perform AIC-based demographic model selection (this is fine for us as I used the write_single_SNP option in Stacks for these data!) − but keep this in mind before you analyse your own data

Demographic model selection - ∂a∂i

- The version of the pipeline we are using is the version before the current version which was uploaded about 2 weeks ago (you need to build Tkinter with your python version to run the new version, which I have not done yet)

- The main difference in the new one is that run optimizations are combined into a single .py script (dadi_Run_3D_Set.py) where previously they were separate for each model

- In public/barratt/work/Lflavomaculatus/Inputs/dadi there are:

a) dadi_3pops_north_south_taratibu_snps.txt − the SNP data input
b) 15 models to test (dadi_Run_Optimizations_MODEL_NAME.py)
c) Optimize_Functions.py (and pyc) where the functions are stored

Demographic model selection - ∂a∂i

- FYI dadi_3pops_north_south_taratibu_snps.txt was created using Dan's other Stacks haplotype converter: https://github.com/dportik/Stacks_pipeline/tree/master/4_haplotype_to_dadi_SNPsFile

- This takes the output haplotypes.tsv file from Stacks and some population information (e.g. those population clusters we found in our DAPC and pop structure analyses)

Demographic model selection - ∂a∂i

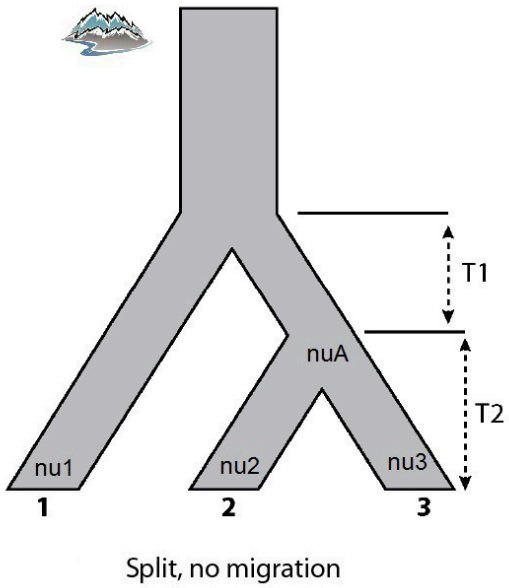
- OK, let's take a look at one of the model python scripts
- There are lots of comments and then the main information telling dadi how to build the SFS from the file (giving population names and allele projection sizes)

```
83  #==============================================================================
84  # Import data to create joint-site frequency dadi.Spectrum
85  #==============================================================================
86
87  #***************
88  snps = "/public/barratt/work/Lflavomaculatus/Inputs/dadi/dadi_3pops_north_south_taratibu_snps.txt"
89  #Create python dictionary from snps file
90  dd = dadi.Misc.make_data_dict(snps)
91
92  #***************
93  #pop_ids is a list which should match the populations headers of your SNPs file columns
94  pop_ids=['north','south','taratibu']
95
96  #***************
97  #projection sizes, in ALLELES not individuals
98  proj = [36,29,15]
99
100 #Convert this dictionary into folded AFS object
101 #[polarized = False] creates folded dadi.Spectrum object
102 fs = dadi.Spectrum.from_data_dict(dd, pop_ids=pop_ids, projections = proj, polarized = False)
103
104 #print some useful information about the afs or jsfs
105 print "\n\n==============================================================================\nData for site frequency dadi.Spectru
106 print "projection", proj
107 print "sample sizes", fs.sample_sizes
108 sfs_sum = numpy.around(fs.S(), 2)
109 print "Sum of SFS = ", sfs_sum, '\n', '\n'
110
```

Demographic model selection - ∂a∂i

- Scroll down a bit further and you will see the models themselves (all models are defined in every script as it does no harm to define them all). Here is the code for the first one, along with a visual representation

```
def split_nomig(params, ns, pts):
    """
    Model with split between pop 1 and (2,3), then split between 2 and 3.
    Migration does not occur between any population pair.
    """
    #6 parameters
    nu1, nuA, nu2, nu3, T1, T2 = params
    xx = Numerics.default_grid(pts)
    phi = PhiManip.phi_1D(xx)
    phi = PhiManip.phi_1D_to_2D(xx, phi)
    phi = Integration.two_pops(phi, xx, T1, nu1=nu1, nu2=nuA, m12=0, m21=0)
    phi = PhiManip.phi_2D_to_3D_split_2(xx, phi)
    phi = Integration.three_pops(phi, xx, T2, nu1=nu1, nu2=nu2, nu3=nu3, m12=0, m21=0, m23=0, m32=0, m13=0, m31=0)
    fs = Spectrum.from_phi(phi, ns, (xx,xx,xx))
    return fs

```



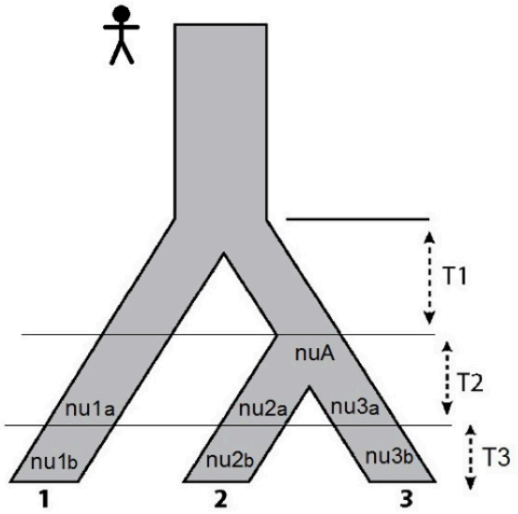
Split, no migration

- Note the important details – 6 params
- The location of the params in relation to where the divergences/pop size changes/ time steps are is important!

Demographic model selection - ∂a∂i

- Another model is very similar at first glance but the inclusion of an extra time step and parameters indicating recent size changes means that the biological interpretation of this model should be quite different

```
241 def split_nomig_human(params, ns, pts):
242     """
243     Model with split between pop 1 and (2,3), then split between 2 and 3.
244     Migration does not occur between any population pair, size change
245     """
246     #10 parameters
247     nu1a, nuA, nu2a, nu3a, nu1b, nu2b, nu3b, T1, T2, T3 = params
248     xx = Numerics.default_grid(pts)
249     phi = PhiManip.phi_1D(xx)
250     phi = PhiManip.phi_1D_to_2D(xx, phi)
251     phi = Integration.two_pops(phi, xx, T1, nu1a, nu2a=nuA, m12=0, m21=0)
252     phi = PhiManip.phi_2D_to_3D_split_2(xx, phi)
253     phi = Integration.three_pops(phi, xx, T2, nu1a, nu2a, nu3a, m12=0, m21=0, m23=0, m32=0, m13=0, m31=0)
254     phi = Integration.three_pops(phi, xx, T3, nu1b, nu2b, nu3b, m12=0, m21=0, m23=0, m32=0, m13=0, m31=0)
255     fs = Spectrum.from_phi(phi, ns, (xx,xx,xx))
256     return fs
257
```

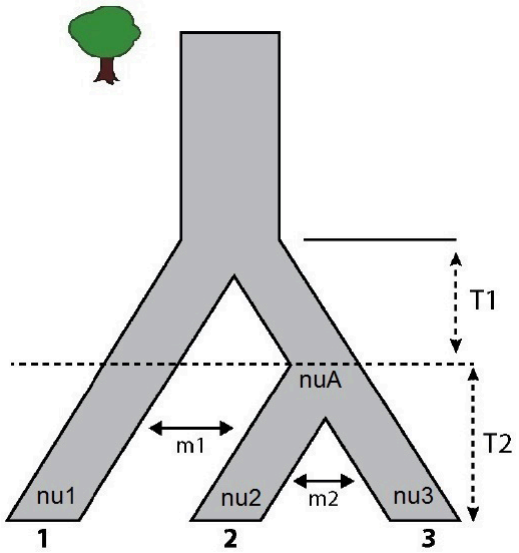


Split, no migration
size change

Demographic model selection - ðaði

- Again, this model is very similar to the first, but added migration after an initial period of isolation changes the interpretation of what is happening. We won't go through all the models but spend a few mins looking at the model code and how they differ from one another, and think about how to apply this to your own study system!

```
def refugia_2(params, ns, pts):
    """
    Model with split between pop 1 and (2,3), gene flow does not occur. Split between pops
    2 and 3, with gene flow. After appearance of 2 and 3, gene flow also occurs between 1
    and 2.
    shorter isolation
    """
    #8 parameters
    nu1, nuA, nu2, nu3, m1, m2, T1, T2 = params
    xx = Numerics.default_grid(pts)
    phi = PhiManip.phi_1D(xx)
    phi = PhiManip.phi_1D_to_2D(xx, phi)
    phi = Integration.two_pops(phi, xx, T1, nu1=nu1, nu2=nuA, m12=0, m21=0)
    phi = PhiManip.phi_2D_to_3D_split_2(xx, phi)
    phi = Integration.three_pops(phi, xx, T2, nu1=nu1, nu2=nu2, nu3=nu3, m12=m1, m21=m1, m23=m2, m32=m2, m13=0, m31=0)
    fs = Spectrum.from_phi(phi, ns, (xx,xx,xx))
    return fs

```



Refugia model 2, shorter isolation

Demographic model selection - ∂a∂i

- A v. important part of each script is telling the optimization routine some information… This needs to be kept consistent across models if you are later comparing them (except the model specific parameters such as p_labels, upper/lower)

```
pts = [50,60,70]
p_labels = "nu1, nuA, nu2, nu3, mA, m1, m2, T1, T2, T3"
upper = [20,20,20,20,20,20,20,15,15,15]
lower = [0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.1,0.1,0.1]
reps = [5,7,10]
maxiters = [3,3,5]
folds = [3,2,1]

for i in range(1,2):
    prefix = "full_3D_dadi_job_{}".format(i)
    Optimize_Functions.Optimize_Routine(fs, pts, prefix, "ancmig_1", ancmig_1, 3, 10,  param_labels = p_labels, in_upper=upper, in_lower=lower, reps = reps, maxiters = maxiters, folds = folds)
```

- pts − extrapolation dimensions for grid size
- p_labels − parameter labels
- Upper/lower − bounds the parameters are allowed to range from
- reps − how many replicates across each round of optimization for each model ?(I reduced this here for runtime − I would recommend at least 10, 20, 25)
- maxiters - maximum iterations of each replicate to do (again I reduced this, I would set it to 3,5,10 for example)
- folds − controls peturbation of input parameters across rounds

Demographic model selection - ∂a∂i

- To run all of these models in parallel I provided a bash script – **run_dadi_jobs.sh**. This needs to be submitted with bash (so bash run_dadi.jobs.sh)

- This is submitting a job for each model (all .sh scripts in submit_scripts/dadi which are in turn calling the python scripts for each model in Inputs/dadi)

- Some of these take >1 day, even with reduced replicates – your own data will take several weeks for complex 3D models!

```
#!/bin/bash

cd /public/barratt/submit_scripts/dadi/

qsub dadi_Run_Optimizations_ancmig_1.sh
qsub dadi_Run_Optimizations_ancmig_2.sh
qsub dadi_Run_Optimizations_ancmig_3.sh
qsub dadi_Run_Optimizations_refugia_1.sh
qsub dadi_Run_Optimizations_refugia_2.sh
qsub dadi_Run_Optimizations_refugia_3.sh
qsub dadi_Run_Optimizations_sim_split_no_mig.sh
qsub dadi_Run_Optimizations_split_no_mig_human.sh
qsub dadi_Run_Optimizations_sim_split_refugia_sym_mig_adjacent.sh
qsub dadi_Run_Optimizations_sim_split_refugia_sym_mig_all.sh
qsub dadi_Run_Optimizations_sim_split_sym_mig_adjacent.sh
qsub dadi_Run_Optimizations_sim_split_sym_mig_all.sh
qsub dadi_Run_Optimizations_split_no_mig.sh
qsub dadi_Run_Optimizations_split_sym_mig_adjacent.sh
qsub dadi_Run_Optimizations_split_sym_mig_all.sh
```

Demographic model selection - ∂a∂i

- For each model there will be an _optimized.txt file output when it is done

- This will contain all of the replicates across all rounds of optimization, with log likelihoods, AIC scores, some other statistics and the optimized estimated parameters per replicate

Model	Replicate	log-likelihood	AIC	chi-squared	theta	optimized_params(nu1, nuA, nu2, nu3, mA, m1, m2, T1, T2, T3)
ancmig_1	Round_1_Replicate_1	-822.49	1664.98	15486.09	36.02	1.4327,0.6762,2.0853,2.487,0.285,2.8134,1.5381,3.3373,2.0472,0.1831
ancmig_1	Round_1_Replicate_2	-981.15	1982.3	10950.46	95.41	1.172,0.2678,0.289,0.3181,1.424,1.0963,0.5437,0.8807,0.6363,0.2788
ancmig_1	Round_1_Replicate_3	-1388.38	2796.76	615127.54	27.68	1.2769,0.1334,7.3991,1.0693,0.5338,1.2723,0.1848,0.6738,0.4646,2.8218
ancmig_1	Round_1_Replicate_4	-1781.81	3583.62	403669.59	68.51	0.1704,6.8577,0.7507,2.1848,0.5484,0.3818,3.3138,0.1929,1.3982,0.6133
ancmig_1	Round_1_Replicate_5	-832.4	1684.8	26787.77	41.21	1.5404,0.153,2.2691,0.7042,2.6457,4.1273,0.1454,0.5548,3.362,0.192
ancmig_1	Round_2_Replicate_1	-782.75	1585.5	8354.27	27.82	2.184,1.5032,5.4246,0.752,0.9313,3.0463,1.3318,3.2099,3.329,0.3844
ancmig_1	Round_2_Replicate_2	-899.21	1818.42	32566.73	45.83	0.8096,0.3784,6.5337,0.715,1.0009,5.8112,0.6821,1.366,2.0611,0.1326
ancmig_1	Round_2_Replicate_3	-982.12	1984.24	2634.0	34.2	0.8993,1.7669,1.2152,6.7562,0.4664,4.0625,0.5983,13.1654,1.3945,0.2537
ancmig_1	Round_2_Replicate_4	-986.35	1992.7	2921.7	49.48	0.5177,0.4968,1.4259,1.9057,0.0779,10.3514,5.5614,7.0213,1.8647,0.2151
ancmig_1	Round_2_Replicate_5	-900.9	1821.8	416748.92	15.84	3.0765,0.6176,7.4957,3.9975,0.6555,1.3272,0.6002,8.426,6.6116,0.2179
ancmig_1	Round_2_Replicate_6	-828.94	1677.88	1291090.33	34.82	1.9682,0.7646,4.3289,2.3223,0.7518,6.9464,1.0897,1.0743,1.2241,0.2029
ancmig_1	Round_2_Replicate_7	-863.76	1747.52	2169.39	45.03	0.9384,1.7474,1.2972,2.8293,0.2616,4.9147,1.8259,0.9694,1.5942,0.4525
ancmig_1	Round_3_Replicate_1	-764.37	1548.74	9416.07	21.46	3.727,2.5675,7.3765,0.7332,0.9901,1.7411,1.592,4.5603,2.6908,0.5703
ancmig_1	Round_3_Replicate_2	-778.81	1577.62	1625.25	33.24	1.4057,0.8069,3.0523,0.9958,0.8666,2.4718,0.9718,2.9808,4.767,0.391
ancmig_1	Round_3_Replicate_3	-825.93	1671.86	7436.22	28.78	2.3032,2.8627,3.6239,0.446,0.5611,1.626,1.0326,5.785,4.5529,0.2557
ancmig_1	Round_3_Replicate_4	-845.84	1711.68	8122.92	29.56	2.2393,1.4579,3.5153,0.4021,1.6669,1.7399,0.7509,2.6914,6.8319,0.2742
ancmig_1	Round_3_Replicate_5	-759.52	1539.04	3140.59	26.91	2.3009,0.9256,6.2178,0.7727,0.656,1.5611,1.3419,2.3177,3.3661,0.5194
ancmig_1	Round_3_Replicate_6	-916.28	1852.56	3638.04	27.41	1.2604,2.6609,8.2263,0.7551,1.1297,4.8528,0.8069,2.5213,4.9567,0.5853
ancmig_1	Round_3_Replicate_7	-904.7	1829.4	4633.93	38.58	1.0946,1.4773,5.1077,0.3784,1.5624,2.4707,1.1402,3.9725,2.6012,0.3267
ancmig_1	Round_3_Replicate_8	-695.22	1410.44	2195.01	27.92	3.2999,1.654,2.9113,0.9562,0.6359,3.811,1.6588,2.8929,2.3593,0.6909
ancmig_1	Round_3_Replicate_9	-756.91	1533.82	178087.77	27.96	3.1851,1.9895,3.264,0.9623,1.6789,6.2076,0.7072,2.0413,2.6597,0.3178
ancmig_1	Round_3_Replicate_10	-762.02	1544.04	2420.8	27.03	1.9484,1.5928,5.0203,1.1238,1.2369,2.5935,1.8578,4.1781,3.3041,0.4667

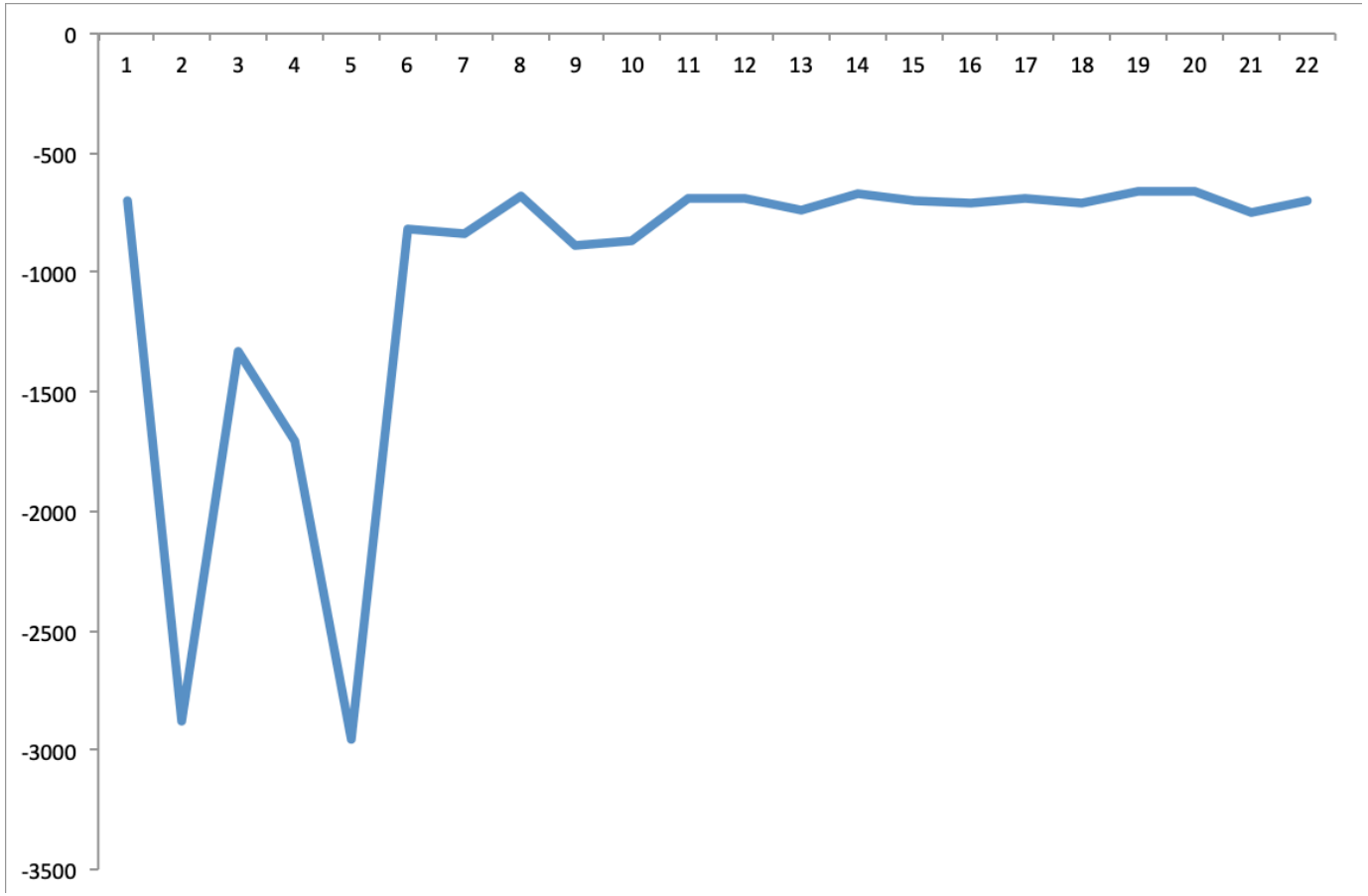
Demographic model selection - ∂a∂i

- The log likelihoods should stabilise across optimization rounds – what the scripts automatically do is supply the parameters from the best scoring replicate of the first round into the second round and then into subsequent rounds so it is always starting the likelihood search from an ‘optimized’ position

- You can check the log likelihoods across replicates to see if the log likelihoods have converged for each model. They should become more stable as you do more rounds of optimization

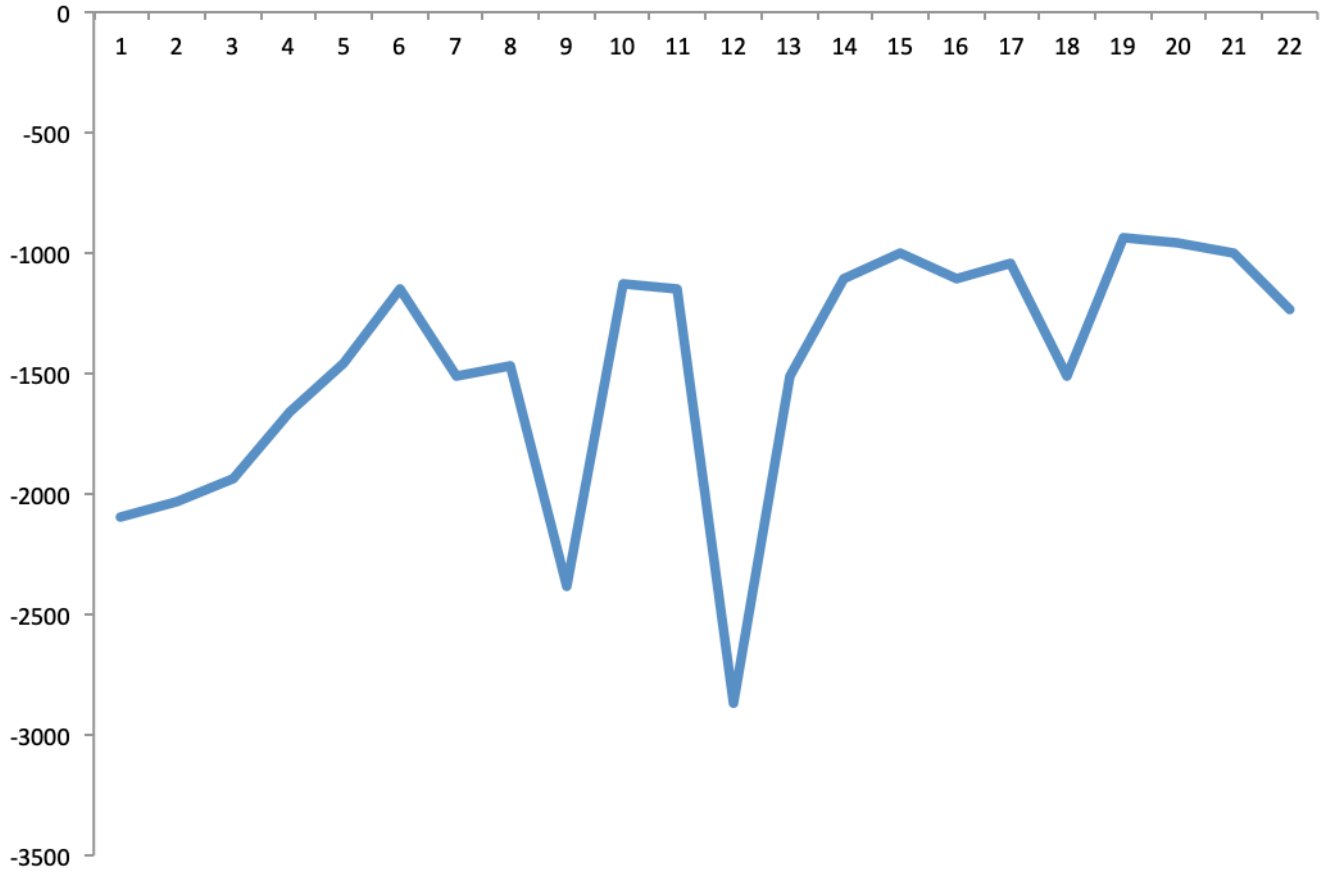
Demographic model selection - ðaði

- Ideally something like this (split_no_mig)
- First 5 are round 1, second 7 are round 2, final 10 are round 3



Demographic model selection - ðaði

- This one (anc_mig_3) not quite as trustworthy….
- If you run all models out long enough they should all converge on stable(ish) log-likelihoods eventually



Demographic model selection - ∂a∂i

- Finally, to see which models you have tested are the best fit to the data take the best AIC model for each tested model
- That alone is informative between models, but you should verify this result using Akaike weights
- I did this next step in excel, just because it's easy and fast… paste your best scoring replicate from each model into excel, so here you have 14 models (one of my models failed…)

	A	B	C	D	E	F	G	H	I	J	K	L
1	Model	Replicate	log-likelihood	AIC	chi-squared	theta	optimized_params(nu1, nuA, nu2, nu3, mA, m1, m2, m3, T1, T2)					
2	split_no_mig	Round_3_Replicate_7	-662.82	1337.64	2454.19	61.14	3.4995,1.9552,1.6082,0.3107,0.3913,0.2069					
3	split_symmig_adjacent	Round_3_Replicate_8	-660.17	1338.34	1799.88	129.42	0.7013,0.0936,0.5126,0.7409,2.1294,0.6855,1.1817,0.7621,0.2746					
4	ancmig_2	Round_3_Replicate_7	-685.75	1385.5	1682.51	43.72	2.3311,1.3484,2.139,0.6952,1.6586,2.0598,0.6345					
5	ancmig_1	Round_3_Replicate_8	-695.22	1410.44	2195.01	27.92	3.2999,1.654,2.9113,0.9562,0.6359,3.811,1.6588,2.8929,2.3593,0.6909					
6	sim_split_no_mig	Round_3_Replicate_2	-710.28	1428.56	2667.44	113.83	0.9366,0.7251,0.2751,0.1959					
7	sim_split_refugia_sym_mig_adjacent	Round_3_Replicate_7	-710.4	1434.8	3918.59	23.95	6.3932,2.9662,0.4447,0.1512,0.3246,3.2925,0.6905					
8	split_symmig_all	Round_3_Replicate_1	-720.42	1460.84	3161.87	151.47	0.4504,0.0758,0.375,0.566,1.973,2.5037,0.9819,0.0826,0.7429,0.4305					
9	sim_split_refugia_sym_mig_all	Round_3_Replicate_10	-777.89	1571.78	188976.08	49.89	2.8305,0.9518,2.5363,0.6463,0.1688,0.5421,0.7801,0.1794					
10	sim_split_sym_mig_adjacent	Round_3_Replicate_2	-824.64	1661.28	16388.56	56.61	1.0694,2.2152,7.1579,0.3934,0.129,0.589					
11	refugia_1	Round_3_Replicate_4	-846.3	1710.6	1836.86	68.1	1.3426,9.8208,0.3138,0.2858,1.1829,0.6509,1.8844,0.1958,0.8112					
12	refugia_3	Round_3_Replicate_3	-859.42	1738.84	4027.72	69.35	1.3113,1.2845,0.2567,0.5708,0.1326,1.8529,2.968,0.6844,1.772,0.2489					
13	sim_split_sym_mig_all	Round_3_Replicate_10	-866.9	1747.8	4144.45	73.09	0.6566,6.1538,0.1749,0.4385,2.226,0.1906,0.7316					
14	ancmig_3	Round_3_Replicate_7	-941.04	1898.08	7796.43	57.91	1.0675,4.6912,0.648,0.4322,0.0731,1.1811,0.4536,0.1655					
15	refugia_2	Round_3_Replicate_1	-964.07	1944.14	49913.56	60.33	1.5485,0.3184,0.1551,3.2688,0.796,3.3901,0.1248,10.8014					
16	split_no_mig_human											

Demographic model selection - ∂a∂i

- Insert 3 empty columns somewhere (doesn't particularly matter where) and call these something informative
- Calculate the delta AIC for each model (the AIC minus the best scoring AIC across all models)
- Calculate the relative likelihood for each model =EXP(-0.5*delta_AIC)
- Calculate the Akaike weight for each model (= relative likelihood/ sum of all relative likelihoods)

Demographic model selection - ∂a∂i

- Your excel sheet should look like below, if so then you've successfully ranked your models using Aikake weights based on Akaike Information Criterion (Burnham and Anderson, 2002)

- Now you can definitively say which of the hypotheses is most supported by your empirical data!

- But make sure you run the models long enough (this was too short here, even though some were running for a day)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Model	Replicate	log-likelihood	AIC	chi-squared	theta	delta_AIC	relative likelihood	wi	optimized_params(nu1, nuA, nu2, nu3, mA, m1, m2, m3, T1, T2)					
2	split_no_mig	Round_3_Replicate_7	-662.82	1337.64	2454.19	61.14	**0**	**1**	**0.586617579**	3.4995,1.9552,1.6082,0.3107,0.3913,0.2069					
3	split_symmig_adjacent	Round_3_Replicate_8	-660.17	1338.34	1799.88	129.42	**0.7**	**0.70468809**	**0.413382421**	0.7013,0.0936,0.5126,0.7409,2.1294,0.6855,1.1817,0.7621,0.2746					
4	ancmig_2	Round_3_Replicate_7	-685.75	1385.5	1682.51	43.72	**47.86**	**4.04886E-11**	**2.37513E-11**	2.3311,1.3484,2.139,0.6952,1.6586,2.0598,0.6345					
5	ancmig_1	Round_3_Replicate_8	-695.22	1410.44	2195.01	27.92	**72.8**	**1.55482E-16**	**9.12086E-17**	3.2999,1.654,2.9113,0.9562,0.6359,3.811,1.6588,2.8929,2.3593,0.6909					
6	sim_split_no_mig	Round_3_Replicate_2	-710.28	1428.56	2667.44	113.83	**90.92**	**1.80706E-20**	**1.06005E-20**	0.9366,0.7251,0.2751,0.1959					
7	sim_split_refugia_sym_mig_adjacent	Round_3_Replicate_7	-710.4	1434.8	3918.59	23.95	**97.16**	**7.97947E-22**	**4.6809E-22**	6.3932,2.9662,0.4447,0.1512,0.3246,3.2925,0.6905					
8	split_symmig_all	Round_3_Replicate_1	-720.42	1460.84	3161.87	151.47	**123.2**	**1.76791E-27**	**1.03709E-27**	0.4504,0.0758,0.375,0.566,1.973,2.5037,0.9819,0.0826,0.7429,0.4305					
9	sim_split_refugia_sym_mig_all	Round_3_Replicate_10	-777.89	1571.78	188976.08	49.89	**234.14**	**1.43597E-51**	**8.42365E-52**	2.8305,0.9518,2.5363,0.6463,0.1688,0.5421,0.7801,0.1794					
10	sim_split_sym_mig_adjacent	Round_3_Replicate_2	-824.64	1661.28	16388.56	56.61	**323.64**	**5.27797E-71**	**3.09615E-71**	1.0694,2.2152,7.1579,0.3934,0.129,0.589					
11	refugia_1	Round_3_Replicate_4	-846.3	1710.6	1836.86	68.1	**372.96**	**1.02983E-81**	**6.04116E-82**	1.3426,9.8208,0.3138,0.2858,1.1829,0.6509,1.8844,0.1958,0.8112					
12	refugia_3	Round_3_Replicate_3	-859.42	1738.84	4027.72	69.35	**401.2**	**7.59499E-88**	**4.45535E-88**	1.3113,1.2845,0.2567,0.5708,0.1326,1.8529,2.968,0.6844,1.772,0.2489					
13	sim_split_sym_mig_all	Round_3_Replicate_10	-866.9	1747.8	4144.45	73.09	**410.16**	**8.60771E-90**	**5.04943E-90**	0.6566,6.1538,0.1749,0.4385,2.226,0.1906,0.7316					
14	ancmig_3	Round_3_Replicate_7	-941.04	1898.08	7796.43	57.91	**560.44**	**2.0045E-122**	**1.1759E-122**	1.0675,4.6912,0.648,0.4322,0.0731,1.1811,0.4536,0.1655					
15	refugia_2	Round_3_Replicate_1	-964.07	1944.14	49913.56	60.33	**606.5**	**1.9962E-132**	**1.171E-132**	1.5485,0.3184,0.1551,3.2688,0.796,3.3901,0.1248,10.8014					
16	split_no_mig_human							**1.70468809**							

Demographic model selection - ∂a∂i

- When you have your best model you might want to plot the model SFS against the empirical data. You can do this with the **plot_dadi_best_model.sh** script

- This is calling Make_Plots.py and some functions in Plotting_Functions.py in the Inputs/dadi folder

- Plotting_Functions doesn't ever need editing but you need to pass the correct information to Make_Plots.py…

```
67  #=============================================================================
68  #get snps file
69  #***************
70  snps = "/public/barratt/work/Lflavomaculatus/Inputs/dadi/dadi_3pops_north_south_taratibu_snps.txt"
71
72  #Create python dictionary from snps file
73  dd = dadi.Misc.make_data_dict(snps)
74
75  #***************
76  #projection sizes, in ALLELES not individuals
77  proj = [36,29,15]
78  #pop_ids is a list which should match the populations headers of your SNPs file columns
79  pop_ids=['north','south','taratibu']
80
```

Demographic model selection - ∂a∂i

- These lines are where you specify which model(s) results to plot, including the optimized parameters and outputfile

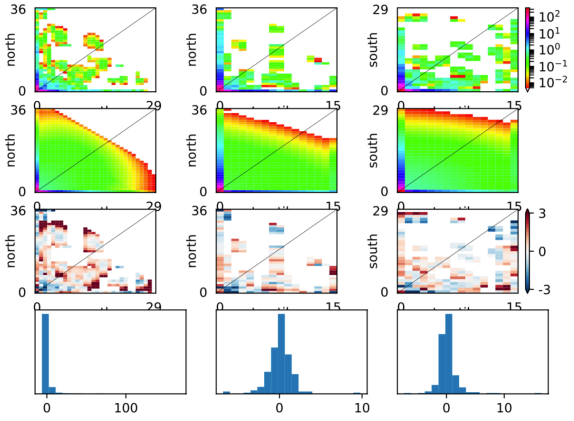
```
#************** "split_nomig"
# 6 Values
split_nomig_params = [3.4995,1.9552,1.6082,0.3107,0.3913,0.2069]


#**************
#Input some of the basic reusable arguments here
pts = [50,60,70]
fs = fs_1
outfile = "north_south_taratibu"


#================================================================================
# returned_model = Three_Pop_Plot(pts, fs, model_name, params):

# Each model is executed with one replicate using fixed parameter values.
# The simulated model is stored as an object to be called on in the
# subsequent plotting function.

#**************
#Models from the Models_3D.py script
split_nomig = Three_Pop_Plot(pts, fs, "split_nomig", split_nomig_params)

#================================================================================================
#write a plotting function for data and model comparison

def plot_all(sim_model, data, outfile, model_name):
    print '{0}_{1}.pdf'.format(outfile,model_name), '\n'
    outname = '{0}_{1}.pdf'.format(outfile,model_name)
    fig = pylab.figure(1)
    fig.clear()
    dadi.Plotting.plot_3d_comp_multinom(sim_model, data, resid_range = 3, vmin=0.005)
    fig.savefig(outname)

```

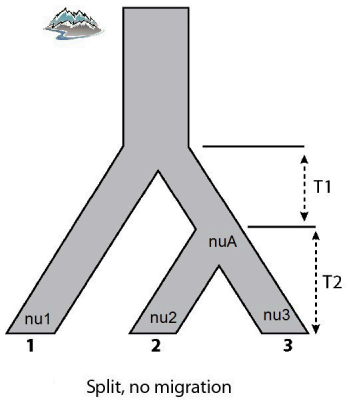
Demographic model selection - ∂a∂i

- Your output will look something like below. Top row are the data itself (3 plots as it is 3 pop comparisons), second row is the model

- Third row and fourth are residuals − model predicts too many or too few alleles in a given cell (these should be close to 0 mostly)

- You can see here that this is not yet a great fit (model was not run long enough to be sufficiently optimised… this is essentially the best of a bad bunch!). Remember that the model is only an approximation of reality, the most likely but not yet the truth!





Split, no migration

Recap

- You've learned how to (begin to) make sense of your short-read data:

- 2 separate population structure analyses (Admixture and DAPC)
- 2 methods to look at evolutionary relationships (RAxML and SNAPP)
- 1 method for investigating barriers and connectivity (EEMS)
- 1 method for investigating likely demographic history (∂a∂i)

- These are just some of the tools, there are many many more which address similar questions….

Recap

- These are just some of the tools which I use, there are many many more which can address similar questions…

- Stacks, iPyrad, dDocent, ANGSD
- Admixture, DAPC, Structure, SNMF, NGSAdmix, fastStructure, fineStructure, fineRADstructure
- RAxML, SNAPP (BEAST), SVDquartets, Astral2, SNPhylo, SnpTree
- EEMS, MAPS, TreeMix
- ∂a∂i, FastSimcoal, Moments

Now go and do something that makes you happy!