

# MIGRATE tutorial

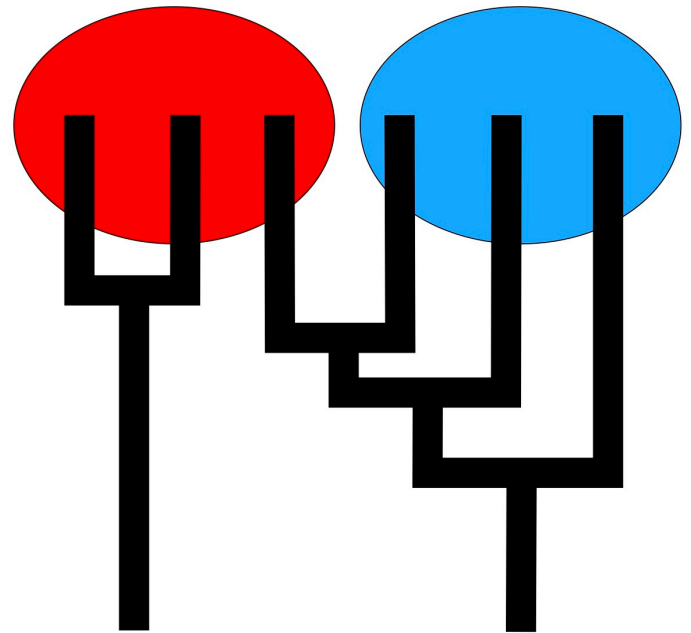
*Peter Beerli, Florida State University, April 2017*

This tutorial has three sections:

1. Short overview of the program
2. How do we specify models
3. Run a model selection exercise

Download the necessary files from [here](#) and unpack the `migrate_tutorial2017.zip` file. You will now have a directory `migrate_tutorial`

I have added binaries for macs and windows into the directory, if you run a unix system you will need to download the source for version 4.2.14 from [here](#)



## Overview of the program

---

[15 minutes] **migrate** is a Bayesian population genetics inference program that is based on the structured coalescence. It uses DNA/RNA sequence data or microsatellite data. It can run on computer clusters (using parallel software tools based on the MPI standard). It is distributed as opensource (MIT license) from [Migrate's main website](#).

**migrate** can estimate parameters, such as populations sizes, immigration rates, and divergence and admixture times of fairly complex models. Originally, it was a pure population genetics programs, but with the introduction of divergence it can handle closely related species, admixed populations including potential hybrids, as long as your data include multiple individuals of the same group (this tutorial will introduce you to these models).

## Efficient running on your system

**migrate** is distributed in source code and binaries for Mac and Windows computers. To run **migrate** efficiently on linux clusters compile it yourself using the MPI libraries openmpi or mvapich2. The Windows binary can run on Windows 7-10, although it may not be the most efficient way to run the program. If you have multiple loci and a computer with multiple cores (>4) use the parallel version (unfortunately you will need to learn some specific commands to run migrate in parallel, look into the instructions of openmpi or mvapich2 for that).

## Menu and starting the program

**migrate-n** uses a menu to operate, the menu can save all options into a file which is called by default *parmfile*. The parmfile is a textfile that can be edited with a text editor, all lines starting with a # are comments and try to explain the options. You can write your own comments into the parmfile, but these will be overwritten when saved again through the menu. Starting the program can be done like this

```
migrate-n
migrate-n parmfile
migrate-n parmfile -nomenu
migrate-n -version
migrate-n -help
```

If the program is started without the *-nomenu* option it displays the main menu

```
+++++
+                                     +
+  POPULATION SIZE, MIGRATION, DIVERGENCE, ASSIGNMENT, HISTORY  +
+  Bayesian inference using the structured coalescent           +
+                                     +
+++++
Using Intel AVX (Advanced Vector Extensions)
Compiled for a SYMMETRIC multiprocessors (GrandCentral)
PDF output enabled [Letter-size]
Version 4.2.8   [June-24-2016]
Program started at   Fri Jul   8 23:18:24 2016

=====
MAIN MENU
=====

D      Data type currently set to: DNA sequence model
I      Input/Output formats and Event reporting
P      Parameters  [start, migration model]
S      Search strategy
W      Write a parmfile
Q      Quit the program

To change the settings type the letter for the menu to change
Start the program with typing Yes or Y
==>
```

Each submenu will handle options concerning Data, Run parameters, and Population model among other

things.

## Running

The MPI version is printing out less information than the single computer version, here an example of a log while running the single-cpu version:

```
23:38:13 Burn-in of 200000 steps (Locus: 1/2, Replicate: 1/5)
23:38:15 Burn-in complete in 22 seconds
23:39:00 Sampling of 100000 steps (Locus: 1/2, Replicate: 1/5)
23:39:15 Sampling complete in 181 seconds
23:39:33 Sampling: prognosed end of run is 23:55 July 08 2016 [0.077777 done])
Parameter      Acceptance Current      PropWindow AutoCorr ESS
-----
Theta_1         0.085      0.00008      0.016    0.520    211.18
Theta_2         0.110      0.00211      0.024    0.515    214.11
M_2->1         0.759     982.58556     85.061    0.956     15.08
Genealogies     0.100    -3210.29272      ---      0.781     82.36
```

## Output

The output will be printed into two files that are called by default *outfile* and *outfile.pdf* (the name can be changed through the menu). There is a example outfile.pdf [here](#).

## How do we specify models

[*about 15 minutes*] **migrate** uses an adjacency matrix to define the interactions among populations.

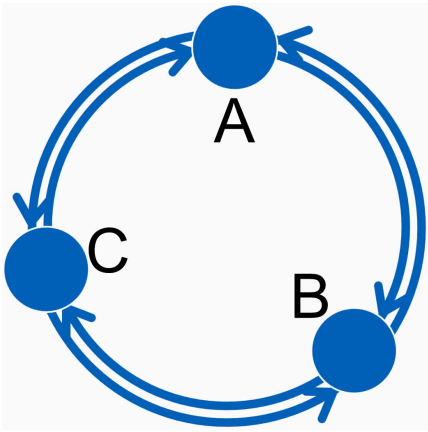
## Defaults

The defaults in **migrate** are to estimate all population sizes  $\Theta$  and all immigration rates  $M$  independently.

For example a 3-population model where all populations A, B, and C exchange migrants at individual rates can be specified like this:

	A	B	C
A	x	x	x
B	x	x	x
C	x	x	x

Of course, this is not all that informative, but it is the default.



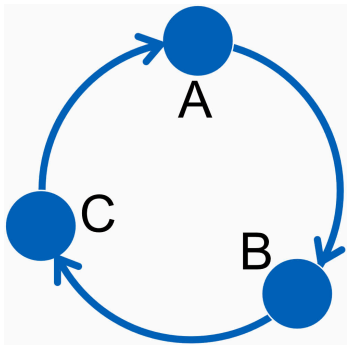
## Specifying models without divergence

There are many ways to specify different structured models, **migrate** has a set of options to help with that:

- x or ‘\*’ means that this parameter is estimated
- 0 means that this parameter is not estimated (works only on off-diagonal elements)
- s means the migration parameter is symmetric, this implies that  $M_{1 \rightarrow 2} = M_{2 \rightarrow 1}$
- c means that the value is constant and not estimated, the values needs to be specified in the the start-parameter option
- m means take the average of all parameters labeled m,  $\Theta$  and  $M$  will be treated separately. A matrix full of m will be a two- parameter model (all populations share the same values for 1  $\Theta$  and 1  $M$ ).
- there are upper case variants for s and m (S, M) that use instead of the migration parameter  $M$  the number of immigrants  $Nm$ . In most cases this is not a good choice because  $\Theta$  and  $Nm$  are more strongly correlated than  $\Theta$  and  $M$ .
- all other lower alphabet letters except c, d, t, m, s, t, and x can be used to form groups of parameters (similar to m but more flexible), see example.

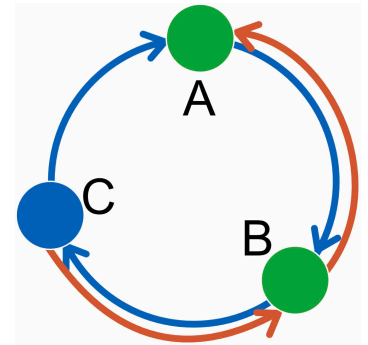
More examples:

	A	B	C
A	x	0	x
B	x	x	0
C	0	x	x





	A	B	C
A	e	b	a
B	a	e	b
C	0	a	x



**REMINDER:** the populations specified on the row of the adjacency matrix are the receiving populations, populations on the columns are the sending populations, thus in the circular unidirectional stepping stone example above, population A receives migrants from population C and we estimate the parameter  $M_{C \rightarrow A}$ , population B receives migrants from population A and we estimate  $M_{A \rightarrow B}$  etc.

**Answer these questions**

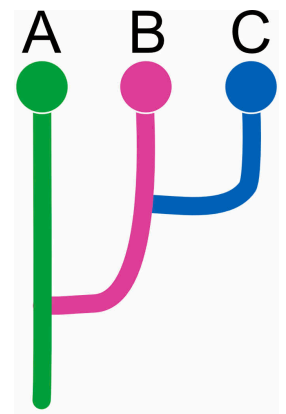
Click on the question to reveal the answer, but write down the answer before you peek :-).

- 1. specify the matrix for a 2-population system with unidirectional migration from 2 → 1
- 2. specify a migration matrix for a 5-population system with where population 1 and 5 are on a mainland, population 2 is and island close to 1, population 4 is close to 5, and population 3 is far out in the sea but closest to 2. ‘Close’ means reachable by rafting, and once on an island it will be difficult to get off again.

**Specifying models with divergence**

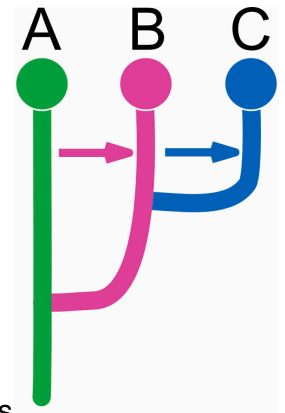
For the divergence specification we also use the adjacency matrix. For example if we have a model where the populations do not exchange migrants we could specify the matrix like this

	A	B	C
A	x	0	0
B	d	x	0
C	0	d	x



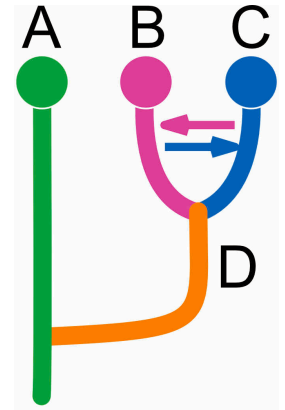
- d means divergence from the ancestral column population
- D means divergence from the ancestral column population with onpoing immigration
- t means divergence time is the same for all the matrix elements with the t.

	A	B	C
A	x	0	0
B	D	x	0
C	0	D	x



Migration is problematic and models with immigration and divergence may need additional ancestral populations to allow the estimation to be identifiable. Here an example that needs ancestral population to get better estimates of divergence and migration.

	A	B	C	D
A	x	0	0	0
B	0	x	x	t
C	0	x	x	t
D	d	0	0	x

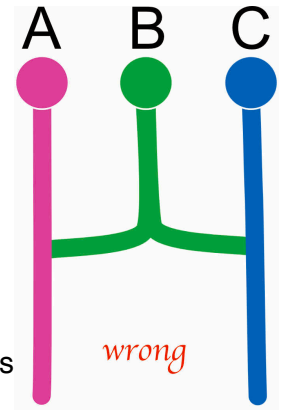


**migrate** has a menu to enter this adjacency matrix, but given the many options it may seem easier for complex scenarios to edit the parameter file (it is a text file) directly. the option is named *custom-migration* and contains the adjacency matrix, for example the example from above looks like this

```
custom-migration={ x000 0xxt 00xxt d00x}
```

If a diagonal element is set to zero the program will crash! There are also models that will not work well or fail. All models essentially need to be able to end in one common ancestor, for example we may be tempted to code an admixture example like this

	A	B	C
A	x	0	0
B	d	x	d
C	0	0	x

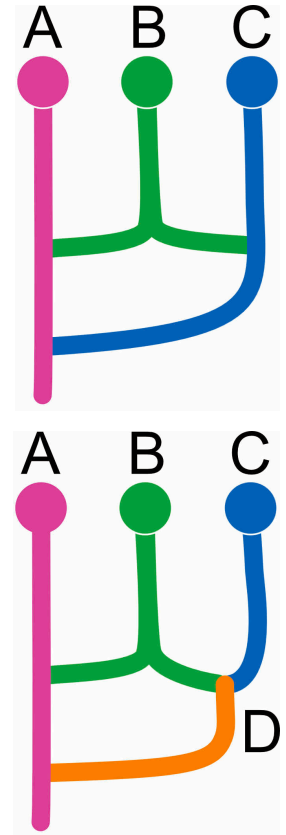


but this will not work well because we force **to have two roots**, and this can lead to failures during the run, one can code this better as this (a) or this (b)

(a)	A	B	C
A	x	0	0
B	d	x	d
C	d	0	x

More complicated than (a) is this (b) model that forces a sequence of divergences. Both models should deliver similar values so, although (b) will estimate a population size for population D.

(b)	A	B	C	D
A	x	0	0	0
B	d	x	d	0
C	0	0	x	d
D	d	0	0	x



### Answer these questions

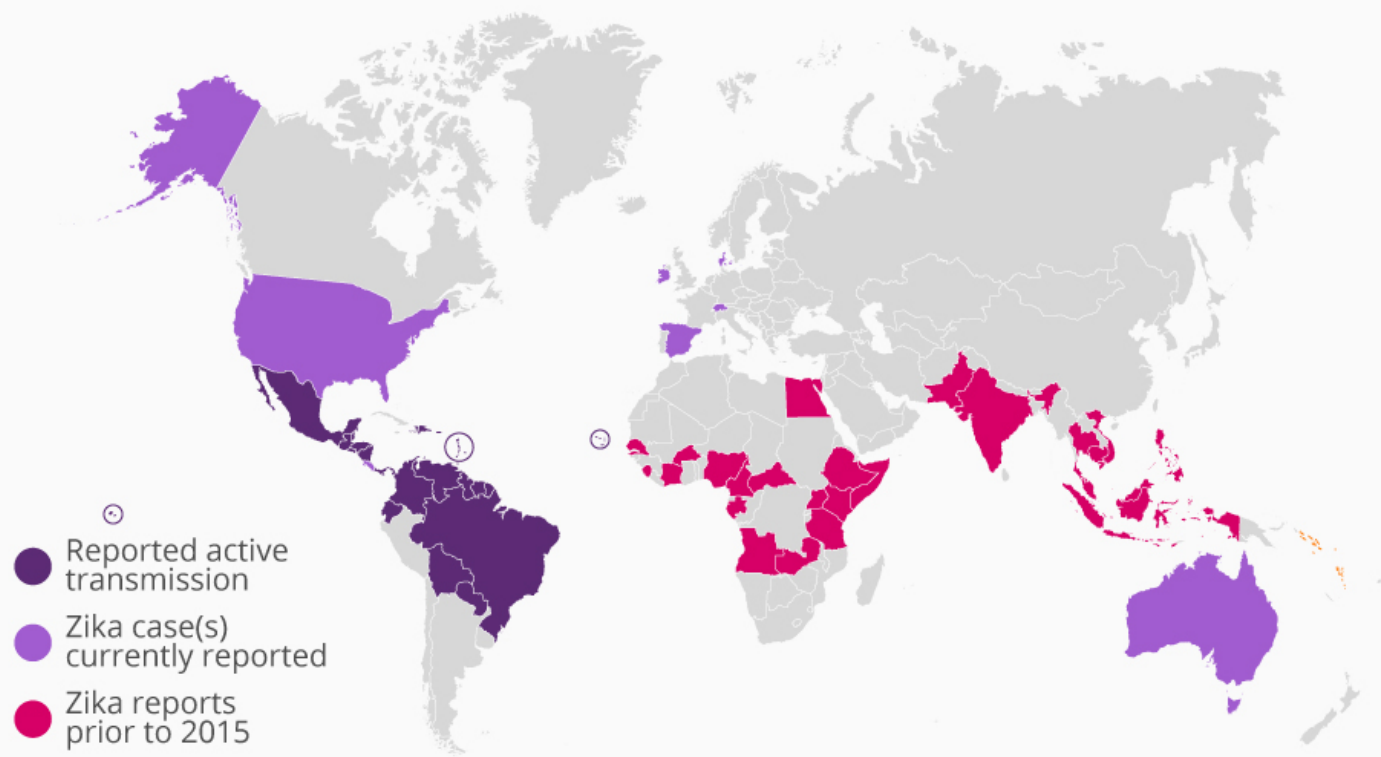
- 1. specify the matrix for two current populations that share a common ancestor
- 2. If the first questions is answered using 'd' what does that mean for the divergence times?

## Run a model selection exercise

[*about 1.5 hours*] We use a real dataset of the zika virus, the data was gathered from the [VIPR database](#). The localities include Africa, China, Brazil, Panama, Guatemala, Mexico, and Puerto Rico. There are only few sequences in some of the populations so they will be pooled in our analysis. To simplify we pool Panama, Guatemala, Mexico, and Puerto Rico into a population labeled 'Mexico'. The general consensus about the spread of zika is from Africa to Asia to the Americas. They mainly used incidence data to come to that conclusion. Here we use complete genomic data of the viruses to select among models that represent alternative hypotheses.

# The Spread Of The Zika Virus

Countries and territories with active Zika virus transmission\* and reported cases



Source: Centers for Disease Control and Prevention  
\*As of February 2016

statista

1. **[to the east and north]** Zika spread from Africa → Asia → Brazil → Mexico
2. **[to the east and north with migration]** Zika spread from Africa → Asia → Brazil → Mexico with recurrent migration from the source after the population split
3. **[panmictic]** The spread of Zika so fast and ubiquitous that we can treat Zika as a single panmictic population
4. **[to the west]** Zika spread from Africa → Asia and independently from Africa → Brazil → Mexico
5. **[migration]** Zika follows an *n-island model*
6. **[to the east and splam]** Zika spread from Africa → Asia → Americas (and we treat all new world locations as a panmictic unit.
7. **[twoways]** The population in the Americas is a 'hybrid' from Africa and Asia.
8. Your own model

For each of the models you will need to edit a parmfile. I suggest that you follow the naming schemes in this tutorial.

There is a *parmfile.template* that specifies model 1 **[to the east and north]**. Use this template to form all the

other models, for each these new models you will need to do the following steps:

1. copy the parmfile.template to a new parmfile
2. check and fix the population relabel option
3. and/or check the custom-migration option

## Model 1 to the east and north

```
#inspect the parmfile_eastnorth  
migrate-n parmfile_eastnorth -nomenu
```

## Model 2 to the east and north with migration

```
#inspect parmfile_eastnorthmig  
migrate-n parmfile_eastnorthmig -nomenu
```

the parmfile needs to have a different *custom-migration* setting, search for the option, compare this “migration” model with the Model 1.

► What is the difference between Model 1 and Model 2?

## Model 3 panmictic

```
#inspect parmfile_panmictic  
migrate-n parmfile_panmictic -nomenu
```

In your editor search for the option **population-relabel** it should look like similar to this

```
# MBL tutorial dataset zika
# in the data file the order of the population is this:
# Africa
# Asia
# Brazil
# Mexico
# Puerto Rico
#
#   population-relabel={assignment for each location in the infile}
#   example is population-relabel={1 2 2}
# for Model 1 and model 2 it will look like this
# population-relabel={1, 2, 3, 4, 4}
#for Model 3 (panmictic)
population-relabel={1, 1, 1, 1, 1}
```

## Model 4 to the west

```
#inspect parmfile_west
migrate-n parmfile_west -nomenu
```

the parmfile needs to have a different *custom-migration* setting, it is adjusted so that we have population splits like this: Africa → Asia, Africa → Brazil → 'Mexico'.

## Model 5 migration only (and not divergence)

```
#inspect parmfile_migration
migrate-n parmfile_migration -nomenu
```

We use a directional migration matrix in *custom-migration*, so that migration goes from Africa → Asia → Brazil → 'Mexico'.

## Model 6 to the east and splam

```
#inspect parmfile_eastsplam
migrate-n parmfile_eastsplam -nomenu
```

this parmfile needs to have a different *custom-migration* setting and a change of the *population-relabel* option. It is changed so that we have Africa → Asia → Americas the custom migration matrix should be a 3x3 matrix (because we treat the model as a 3 population case); in addition *population-relabel* option is also changed to pool Brazil and Mexico and Puerto Rico.

## Model 7 to the west, to the east and slam

```
#inspect parmfile_twoways  
migrate-n parmfile_twoways -nomenu
```

We have Africa → Asia → Americas **and** Africa \rightarrows Americas the custom migration matrix should be a 3x3 matrix; the *population-relabel* option is used to pool Brazil and Mexico and Puerto Rico like in Model 6.

► Once you edited your copy compare with this

## Reporting

---

If we have time we will collect results from the student model runs. Otherwise we will compare the models Peter has run earlier for this tutorial.

You have run several models. Since we ran them not really long enough we want to see whether the best estimate for each model among students may give some intuition what is going on with the Zika virus. Each of the outfiles has a table of the marginal likelihoods, report the value for your output files (you can report these before you have run all of them) Report the number circled in **RED**. We will run an excel sheet to look at the model probabilities and marginal likelihoods reported by the class.

## Log-Probability of the data given the model (marginal likelihood)

Use this value for Bayes factor calculations:

$BF = \text{Exp}[\ln(\text{Prob}(D \mid \text{thisModel})) - \ln(\text{Prob}(D \mid \text{otherModel}))]$

or as  $LBF = 2 (\ln(\text{Prob}(D \mid \text{thisModel})) - \ln(\text{Prob}(D \mid \text{otherModel})))$

shows the support for thisModel]

Method	$\ln(\text{Prob}(D \mid \text{Model}))$	Notes
Thermodynamic integration	-28474.451244	(1a)
	-26023.532508	(1b)
Harmonic mean	-25640.444721	(2)

(1a, 1b and 2) are approximations to the marginal likelihood, make sure that the program run long enough!  
 (1a, 1b) and (2) should give similar results, in principle.  
 But (2) is overestimating the likelihood, it is presented for historical reasons and should not be used  
 (1a, 1b) needs heating with chains that span a temperature range of 1.0 to at least 100,000.  
 (1b) is using a Bezier-curve to get better approximations for runs with low number of heated chains

I have added a script that allows you to generate a table of the marginal likelihoods, the Bayes factors compared to the best model, and the model probabilities (probability weight). The script uses *grep* and a small *python* script that is doing the calculations. It will generate a table by extracting the marginal likelihoods from all **outfiles** in the directory, call the script like this

```
./modelprohtable
```

or you can get the same result with

```
grep "All" outfile* | sort -n -k 4,4 | bf.py
```