Instruction Manual

EEMS: a method to visualize patterns of non-stationary isolation by distance

Version 0.0.0.9000

Desislava Petkova <desislavka@gmail.com>

June 16, 2015

Contents

List of Figures

EEMS is a program to analyze and visualize spatial population structure from geo-referenced genetic samples. EEMS uses the concept of *effective migration* to model the relationship between genetics and geography, and it outputs an *e*stimated *e*ffective *m*igration *s*urface (hence, EEMS) – a visual representation of population structure that highlights potential regions of higher-than-average and lower-than-average historic gene flow.

1 Installation

This EEMS implementation uses Eigen for linear algebra computations and Boost for random number generation and the habitat geometry. The Eigen template library can be downloaded from `http://eigen.tuxfamily.org`, and the Boost libraries can be downloaded from `http://www.boost.org`. EEMS has been tested with Eigen 3.2.2 and Boost 1_57.

After downloading Eigen (which does not need installation) and installing Boost, update the variables `EIGEN_INC`, `BOOST_INC`, `BOOST_LIB` in the Makefile. The dynamic Boost libraries are linked to slightly differently on a Mac and a Linux machine, so run '`make darwin`' on a Mac or '`make linux`' on a Linux machine.

There are two versions of EEMS: `runeems_snps` for SNP data and `runeems_sats` for microsatellite data. The data input format and the EEMS model are somewhat different for SNPs and microsatellites, hence the two versions. The source code can be found in `runeems_snps/src` and `runeems_sats/src`, respectively. The directories `runeems_snps/data` and `runeems_sats/data` contain data, simulated with `ms` [Hudson, 2002], to illustrate the input file format.

2 Input data

There are three data input files as well as a parameter input file. The data input files should be in the same directory, and the description below assumes that `datapath` is the full path + the file name (but without the extension).

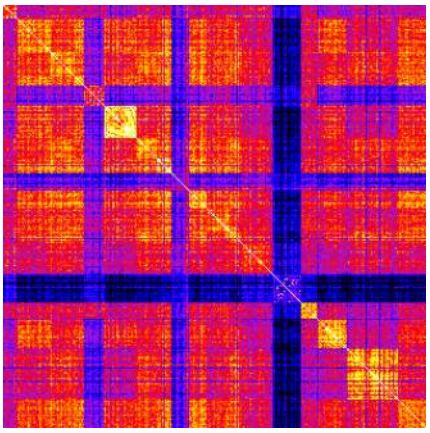
For SNP data on n individuals, the input files are:

- `datapath.diffs`: the matrix of average pairwise genetic dissimilarities. This can be computed with the small program `bed2diffs` if the dataset is already in plink binary format. See Figure 1**(a)**.
- `datapath.coord`: the sample coordinates, two coordinates per sample, one sample per line. See Figure 1**(b)**.
- `datapath.outer`: the habitat coordinates, as a sequence of vertices that form a closed polygon. See Figure 1**(c)**.

For microsatellite data on n individuals at L loci, the input files are:

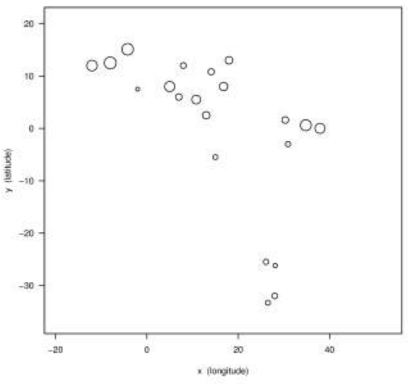
- `datapath.sites`: the matrix of allele copies (This matrix is $n \times L$ for L haploid markers and $n \times 2L$ for L diploid markers; missing alleles are specified by any negative number.)
- `datapath.coord`, `datapath.outer`: as described above for SNP data.

```
       0  0.656010  0.666522   ...
0.656010         0  0.661998   ...
0.666522  0.661998         0   ...
0.662323  0.652633  0.651595   ...
0.675922  0.668446  0.668843   ...
     ...       ...       ...   ...
```



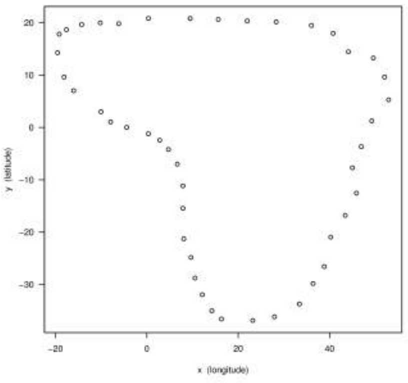
(a) The observed genetic dissimilarity matrix: the top left corner of `datapath.diffs` on the left and the entire matrix as a heatmap on the right.

```
30.9  -3.0
30.9  -3.0
30.9  -3.0
30.9  -3.0
30.9  -3.0
...
```



(b) The sample coordinates: the top 5 lines in `datapath.coord` on the left and the complete sampling scheme on the right. A bigger circle means that more samples are taken from that location.

```
-18.1   9.6
-16.0   7.0
-10.0   3.0
...
-19.2  17.8
-19.5  14.3
-18.1   9.6
```



(c) The habitat polygon: the top and bottom lines in `datapath.outer` on the left and the entire habitat outline on the right.

Figure 1 EEMS requires three data input files. **(a)** `datapath.diffs` is the observed genetic dissimilarity matrix. There is a lot of population structure in this example and the rows and columns can be ordered to emphasize the pattern further. This ordering, however, is irrelevant to EEMS. **(b)** The sampling locations in `datapath.coord` should be given in the same order as the rows and columns of the dissimilarity matrix. **(c)** The habitat vertices in `datapath.outer` should be listed *counterclockwise* and the first vertex should also be the last vertex, so that the outline is a *closed ring*. Otherwise, EEMS attempts to "correct" the polygon and prints a warning message.

3 Running the EEMS program

This example shows how to run EEMS on a small simulated dataset (one of the "barrier to migration" datasets used for Figure 2 in the EEMS paper.)

First, copy/create the three input parameter files. All input parameters are the same, except for the output directory `mcmcpath`. Running `runeems_snps` three times will sample – independently – three MCMC chains from the same target distribution: the posterior distribution of the EEMS model parameters.

params-chain1.ini:

```
datapath = ../data/barrier-schemeX-nIndiv300-nSites3000
mcmcpath = ../data/barrier-schemeX-nIndiv300-nSites3000-EEMS-nDemes200-chain1
nIndiv = 300
nSites = 3000
nDemes = 200
diploid = false
numMCMCIter = 2000000
numBurnIter = 1000000
numThinIter = 9999
```

params-chain2.ini:

```
datapath = ../data/barrier-schemeX-nIndiv300-nSites3000
mcmcpath = ../data/barrier-schemeX-nIndiv300-nSites3000-EEMS-nDemes200-chain2
nIndiv = 300
nSites = 3000
nDemes = 200
diploid = false
numMCMCIter = 2000000
numBurnIter = 1000000
numThinIter = 9999
```

params-chain3.ini:

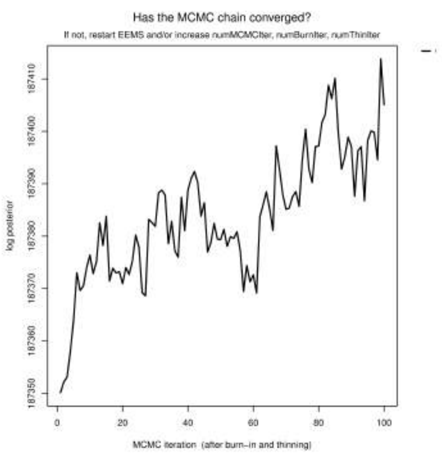
```
datapath = ../data/barrier-schemeX-nIndiv300-nSites3000
mcmcpath = ../data/barrier-schemeX-nIndiv300-nSites3000-EEMS-nDemes200-chain3
nIndiv = 300
nSites = 3000
nDemes = 200
diploid = false
numMCMCIter = 2000000
numBurnIter = 1000000
numThinIter = 9999
```

Then run EEMS (on the command line) with three different random seeds. The argument `--seed` is optional; it can be specified in the parameter file as well, and if not specified, the seed is randomly assigned.

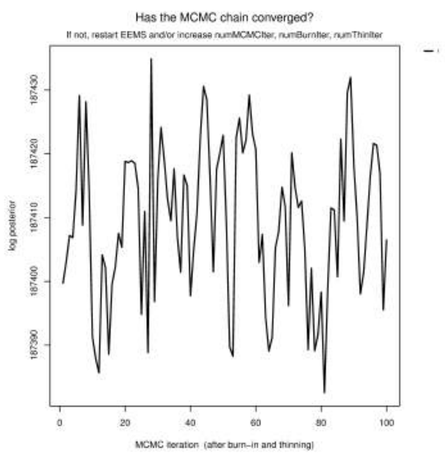
```
./runeems_snps --params params-chain1.ini --seed 123
./runeems_snps --params params-chain2.ini --seed 456
./runeems_snps --params params-chain3.ini --seed 789
```

3.1 How to restart EEMS

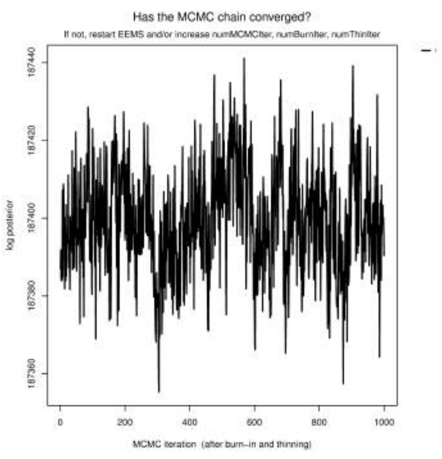
Suppose that `runeems_snps` (or `runeems_sats`, if the data is microsatellites) has finished running but the MCMC chain has not converged, as indicated by the posterior trace plot. [Section 4 describes how to generate this plot as well as two other diagnostic figures.]



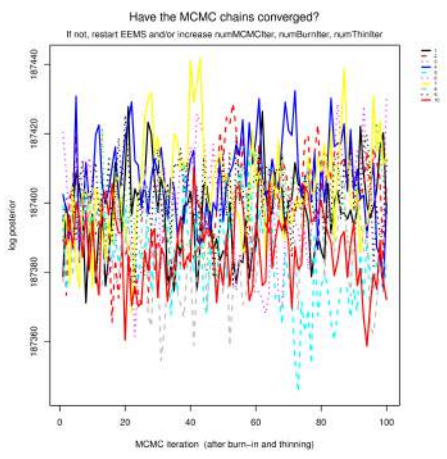
(a)



(b)



(c)



(d)

Figure 2 The `eems.plots` function in the `rEEMSplots` package generates a posterior trace plot. **(a)** This MCMC chain obviously has not converged. **(b)** There is no indication that the MCMC chain has not converged. This is not quite the same as there is evidence that the MCMC chain has converged. **(c)** To be confident that EEMS has converged, we can run the MCMC sampler for more iterations. **(d)** Alternatively, to be confident that EEMS has converged, we can run the MCMC sampler several times, each time starting from a different randomly initialized parameter state.

Rather than initializing the parameter state randomly, the user can restart EEMS from the final parameter state in a previous run, by providing the path to an existing EEMS output directory `prevpath`. The required arguments remain the same. The `datapath`, `nIndiv`, `nSites` are fixed as they describe the dataset, but new values can be chosen for all other arguments, e.g., no burn-in (`numBurnIter = 0`) or a denser grid (with more `nDemes`).

If `mcmcpath` is the same as `prevpath`, then the previous results will be overwritten.

```
datapath = ../data/barrier-schemeX-nIndiv300-nSites3000
prevpath = ../data/barrier-schemeX-nIndiv300-nSites3000-EEMS-nDemes200-chain1
mcmcpath = ../data/barrier-schemeX-nIndiv300-nSites3000-EEMS-nDemes200-chain1
nIndiv = 300
nSites = 3000
nDemes = 200
diploid = false
numMCMCIter = 1000000
numBurnIter = 0
numThinIter = 9999
```

If `mcmcpath` is different from `prevpath`, then the previous results will not be overwritten.

```
datapath = ../data/barrier-schemeX-nIndiv300-nSites3000
prevpath = ../data/barrier-schemeX-nIndiv300-nSites3000-EEMS-nDemes200-chain1
mcmcpath = ../data/barrier-schemeX-nIndiv300-nSites3000-EEMS-nDemes200-chain1.1
nIndiv = 300
nSites = 3000
nDemes = 200
diploid = false
numMCMCIter = 1000000
numBurnIter = 0
numThinIter = 9999
```

After running EEMS (either `runeems_snps` for SNP data or `runeems_sats` for microsatellite data), the next step to visualize the results.

4 Plotting the EEMS results

First install the R package `rEEMSplots` which provides the function `eems.plots`. It is not on CRAN, so install it from source instead.

```r
## Check that the current directory contains the rEEMSplots source directory (from GitHub)
if (file.exists("./rEEMSplots")) {
  install.packages("rEEMSplots", repos=NULL, type="source")
} else {
  stop("Move to the directory that contains the rEEMSplots source to install the package.")
}
```

`rEEMSplots` provides a single function, with a variety of options. I will use an example to demonstrate those options.

```r
## Use the provided example or supply the path to your own EEMS run.
eems.results.to.plot = paste(path.package("rEEMSplots"),"/extdata/EEMS-example",sep="")
name.figures.to.save = "EEMS-example-rEEMSplots"

if (!file.exists(eems.results.to.plot)) {
  stop("Check that the rEEMSplots package is installed without errors.")
}
```

4.1 Required arguments

1. `mcmcpath` is a list of EEMS output directories, for the same dataset. It can be a single directory but it is better to run EEMS several times, randomly initializing the MCMC chain each time. [In other words, it is a good idea to simulate several realizations of the Markov chain, each realization starting with a different value of the EEMS parameters.] Warning: There is minimal checking that all directories in the list are for the same dataset.

2. `plotpath` is the full path + the file name for the graphics to be generated. There are five output figures: `plotpath-mrates01` (effective migration surface), `plotpath-qrates01` (effective diversity surface), `plotpath-rdist01` (between-demes component of genetic dissimilarity), `plotpath-rdist02` (within-demes component of genetic dissimilarity), `plotpath-pilogl01` (posterior probability trace). The three latter figures are helpful in checking that the MCMC sampler has converged (the trace plot) and that the EEMS model fits the data well (the scatter plots of genetic dissimilarities).

3. `longlat` specifies whether coordinates are given as pairs (longitude, latitude) or (latitude, longitude). In the former case, `longlat = TRUE`, and in the latter case, `longlat = FALSE`. [More generally, the coordinates are ordered pairs of numbers. Almost all plotting options are valid for general coordinates, except for the options to specify the projection and to add the geographic map.]

I will use the PoBI data to demonstrate how to visualize EEMS results. PoBI stands for "People of the British Isles" – see [Winney et al., 2012] for information on how the data was collected and an initial analysis, as well as [Leslie et al., 2015] for extensive population structure analysis using fineSTRUCTURE [Lawson et al., 2012].

Figure 4 shows the figures produced by `eems.plots` when only the three required arguments are specified.

```r
## Produce the five EEMS figures, with default values for all optional parameters.
eems.plots(mcmcpath = eems.results.to.plot,
           plotpath = paste(name.figures.to.save,"-default",sep=""),
           longlat = TRUE)
```

Passing `longlat = FALSE` instead of `longlat = TRUE` has the effect of flipping the x-axis and the y-axis – compare Figures 4**(a)** and 4**(b)** to Figures 5**(a)** and 5**(b)**. It is particularly important to specify this argument correctly if adding a geographic map to the migration and diversity contour plots. [That's why `longlat` is a required, not an optional argument.]

```r
## Flip the x and y axis, i.e., assume that the x coordinate is the latitude and the y coordinate is the longitude.
eems.plots(mcmcpath = eems.results.to.plot,
           plotpath = paste(name.figures.to.save,"-axes-flipped",sep=""),
           longlat = FALSE)
```

4.2 Optional arguments about the graphics format

`eems.plots` generates output graphics as either PNGs (the default) or PDFs.

- `plot.height/plot.width`: dimensions of the display region, in inches.
- `out.png`: TRUE or FALSE, specifies whether to generate PNGs or PDFs.
- `res`: resolution, in dots per inch, only relevant if the output is PNG graphics. The default is 600.

```r
## Generate PNG figures with height 9 inches, width 8 inches and resolution 600 dots per inch.
eems.plots(mcmcpath = eems.results.to.plot,
           plotpath = paste(name.figures.to.save,"-output-PNGs",sep=""),
           longlat = TRUE,
           plot.height = 8,
           plot.width = 7,
           res = 600,
           out.png = TRUE)

## Generate PDF figures with height 9 inches and width 8 inches. The resolution option, res, is ignored.
eems.plots(mcmcpath = eems.results.to.plot,
           plotpath = paste(name.figures.to.save,"-output-PDFs",sep=""),
           longlat = TRUE,
           plot.height = 8,
           plot.width = 7,
           res = 600,
           out.png = FALSE)
```

4.3 Optional arguments about the population grid

- `add.grid`: TRUE or FALSE, specifies whether to add the population grid or not.

  - col.grid: the color of the grid (default is gray80)
  - lwd.grid: the line width (default is 1)

- add.outline: TRUE or FALSE, specifies whether to add the habitat outline or not.
  - col.outline: the color of the outline (default is white)
  - lwd.outline: the line width (default is 2)

- add.demes: TRUE or FALSE, specifies whether to add the observed demes or not.
  - col.demes: the color of the demes (default is black)
  - pch.demes: the symbol/character used for plotting the demes (default is 19)
  - min.cex.demes: the minimum size of the deme symbol/character (default is 1)
  - max.cex.demes: the maximum size of the deme symbol/character (default is 3)

The unusual options are min.cex.demes and max.cex.demes. If max.cex.demes > min.cex.demes, then demes with more samples also have bigger size: the deme with the fewest samples has size min.cex.demes and the deme with the most samples has size max.cex.demes. So if the sampling is uneven, then max.cex.demes > min.cex.demes underlines this fact. See Figure 6 for an example.

```r
## Choose somewhat impractical colors and shapes for the outline, the grid and the demes.
eems.plots(mcmcpath = eems.results.to.plot,
           plotpath = paste(name.figures.to.save,"-demes-and-edges",sep=""),
           longlat = TRUE,
           add.grid = TRUE,
           col.grid = "gray90",
           lwd.grid = 2,
           add.outline = TRUE,
           col.outline = "blue",
           lwd.outline = 5,
           add.demes = TRUE,
           col.demes = "red",
           pch.demes = 5,
           min.cex.demes = 0.5,
           max.cex.demes = 1.5)
```

4.4 Optional arguments about the cartographic projection

If we know the projection of the input coordinates and the R package rgdal is installed, then we can plot the effective migration and diversity surfaces in another projection. See Figure 7 for an example.

- projection.in: input projection, as a valid PROJ.4 string.
- projection.out: output projection, as a valid PROJ.4 string.

```r
library(rgdal)

## Produce contour plots in the Mercator projection (used by Google Maps)
eems.plots(mcmcpath = eems.results.to.plot,
           plotpath = paste(name.figures.to.save,"-merc-projection",sep=""),
           longlat = TRUE,
           projection.in = "+proj=longlat +datum=WGS84",
           projection.out = "+proj=merc +datum=WGS84")
```

4.5 Optional arguments about the geographic map

If we know the projection of the input coordinates and the R packages `rgdal`, `rworldmap`, `rworldxtra` are installed, then we can add a geographic map to the two contour plots. See Figure 8 for an example.

- `add.map`: TRUE or FALSE, specifies whether to add a geographic map or not.
- `col.map`: the color of the map (default is `gray60`)
- `lwd.map`: the line width (default is 2)

```r
library(rworldmap)
library(rworldxtra)

## Add a high-resolution geographic map
eems.plots(mcmcpath = eems.results.to.plot,
           plotpath = paste(name.figures.to.save,"-geographic-map",sep=""),
           longlat = TRUE,
           projection.in = "+proj=longlat +datum=WGS84",
           projection.out = "+proj=merc +datum=WGS84",
           add.map = TRUE,
           col.map = "black",
           lwd.map = 5)
```

4.6 Optional arguments about the color scheme

Finally, we can also specify the color palette as a vector of colors, `eems.plot`, ordered from low to high. See Figure 9 for an example.

```r
library(RColorBrewer)

## Use a divergent Red to Blue color scheme from 'RColorBrewer' instead of the default DarkOrange to Blue color scheme.
eems.plots(mcmcpath = eems.results.to.plot,
           plotpath = paste(name.figures.to.save,"-new-eems-colors",sep=""),
           longlat = TRUE,
           projection.in = "+proj=longlat +datum=WGS84",
           projection.out = "+proj=merc +datum=WGS84",
           eems.colors = brewer.pal(11,"RdBu"))
```

4.7 A test function to generate Voronoi diagrams

Given one EEMS output directory, the function `eems.voronoi`, also available in the `rEEMSplots` package, produces Voronoi diagrams drawn from the posterior distribution of the rate parameters: there is one series of tessellations for the effective migration rates and another series for the effective diversity rates. Both series contain one figure for each saved MCMC iteration, after burn-in and thinning. Warning: `eems.voronoi` potentially generates a large number of figures. See Figure 10 for a small example.

```r
library(deldir)

## Plot a series of Voronoi diagrams for the EEMS model parameters:
## the effective migration rates (m) and the effective diversity rates (q).
eems.voronoi(mcmcpath = eems.results.to.plot,
             plotpath = paste(name.figures.to.save,"-default",sep=""),
             longlat = TRUE,
             plot.height = 7,
             plot.width = 5)
```

5 Input parameters

There are a number of program parameters that can be set by the user. First, there are several parameters that have to be specified (they have no default values):

- `datapath`: path to the input data. For SNP data, EEMS expects three files: a matrix of average pairwise differences, `datapath.diffs`; a list of sample coordinates, `datapath.coord`; a list of habitat boundary points, `datapath.outer`. For microsatellite data, EEMS also expects three files: a matrix of allele copies, `datapath.sites`; a list of sample coordinates, `datapath.coord`; a list of habitat boundary points, `datapath.outer`.

- `mcmcpath`: path to the output directory. EEMS creates this directory and saves all results there. There is an accompanying R package which parses the output files and generates several figures to visualize the EEMS results. See Section 4.

- `nIndiv` and `nSites`: number of samples and number of markers. For SNP data, the `diffs` matrix is `nIndiv`×`nIndiv` but the input data files do not otherwise contain information about `nSites`. For microsatellite data, the `sites` matrix is `nIndiv`×`nSites` if the species is haploid, and `nIndiv`×`2nSites` is the species is diploid.

- `numMCMCIter`, `numBurnIter`, `numThinIter`: number of MCMC iterations, number of burn-in iterations to discard at the start, and number of iterations to thin between two writing steps.

- `nDemes`: the (approximate) number of demes in the population graph. If the habitat is irregular, the actual number of demes in the grid might not be exactly equal to the specified number of demes. See Figure 3. Alternatively, instead of `nDemes`, specify `gridpath` where the file `gridpath.demes` is a list of demes, the file `gridpath.edges` is a list of edges and `gridpath.ipmap` is a mapping which assigns samples to demes. See Section 5.1.

EEMS assumes that the samples come from a diploid species [`diploid = true`, which is the default] or a haploid species [`diploid = false`]. The difference is only in how the matrix of expected genetic dissimilarity is scaled. See Section S1.2 in the supplement of the EEMS paper.

Further, there are several extra parameters that can be specified optionally (they have default values but modifying those values could improve convergence).

- Variances for the proposal distributions of migration parameters:
  - `mEffctProposalS2`: proposal variance for the migration cell effects, $e_1, \ldots, e_{C_m}$. Defaults to 0.1.
  - `mSeedsProposalS2`: proposal variance for the migration cell locations, $s_1, \ldots, s_{C_m}$. Defaults to 0.01.
  - `mrateMuProposalS2`: proposal variance for the overall migration rate μ (on the $\log_{10}$ scale). Defaults to 0.01.

- Variances for the proposal distributions of diversity parameters:

  - `qEffctProposalS2`: proposal variance for the diversity cell effects, $f_1, \ldots, f_{C_q}$. Defaults to 0.001.
  - `qSeedsProposalS2`: proposal variance for the diversity cell locations, $t_1, \ldots, t_{C_q}$. Defaults to 0.1.

- Variance for the proposal distribution on the degrees of freedom:
  - `dfProposalS2`: only relevant for SNP data and thus only used in `runeems_snps`. The default value is the square root of the number of markers `nSites`. For microsatellite data, the degrees of freedom are equal to the number of loci.

- Hyperparameters:
  - `mrateShape,mrateScale`: (inverse gamma) hyperparameters for the variance ω_m^2 of the migration effects. Default to 0.001 and 1, respectively.
  - `qrateShape,qrateScale`: (inverse gamma) hyperparameters for the variance ω_q^2 of the diversity effects. Default to 0.001 and 1, respectively.
  - `s2locShape,s2locShape`: (inverse gamma) hyperparameters for the scale parameters $\sigma_1^2, \ldots, \sigma_p^2$. Default to 0.001 and 1, respectively.
  - `negBiSize`: number of failures for the Negative-Binomial prior on the number of Voronoi tiles. Defaults to 10.
  - `negBiProb`: success probability for the Negative-Binomial prior on the number of Voronoi tiles. Defaults to 0.67.

For all datasets analyzed in the EEMS paper, we used the default hyperparameter values and we tuned the proposal variances. Choosing values for the proposal variances is something of a dark art – the goal is to choose the parameters so that proposals are accepted about 20% – 30% of the time. In practice, it seems to be sufficient to choose the variances so that proposals are not accepted too rarely (less than 10% of the time) or too often (more than 40% of the time).

Suppose that we run EEMS (say, with the default values for all proposal distributions) and at the end EEMS reports the following information about the frequency of accepting proposals of different types.

```
(2050/2593) = 79.06% for type "qTileRate"          ## change the rate of one tile in the diversity tessellation
(1414/2478) = 57.06% for type "qTileMove"          ## move the seed of one tile in the diversity tessellation
(1047/2494) = 41.98% for type "qBirthDeath"        ## add/remove one tile in the diversity tessellation
(20625/47530) = 43.39% for type "mTileRate"        ## change the rate of one tile in the migration tessellation
(15486/50098) = 30.91% for type "mMeanRate"        ## change the overall log10 migration rate
(22352/47532) = 47.03% for type "mTileMove"        ## move the seed of one tile in the migration tessellation
(26305/47275) = 55.64% for type "mBirthDeath"      ## add/remove one tile in the migration tessellation
```

It seems that the `qTileRate` and `qTileMove` are accepted too often. So it might be a good idea to increase `qEffctProposalS2` and `qSeedsProposalS2`. [If the proposal variance is higher, then bigger changes/moves will be proposed and hence the updates will be accepted less often.]

Finally, the parameter `qVoronoiPr` specifies how often to propose updating the diversity Voronoi tessellation. Its default value is 0.05, which means that about 5% of the time the proposals are about the diversity tessellation and about 95% of the time the proposals are about the migration tessellation.

5.1 Population grid

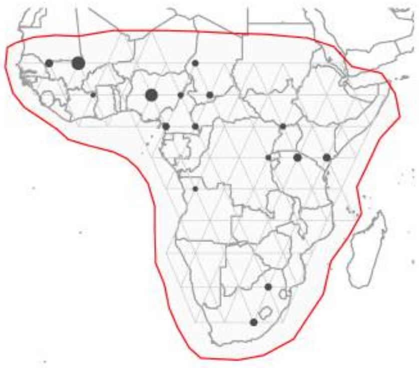
The population graph is a triangular grid contained entirely inside the habitat. The user specifies its density as a number of demes, `nDemes`, and EEMS constructs a grid with about the same number of demes which fills the habitat. This procedure is not necessarily optimal (i.e., it might not cover as much area as possible near the boundaries) and it might be helpful to iterate between refining the habitat specification and the density argument `nDemes`, to choose a satisfactory grid. [Or even better, since the assignment of samples to demes changes with the grid, run EEMS with slightly different grids to investigates how/whether this changes the results.] See Figure 3 for an illustration. Furthermore, there exists a convenient online tool to draw polygons with Google Maps API, available at `http://www.birdtheme.org/useful/v3tool.html`, that might be useful for outlining an irregularly shaped habitat, in longitude and latitude coordinates.

Alternatively, EEMS can read in the population graph. In this case, there are three necessary files:

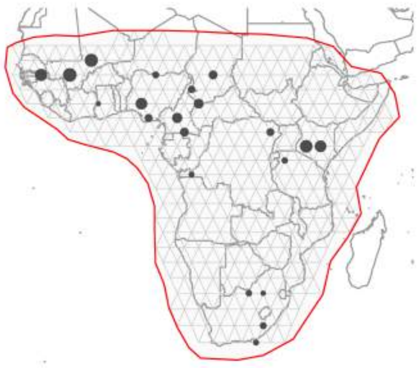
- `gridpath.demes`: list of demes, two coordinates per deme, one deme per row.
- `gridpath.edges`: list of connected pairs of demes, two indices per edge, one edge per row. Demes are indexed from 1 to d (where d is the number of demes in the grid), in the order implied by `gridpath.demes`, i.e., the deme with index 1 has the coordinates on the 1st row in `gridpath.demes`.
- `gridpath.ipmap`: list of deme indices which indicate the deme each sample is assigned to. There should be n rows (where n is the number of samples), with one row for each sample in `datapath.coord`, in the same order.

The sample and habitat coordinates can be specified either latitude first or longitude first as long as the files `datapath.coord` and `datapath.outer` use the same convention. However, the ordering of the coordinates must be specified when plotting the results; see Section 4.

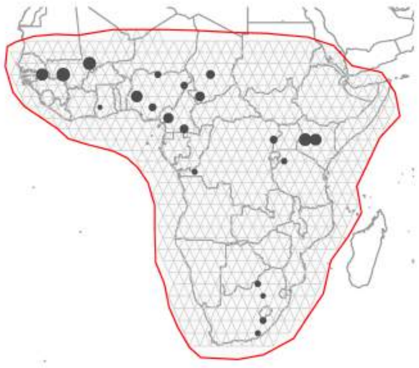
Finally, the vertices of the population graph (the demes) fall inside the habitat by construction. However, EEMS does not check that the sampling locations fall inside the habitat – instead, each sample is assigned to the closest deme. Therefore, the user should specify a habitat that is sufficiently large to cover all sampling locations.



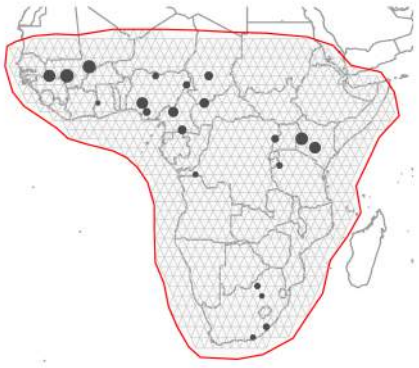
(a) `nDemes = 100` (There are 75 vertices.)



(b) `nDemes = 400` (There are 358 vertices.)



(c) `nDemes = 700` (There are 651 vertices.)



(d) `nDemes = 1000` (There are 927 vertices.)

Figure 3 Given the grid density parameter, `nDemes`, EEMS automatically generates a triangular grid, which is contained entirely inside the habitat. Then EEMS assigns each sample to its closest deme. Here the habitat is outlined in red. [The outline was created, as a "*polyline*", with a convenient online tool from Google, available at `http://www.birdtheme.org/useful/v3tool.html`.] A higher value for `nDemes` produces a denser grid, and a denser grid means a more flexible EEMS model because, with more vertices and edges, there are more rate parameters in the model. However, the computational cost of running EEMS increases as well.

6 Observed vs fitted dissimilarities

EEMS aims to model the genetic differentiation between individuals in space. To achieve this, EEMS constructs a triangular grid, assigns each sample to its closest vertex (deme) and estimates a migration rate for every edge and a diversity rate for every vertex in the grid.

To be specific, consider two different demes α and β. Suppose that individual i is assigned to deme α, which we denote by $\delta(i) = \alpha$. Similarly, suppose that $\delta(i^*) = \alpha$ but i and i^* are distinct individuals, that $\delta(j) = \beta$ and $\delta(j^*) = \beta$ but j and j^* are distinct individuals. Of course, i and j are distinct because they come from different demes.

In EEMS, the observed genetic dissimilarity is the average squared genetic difference, across p markers:

$$D_{ij} = \frac{1}{p}\sum_{k=1}^{p}\left(Z_{ik} - Z_{jk}\right)^2, \tag{1}$$

where Z_{ik}, Z_{ik} are the genotypes of individuals i, j at marker k. There are various other ways to define genetic (dis)similarity – for example, fineSTRUCTURE uses a statistic called "coancestry" which is a measure of genetic similarity. Proportion of the genome shared identical by descent (IBD fraction) is another statistic informative about recent population structure.

EEMS assumes that individuals in the same deme are exchangeable. Since $\delta(i) = \delta(i^*) = \alpha$ and $\delta(j) = \delta(j^*) = \beta$, exchangeability implies that:

$$\mathrm{E}\{D_{ij}\} = \mathrm{E}\{D_{ij^*}\} = \mathrm{E}\{D_{i^*j}\} = \mathrm{E}\{D_{i^*j^*}\} = \Delta_{\alpha\beta}, \qquad \mathrm{E}\{D_{ii^*}\} = \Delta_{\alpha\alpha}, \qquad \mathrm{E}\{D_{jj^*}\} = \Delta_{\beta\beta}. \tag{2}$$

That is, the expected genetic dissimilarity between two distinct individuals depends only on their locations. In matrix notation, the observed data is the matrix of average squared genetic differences, $D = (D_{ij})$, which is modeled by a *block* matrix of expected genetic dissimilarities, $\Delta = (\Delta_{\delta(i)\delta(j)}) = (\Delta_{\alpha\beta})$. Both matrices have a main diagonal of 0s because self-dissimilarity is 0.

EEMS is a spatially explicit model and it is consistent with the following idea: We can expect that $\langle i, j\rangle$ are more dissimilar than either $\langle i, i^*\rangle$ or $\langle j, j^*\rangle$ because i and j come from different locations. However, in general, we can't expect $\langle i, i^*\rangle$ to be as dissimilar as $\langle j, j^*\rangle$ – there might be some differences in local genetic diversity. And to model the genetic dissimilarity due to migration in space we need to take into account any local variation in genetic diversity. For this reason, EEMS decomposes the genetic dissimilarity into two components, between demes and within demes:

$$\Delta_{\alpha\beta} = \underbrace{\Delta_{\alpha\beta} - \left(\Delta_{\alpha\alpha} + \Delta_{\beta\beta}\right)/2}_{\text{between demes}} + \underbrace{\left(\Delta_{\alpha\alpha} + \Delta_{\beta\beta}\right)/2}_{\text{within demes}}. \tag{3}$$

The within-demes component, w, characterizes the expected genetic dissimilarity between two distinct individuals from the same deme and is a function of the effective diversity rates:

$$w_\alpha = \Delta_{\alpha\alpha} = g(q_\alpha). \tag{4}$$

Here q is a vector of effective diversity rates and q_α is the element that corresponds to deme α. The function g is the identity.

The between-demes component, B, characterizes the expected genetic dissimilarity between two individuals from distinct demes, *after* correcting for the local differences in genetic diversity, and is a function of the effective migration rates:

$$B_{\alpha\beta} = \Delta_{\alpha\beta} - \left(\Delta_{\alpha\alpha} + \Delta_{\beta\beta}\right)/2 = f(m)_{\alpha\beta}. \tag{5}$$

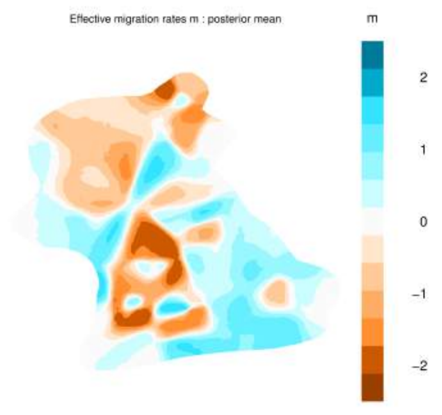
Here m is a sparse matrix that represents an undirected, connected, weighted grid, with weights equal to the effective migration rates between adjacent demes. The function f return the effective resistance distances between vertices in the grid, as a dense matrix, and $f(m)_{\alpha\beta}$ is the element that corresponds to the pair of demes α and β.

Therefore, EEMS models expected genetic dissimilarity as:

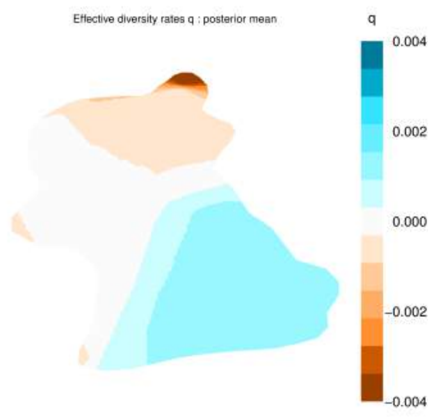
$$\begin{aligned} \Delta_{\alpha\beta} &= \underbrace{\Delta_{\alpha\beta} - \left(\Delta_{\alpha\alpha} + \Delta_{\beta\beta}\right)/2} + \underbrace{\left(\Delta_{\alpha\alpha} + \Delta_{\beta\beta}\right)/2} && (6) \\ &= \qquad B_{\alpha\beta} \qquad + \left(w_\alpha + w_\beta\right)/2. && (7) \end{aligned}$$

The between-demes component of the genetic dissimilarity is plotted in `plotpath-rdist01`, as a scatterplot of observed vs fitted values. The within-demes component of the genetic dissimilarity is plotted in `plotpath-rdist02`. See Figure 11 for an example. If the EEMS model fits the data well (after the MCMC chain has converged), we expect to see a strong linear relationship between the observed and fitted values in both scatterplots. Therefore, these scatterplots are useful for model-checking diagnostics.

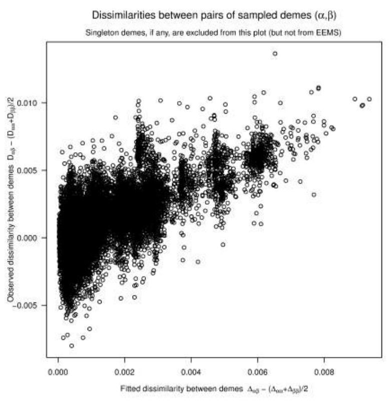
However, once we are reasonably convinced that the EEMS model is appropriate for the data being analyzed, the parameters of the model are of greater interest than the expected dissimilarities, i.e., m and q are more interesting than the deterministic functions $f(m)$ and $g(q)$. The rate parameters are visualized in the colored contour plots: the effective migration rates m are plotted in `plotpath-mrates01`, the effective diversity rates q are plotted in `plotpath-qrates01`, in both cases on the $\log_{10}$ scale and after some normalization.



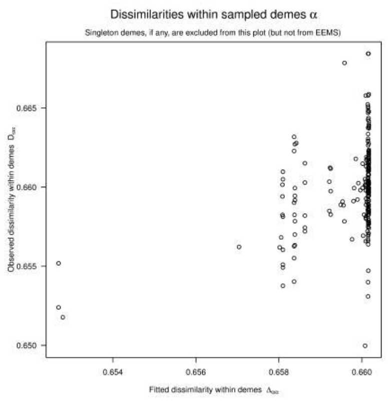
(a)



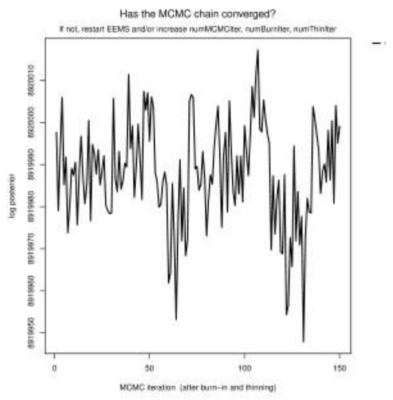
(b)



(c)

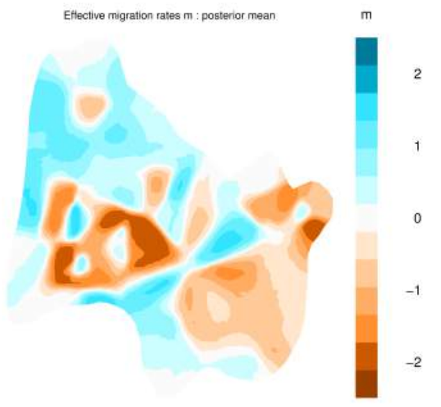


(d)

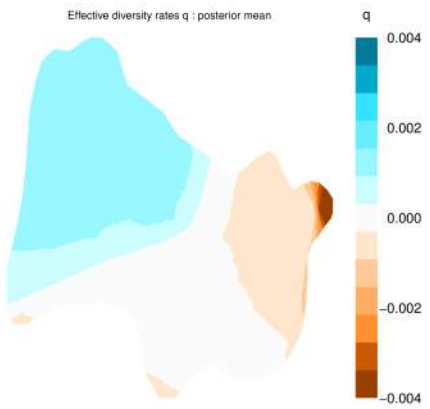


(e)

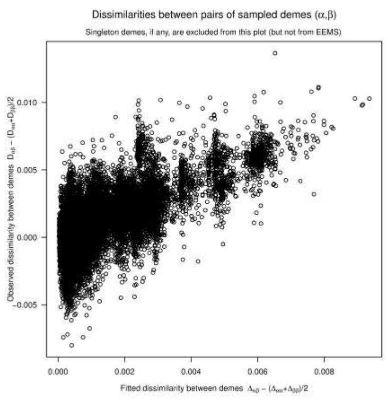
Figure 4 The `eems.plots` function in the `rEEMSplots` package produces five figures. **(a)** Estimated effective migration surface. **(b)** Estimated effective diversity surface. **(c)** Observed vs fitted dissimilarities: between-demes component. See Section 6 for more details. **(d)** Observed vs fitted dissimilarities: within-demes component. See Section 6 for more details. **(e)** Posterior probability trace.



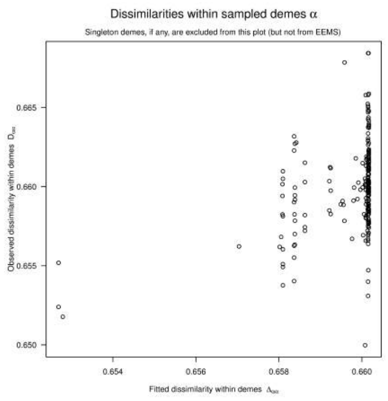
(a)



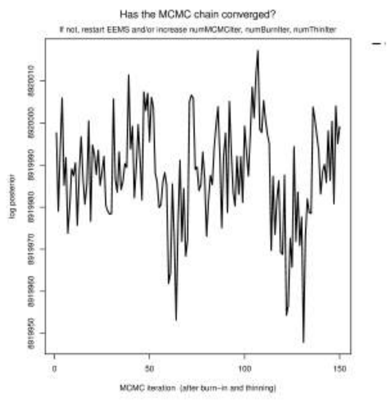
(b)



(c)

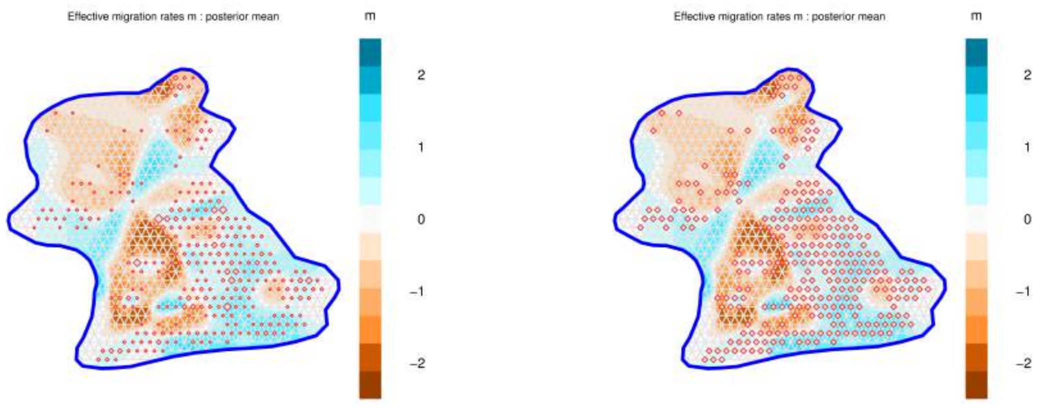


(d)



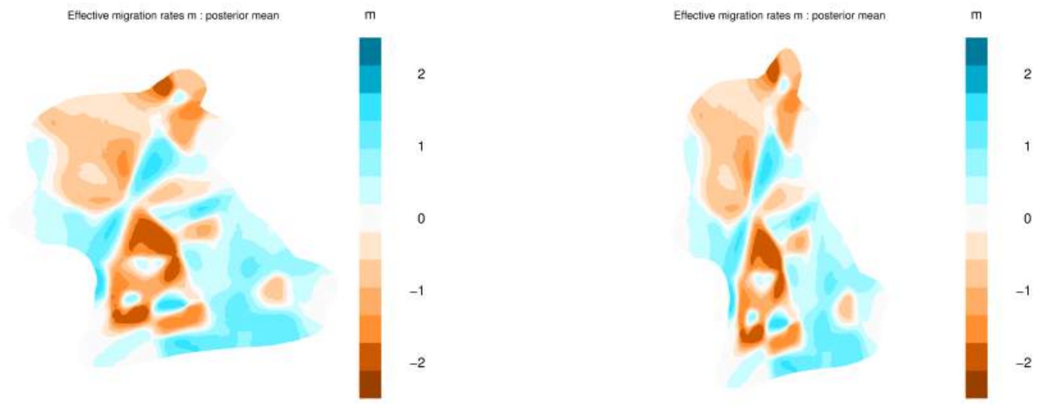
(e)

Figure 5 In EEMS points in the habitat (e.g., the sampling locations, the demes, the tile seeds) are represented as ordered pairs (x, y). The x and y axes can be flipped with the `longlat` argument to `eems.plots`: use `longlat = TRUE` for the original ordering of the two coordinates and `longlat = FALSE` to flip them. This changes the migration and the diversity figures, **(a)** and **(b)**, but not the three diagnostic figures, **(c)**, **(d)** and **(e)**. In fact, all optional arguments to the plotting function modify only the migration and diversity contour plots.


Effective migration rates m : posterior mean
m
2
1
0
−1
−2
Effective migration rates m : posterior mean
m
2
1
0
−1
−2

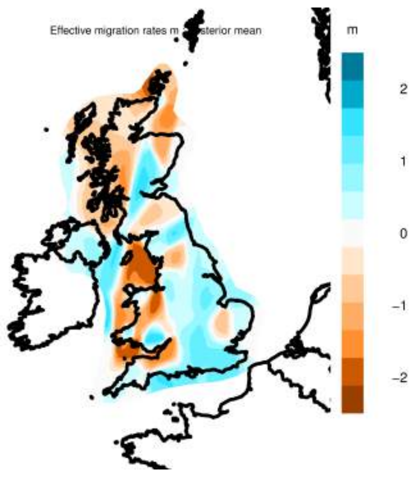
(a) (b)

Figure 6 Use the options `add.grid`, `add.demes`, `add.outline` to add the population grid, the sampled demes and/or the habitat outline. The colors, line width and the symbol/character can all be specified. Most importantly, the arguments `min.cex.demes` and `max.cex.demes` can be used to underline the sampling scheme, so that a deme with more samples is indicated with a larger symbol/character. **(a)** `min.cex.demes = 0.5`, `max.cex.demes = 1.5` **(b)** `min.cex.demes = max.cex.demes = 1`

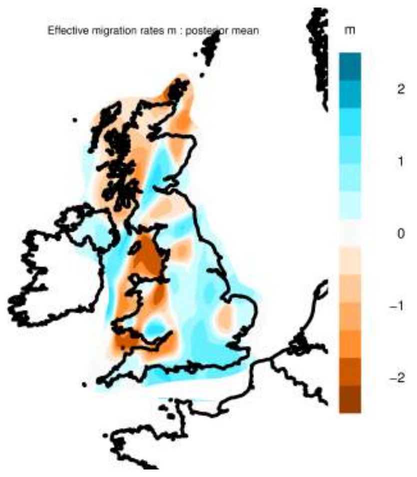

Effective migration rates m : posterior mean
m
2
1
0
−1
−2
Effective migration rates m : posterior mean
m
2
1
0
−1
−2

(a) (b)

Figure 7 Use the arguments `projection.in` and `projection.out` to specify the input and output projection, as valid PROJ.4 strings. This requires the `rgdal` package. **(a)** The effective migration surface with longitude and latitude untransformed: `projection.out = "+proj=longlat +datum=WGS84"`. **(b)** The same effective migration surface in the Mercator projection: `projection.out = "+proj=merc +datum=WGS84"`.

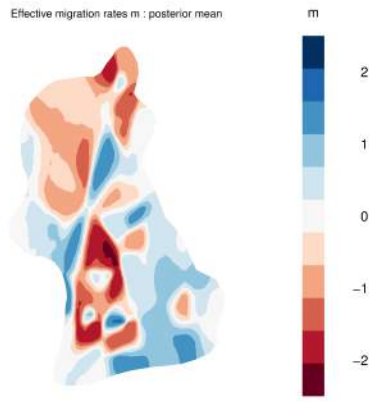


(a)

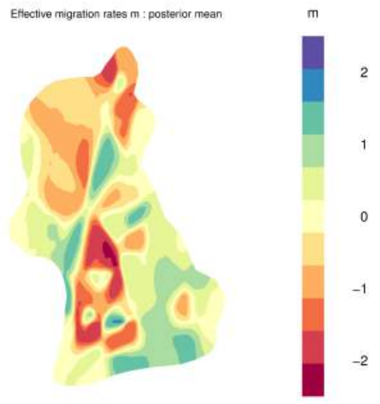


(b)

Figure 8 If we know the input projection (e.g., if we know that the input coordinates are given in longitude/latitude), then we can add a geographic map to the migration and diversity contour plots. This requires the `rworldmap` package (for the `getMap` function which provides access a world map derived from Natural Earth data) and the `rworldxtra` package (for the high-resolution version of the map). **(a)** In the Mercator (cylindrical) projection. **(b)** In the Lambert-93 (conic) projection, with PROJ.4 string "`+proj=lcc +lat_1=49 +lat_2=44 +lat_0=46.5 +lon_0=3 +x_0=700000 +y_0=6600000 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m +no_defs`".



(a)



(b)

Figure 9 By default, the plotting function `eems.plots` uses a DarkOrange to Blue color scheme, adapted from the `dichromat` package. [This scheme takes the dark oranges from the `BluetoDarkOrange.12` theme and the blues the `BrowntoBlue.12` theme.] To specify a different color scheme, we can supply a vector of colors with the `eems.colors` argument. These can be any colors but only a divergent scheme makes sense for a contour plot. **(a)** The effective migration surface in the `RdBu` palette from the `RColorBrewer` package. **(b)** The effective migration surface in the `Spectral` palette from the `RColorBrewer` package.

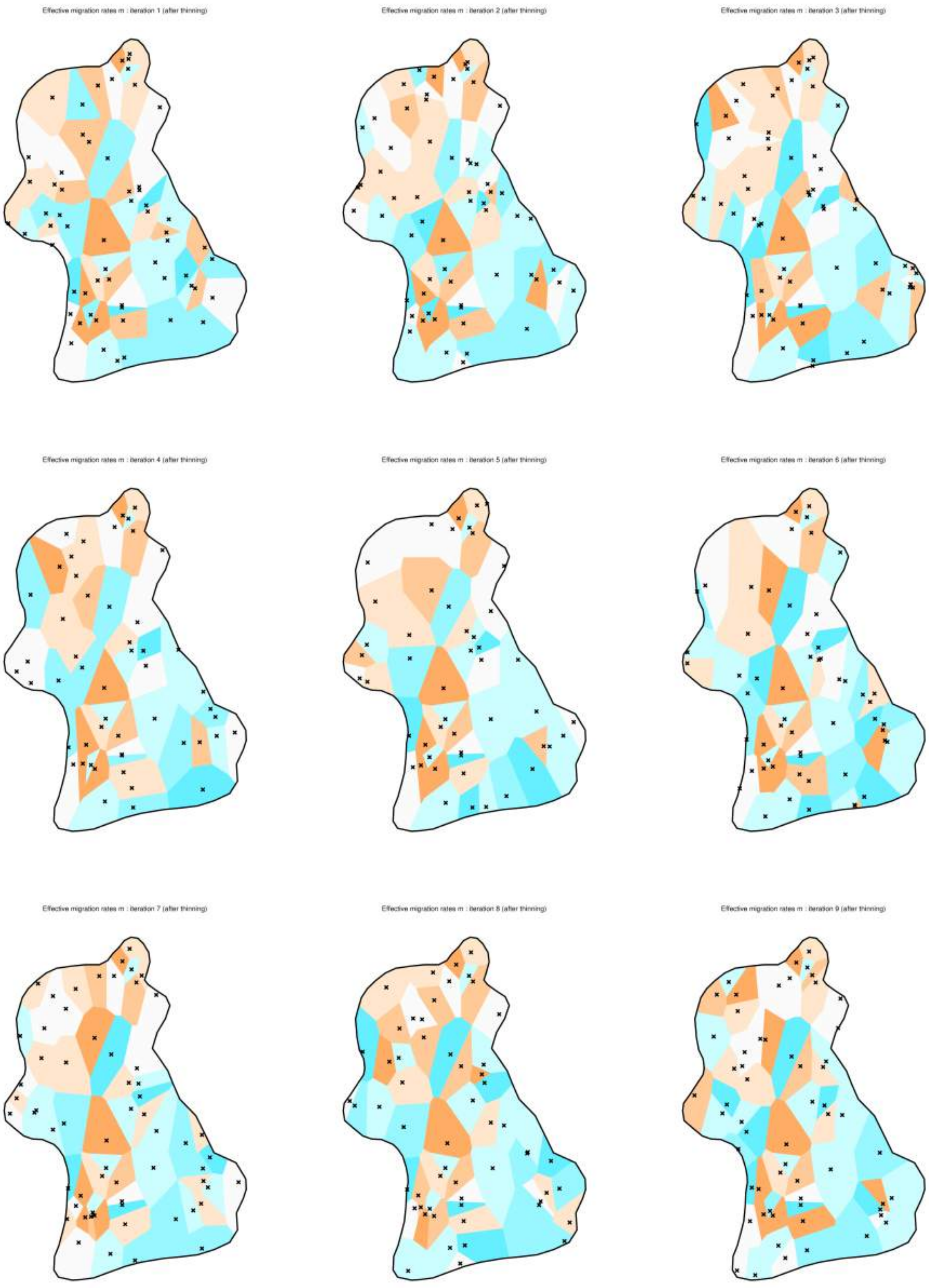
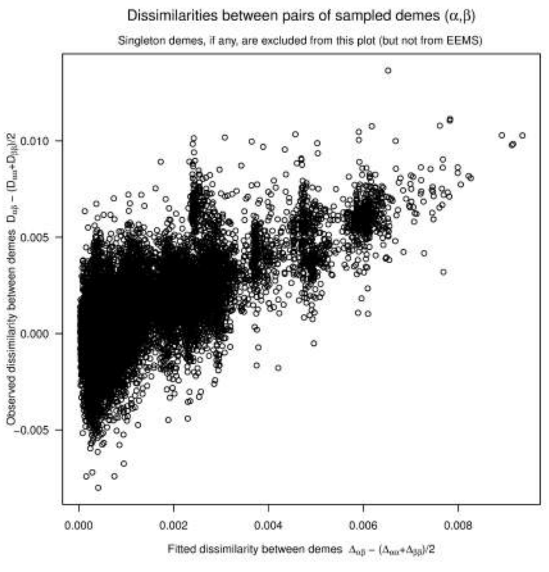
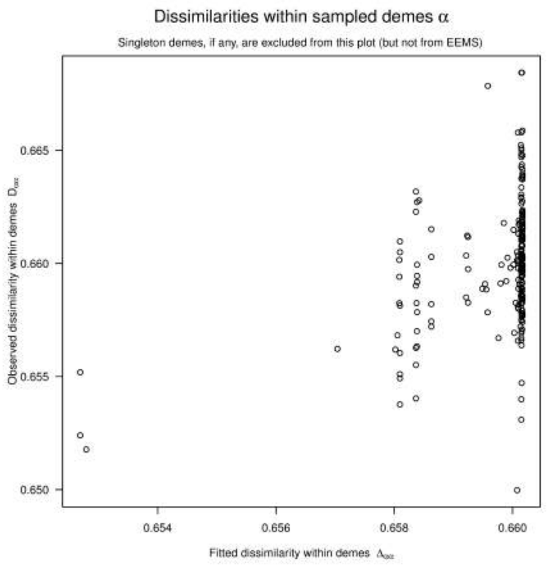

Effective migration rates m : iteration 1 (after thinning)
Effective migration rates m : iteration 2 (after thinning)
Effective migration rates m : iteration 3 (after thinning)
Effective migration rates m : iteration 4 (after thinning)
Effective migration rates m : iteration 5 (after thinning)
Effective migration rates m : iteration 6 (after thinning)
Effective migration rates m : iteration 7 (after thinning)
Effective migration rates m : iteration 8 (after thinning)
Effective migration rates m : iteration 9 (after thinning)

Figure 10 Voronoi diagrams drawn from the posterior distribution of the effective migration rates m, generated by `eems.voronoi`.



(a)



(b)

Figure 11 Observed vs fitted genetic dissimilarities, between and within demes. The observed genetic dissimilarities between individuals, D_{ij}, are averaged so that $D_{\alpha\beta} = \frac{1}{n_{\alpha\beta}} \sum_{\delta(i)=\alpha,\delta(j)=\beta} D_{ij}$ where $n_{\alpha\beta}$ is the number of pairs (i, j) such that $\delta(i) = \alpha$, $\delta(j) = \beta$, i.e., sample i is assigned to deme α, sample j is assigned to deme β, and i, j are distinct individuals. Singleton demes (those with a single sample) are excluded from both scatterplots. **(a)** Dissimilarities between demes: the fitted values $B_{\alpha\beta} := \Delta_{\alpha\beta} - (\Delta_{\alpha\alpha} + \Delta_{\beta\beta})/2$ comprise the between-demes component of the genetic dissimilarity, B, which is modeled by the effective migration rates: $B = f(m)$. **(b)** Dissimilarities within demes: the fitted values $w_\alpha := \Delta_{\alpha\alpha}$ comprise the within-demes component of the genetic dissimilarity, w, which is modeled by the effective diversity rates: $w = g(q)$. If the EEMS model fits the data well (after the MCMC chain has converged), we expect to see a strong linear relationship between the observed and fitted values in both scatterplots. See Section 6 for more details.

References

[Hudson, 2002] Hudson, R. R. (2002). Generating samples under a Wright-Fisher neutral model of genetic variation. *Bioinformatics*, 18:337–338.

[Lawson et al., 2012] Lawson, D. J., Hellenthal, G., Myers, S., and Falush, D. (2012). Inference of population structure using dense haplotype data. *PLoS Genet.*, 8:e1002453.

[Leslie et al., 2015] Leslie, S., Winney, B., Hellenthal, G., Davison, D., Boumertit, A., Day, T., Hutnik, K., Royrvik, E. C., Cunliffe, B., 2, W. T. C. C. C., Consortium, I. M. S. G., Lawson, D. J., Falush, D., Freeman, C., Pirinen, M., Myers, S., Robinson, M., Donnelly, P., and Bodmer, W. (2015). The fine-scale genetic structure of the British population. *Nature*, 519:309–314.

[Winney et al., 2012] Winney, B., Boumertit, A., Day, T., Davison, D., Echeta, C., Evseeva, I., Hutnik, K., Leslie, S., Nicodemus, K., Royrvik, E. C., Tonks, S., Yang, X., Cheshire, J., Longley, P., Mateos, P., Groom, A., Relton, C., Bishop, D. T., Black, K., Northwood, E., Parkinson, L., Frayling, T. M., Steele, A., Sampson, J. R., King, T., Dixon, R., Middleton, D., Jennings, B., Bowden, R., Donnelly, P., and Bodmer, W. (2012). People of the British Isles: preliminary analysis of genotypes and surnames in a UK-control population. *Eur. J. Hum. Genet.*, 20:203–210.