Population genetics analysis using R and Geneland

Gilles Guillot*, Filipe Santos, Arnaud Estoup

September 25, 2008

Contents

*Centre for Ecological and Evolutionary Synthesis, Department of Biology, University of Oslo, email: gilles.guillot@bio.uio.no

1 Overview

1.1 About Geneland

1.1.1 History

The work around Geneland started in 2002-2003 from discussions between statisticians, population geneticists and ecologists at INRA in Avignon and Montpellier in France. Gilles Guillot designed the model and wrote the initial Fortran and R code in 2003-2004. This resulted in a regular R package released in 2005. People who helped designing and improving this initial package include: Annie Bouvier, Aurélie Coulon, Jean-Francois Cosson, Arnaud Estoup and Frédéric Mortier. Filipe Santos joined the project in 2007 and wrote the graphical user interface in R-Tcl/Tk together with Arnaud Estoup.

The program also benefitted from comments from users of the Geneland mailing list and also from several people at the Conservation Genetics summer school in Liblice and in Porto in September 2008.

Subsequent developments include a scheme accounting for the presence of null alleles, an improvement of the inference technique under the correlated frequency model and a more efficient post-processing scheme. The GUI is currently maintained by Filipe Santos and the R and Fortran codes are maintained by Gilles Guillot.

1.1.2 Contact, info, mailing list

Geneland is distributed through the Comprehensive R Archive Network (CRAN). It consists of a network of mirroring sites throughout the world. This distribution method is very efficient but does not allow to know how many users have dowloaded or used a specific package. In order for people developping Geneland to have an idea about that and also for users to be informed of updates and related publications, a mailing list is operated.
Please register on `folk.uio.no/gillesg/Geneland/register.php`.

Although this new manual is intended to replace previous sources of information, you may still find additional details on the Geneland homepage `folk.uio.no/gillesg/Geneland/Geneland.html`.

Please report bugs in the package or typos and mistakes in this manual to `gillesg @ bio . uio .no`.

1.1.3 How to request help

If you are experiencing troubles with Geneland, before requesting help, please

- make sure you are using the latest versions of R and Geneland

- read carefully this documentation

- make sure you are able to describe and reproduce the problem you are experiencing

- make sure you are a registered Geneland user

- send enough data and information about what you did in R for us to reproduce the problem, namely

  - if your problem is related to the use of Geneland with the R command line, give the exact sequence of commands that causes the problem

– if your problem appears with the GUI, please send the file `ExecLog.txt` located in the output directory

1.1.4 Citation

Developping, improving and maintaining Geneland represents a tremendous amount of work. If you use it for your own scientific work, please cite the related publications (Guillot et al. [2005a],Guillot et al. [2005b],Guillot et al. [2008],Guillot [2008]) detailed in the reference list at the end of the present document.

1.2 System and hardware requirements

1.2.1 Operating system

Geneland is an add-on to the statistical software R. R is a free software and is becoming a standard in many research communities, in particlular in bioinforamtics.
See `http://en.wikipedia.org/wiki/R_(programming_language)` for details.

To install and run Geneland, you need first to have R installed on your computer. R is available for MS-Windows, Linux and Mac-OS.
See `http://cran.r-project.org` .

1.2.2 Memory

Computations in Geneland are carried out through a so-called Markov Chain Monte-Carlo (MCMC) technique. This implies that the overall computing task consists of a (very) long sequence of rather simple tasks. A small set of variables is stored in RAM, updated sequentially and written to the disk from time to time. This requires a few Mb of RAM. The exact amount varies with datasets and computing options but it is fulfilled by any computer.

1.2.3 Disk space

The amount of disk space required depends on which fraction of the computations are stored on the disk. This amount can be fairly large from a few tens of Mega bytes to several Giga bytes (see section A.11 for details).

1.2.4 Computer speed

The model implemented in Geneland is fairly complex. A run of 100000 iterations for a dataset of 200 individuals at 10 loci) takes typically 3-15 minutes with a computer equipped with a 2 GHz chipset. This time could vary with the specific modelling options chosen.

1.3 Installation

1.3.1 Installing R

Instructions for installation of R are continuously updated.
See `http://cran.r-project.org/sources.html` for details and update.
Typically, under Windows:

- Go to the Windows binary repository `http://cran.r-project.org/bin/windows/base/`

- Download the executable R-x.x.x-win32.exe

- Launch this executable.

Usefull sources of information include the various R manuals [1] and [Paradis, 2005, 2006] among others.

1.3.2 Installing Geneland

Once R is installed,

- launch R

- type `install.packages("Geneland")`.

- follow the instructions.

Geneland is based on the add-on packages `field`, `RandomFields`, `maps`,`mapproj` and `snow`. They have to be also installed as they do not belong to the R base distribution. All packages can be installed at the same time bia the command-line by typing `install.packages(c("Geneland","fields","Rand`

The use of the graphical interface for parallel computations with `snow` requires also packages and `Rmpi` or `rpvm`.

Under Mac-OS, make sure that X11 is installed and running when you install and use R. If you do not have X11 installed, you can find it on the Mac-OS installation DVD.

1.4 Tasks performed

1.4.1 Estimating the number of panmicitic groups and locating their spatial boundaries

The main task of Geneland consists in clustering a sample of population genetics data into a certain number of populations in such a way that each population is approximately at Hardy-Weinberg equilibrium with linkage equilibrium between loci (HWLE). Different algorithms based on different models are implemented. The most popular algorithm is based on a spatial model and makes use not only of genotypes but also of spatial coordinates of sampled individuals (or populations).

1.4.2 Input files

The research project that lead to the development of Geneland was focused on the combined use of genetics and geographic informaton to understand the factor affecting gene flow accross space. Hence, a typical dataset treated by Geneland consists of

- a file containing the genotypes of n haploid or diploid individuals at L co-dominant markers (micro-satellites, SNPs);

- a file containing the spatial coordinates representative of each individual.

This second file is actually optionnal and Geneland can also be used without spatial information. See section 3 for detail about data format.

---

[1]available from `http://cran.r-project.org`

1.4.3 Output files and graphics

The output of Geneland consists of

- an estimation of the number of HWLE populations,

- a map of the geographic spread of these various populations,

- a file giving the estimated population membership of each individual

- a file giving the estimated population membership of pixel of the study domain (the size of the pixel being prescribed by the user).

Optionnal computing options allow to

- account for null alleles (diploid data only)

- account for spatial coordinates uncertainty

Additional outputs include

- computation of pairwise population F_{ST}

- computation of individual population F_{IS}

2 Models

Three types of quantities are involved:

- the (usually unknown) number of populations K

- the parameters (or hidden vairable) coding for population membership (of individuals and pixels)

- the parameters of the genetic model conditionnally on the the number of populations and on population memberships.

They are modelled separately. K is assumed to follow a uniform distribution between 0 and an upper bound $K_{\max}$ prescribed by the user. The genetic and the spatial model are specified conditionnally on K. This is described below.

2.1 Genetic models

2.1.1 Within group Hardy-Weinberg equilibrium and linkage equilibrium

It is assumed that the overall dataset consists of individuals belonging to K populations, each of these populations being at Hardy-Weinberg equilibrium with linkage equilibrium between loci. For n individuals genotyped at L loci, denoting by f_{klj} the frequency of allele j of locus l in population k, by p_i population membership of individual i ($p_i \in \{1, ..., K\}$) and by $z_{il} = (\alpha_{il}, \beta_{il})$ the genotype of individual i at locis l, HWLE writes:

$$\pi(z) = \pi((z_{il})_{il}) = \prod_{i=1}^{n} \prod_{l=1}^{L} f_{p_i l \alpha_i} f_{p_i l \beta_i} (2 - \delta_{\alpha_i}^{\beta_i}) \tag{1}$$

where $\delta_{\alpha_i}^{\beta_i} = 1$ if $\alpha_i = \beta_i$ and 0 otherwise.

2.1.2 The uncorrelated frequency model

Allele frequencies in the various sought populations are unknown and although they are not of direct interest, it is convenient to introduce them in the statistical computations. Indeed, once they are introduced and although they are unknown, Equation (1) allows to compute the likelihood. Plugging this equation in an iterative scheme known as Metropolis-Hastings algorithm allows to start from arbitrary values for all unknown parameters and to modify them in such a way that after many iterations, these values are close to the true values. The trick consisting in including extra unknown parameters in the inference not of direct interest but for computationnal purpose is known as data augmentation in statistics.

Once we have introduced the allele frequencies in the various sought populations, we need to place a prior distribution on them in view of Bayesian inference (see section B).

For each population and each locus, the entries of the vector $f_{kl1}, ..., f_{klJ_l}$ sum up to one. The simplest probability distribution fulfilling this condition is the Dirichlet distribution [2]. Beyond this algebraic property, it also has the interest to comply with a Wright-Fisher island model, the asymptotic distribution of allele frequency being Dirichlet under this model.

This distribution depends on a single vector parameter which might vary across populations and loci. Assuming that this parameter α_{kl} is not common accross populations and loci, assuming Dirichlet a priori distribution for f_{klj} writes:

$$\pi(f_{klj}) = \Gamma(J_l) \tag{2}$$

This probability does not depend on the actual values taken by f_{klj} and this model turns out to give the same a priori probability to any allele frequencies. The key assumption consists now in assuming that the vectors $f_{kl.}$ are mutually independent across populations. Independence of the vectors $f_{kl.}$ is of course assumed across loci.

2.1.3 The correlated frequency model

The previous model is somehow over simplistic in the sense that most often, allele frequencies tend to be similar across populations, in particular, rare alleles in a certain population are also rare in other populations. From a statistical point of view, this property can be viewed as a correlation of f_{klj} and $f_{k'lj}$, the correlation originating from the common recent (micro-)evolutionary history of populations k and k'.

The correlated frequencies model has been introduced previously (see e.g. [Nicholson et al., 2002]) to account exactly for this property.

In this model, one introduces the frequencies of an ancestral population denoted by f_{Alj} also assumed to be independently Dirichlet distributed and a vector of population specific drift parameters $(d_1, ..., d_K)$ so that $f_{kl.}|f_A, d$ has a Dirichlet distribution

$$D\left(f_{Al1}(1 - d_k)/d_k, ..., f_{AlJ_l}(1 - d_k)/d_k\right) \tag{3}$$

In this model and conditionally on f_A and d, the frequencies are independent across populations, but marginally (integrating out f_A and d) elementary computations (see [Guillot, 2008]) show that the correlation of allele frequencies across population is:

$$Cor(f_{klj}, f_{k'lj}) = 1 \left/ \left[1 + E[d_k]\frac{E[f_{Alj}] - E[f_{Alj}^2]}{E[f_{Alj}^2] - E[f_{Alj}]^2}\right]\right. \tag{4}$$

---

[2]See `en.wikipedia.org/wiki/Dirichlet_distribution`.

In the most general case, the distribution of the f_{klj}s in the uncorrelated model may depend on population-, locus- and allele-specific parameters α_{klj}. In practice, the α_{klj} are always assumed to be common across populations, locus and alleles, and most often set to one. Similarly, in the correlated model, the f_{Alj} might have locus- and allele-specific parameters but I do not consider this case here. I set it to one as it is most often done in practice, although the effect of this assumption has not been yet thoroughly assessed in the context of clustering (but see [Foll et al., 2008] in another context). Independence is always assumed across loci.

Specifying fully the model also requires to place a prior on the drift parameters d_k. As this parameter has to lie in $[0, 1]$, it is natural to consider a Beta prior, with independence across populations. A Beta distribution depends on two parameters.

The correlated model can be viewed as a Bayesian and biologically grounded way to make inference under the uncorrelated model with population-, locus- and allele-specific parameters.

It has been observed that using the correlated frequency model could be more powerfull at detecting subtle differentiations. On the other hand, this model seems to be more prone to algorithm instabilities and more sensitive to departure from model assumptions. It is recommended to start data analysis with the uncorrelated frequency model and then to check how these initial results are modified by the use of the correlated frequency model. It is also important to check ex-post that the inferred groups are significantly differentiated and at HWLE.

2.2 Models underlying population membership

2.2.1 The non-spatial model

I denote by p a vector parameterizing the population memberships. In case population membership is modeled at the individual level, this vector can be simply $p = (c_1, ..., c_n)$ where $c_i \in \{1, ..., K\}$; In this case, the simplest form of prior that can be placed on it is an i.i.d prior $\pi(p|K) = 1/K^n$.

Spatial representation for six simulations from this prior model are given on figure 1.
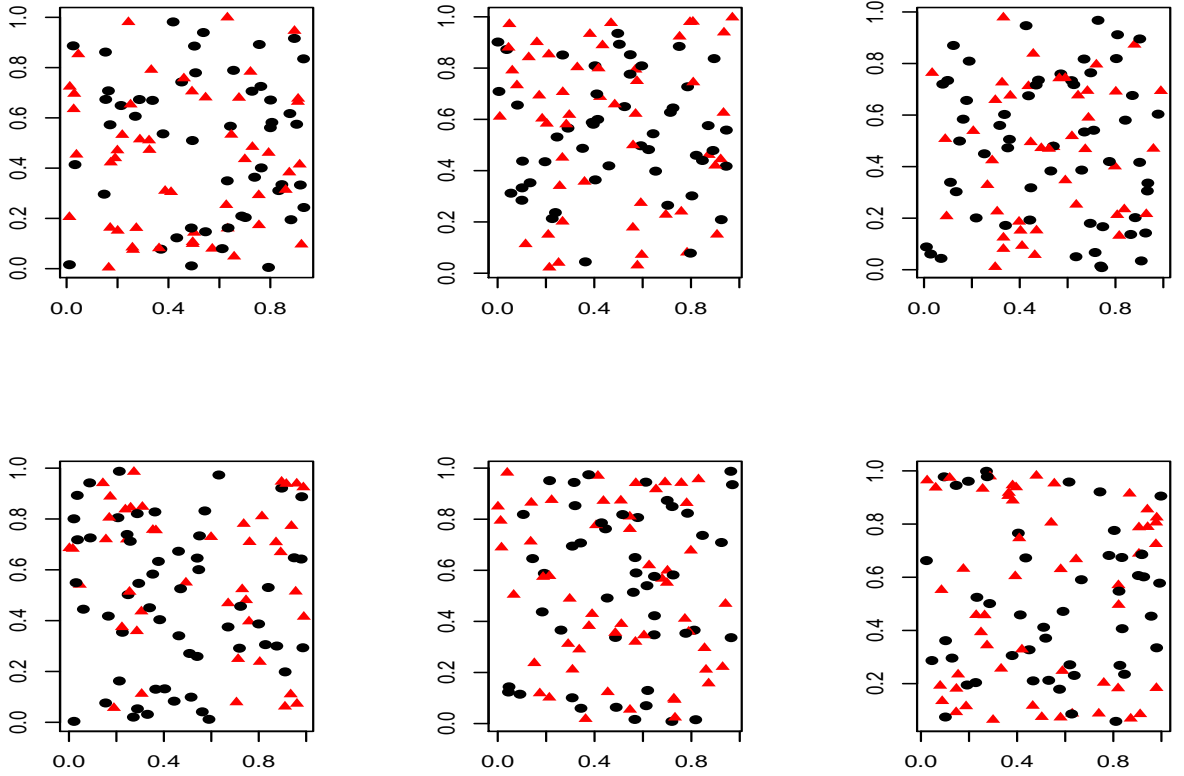
Figure 1: Six examples of 100 individuals belonging to two populations spread totally at random across space.

The previous model is numerically convenient but it is biologically questionable in the sense that one may wonder how differentiation might have occured between populations in case of such spatial overlap between them.

2.2.2 The spatial model

There different ways to specify a model in which the various population sought are assumed to display a weak spatial overlap. The so-called colored Poisson-Voronoi tessellation model is one of them. This model consists in assuming that spatial domain of each population can be approximated by the union of a few polygonal domains, see figure 2 for examples.

This model corresponds to the spatial patterns that can be expected when differentiation occurs

11

Figure 2: Six examples of 100 individuals belonging to two populations where the spatial domain of each population can be approximated by the union of a few polygonal domains.

by limited gene flow induced by the presence of physical barriers such as road, rivers, mountain ranges, human activity.

Formally, the colored Poisson-Voronoi tessellation model consists in assuming that there is an unknown number of polygons m that approximate the true pattern of population spread across space. These polygons are "centered" (this term is actually a bit inaccurate mathematically, but see below) around spatial points $u_1, ..., u_m$ and each polygon belongs to one of the K population which is coded by an integer (or color for grahical representation) denoted by $c_1, ..., c_m$. Examples are given on figure 4.

The exact mathematical definition of the colored Poisson-Voronoi tessellation model is as follows:

- the number of polygons follows a Poisson distribution with paramter λ: $m \sim \text{Poisson}(\lambda)$

- conditionnally on m, there are m mutually independent points $u_1, ..., u_m$ with unifrom distribution over the study domain D: $(u_1, ..., u_m)|m \overset{\text{i.i.d}}{\sim} \text{Uniform}(D)$

- each points u_i defines a set V_i of points in D that are closer to u_i than to any other points in $(u_1, ..., u_m)$. This set V_i is the i-th cell of the so-called Voronoi tessellation induced by $(u_1, ..., u_m)$.

- the points $(u_1, ..., u_m)$ (or now equivalently the sets $V_1, ..., V_m$) receive a mark with value in

$\{1, ..., K\}$ coding for population membership and displayed as different colors. These colors a assumed to be sampeld from a probability distribution, which in Geneland (but this is not a requirement) is assuemd to be an independent, identically distributed and uniform distribution on $\{1, ..., K\}$: $(c_1, ..., c_m)|m \overset{\text{i.i.d}}{\sim} \text{Uniform}(\{1, ..., K\})$
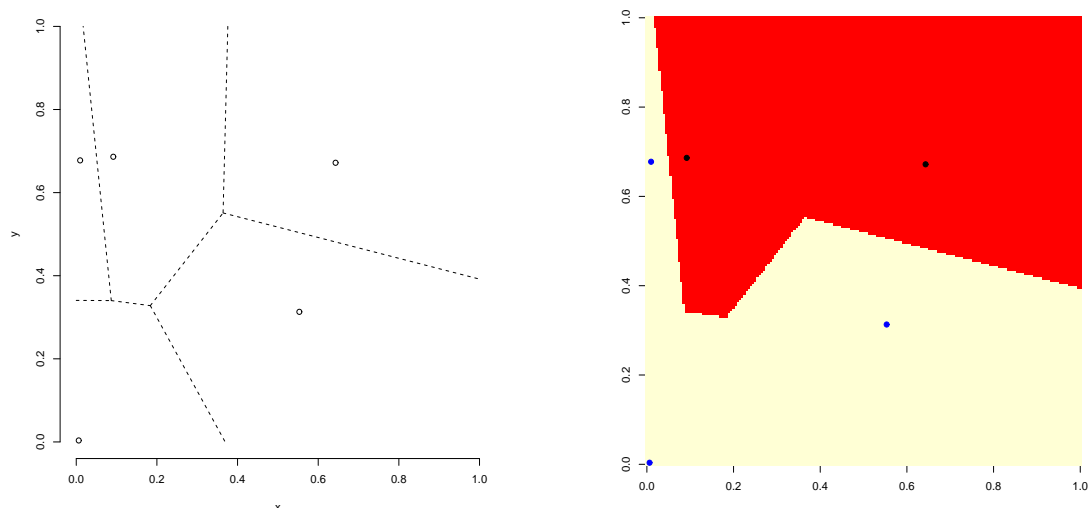
Figure 3: Example of colored Poisson-Voronoi tessellation. Left panel: location of cell "center" and voronoi cells induced. Right panel: a example of colored tessellation obtained after coloring at random each cell as red or white. In this example, the number of population $K = 2$ and the number of polygons $m = 5$. There is no attempt to reprensent the location of any sampled individual.

2.3 Additional modeling options

2.3.1 Null alleles

A well known source of potential problems is the presence of null alleles arising from variation in the nucleotide sequences of flanking regions that prevent the primer annealing to template DNA during PCR amplification of the microsatellite locus [Dakin and Avise, 2004, Pompanon et al., 2005]. The presence of null alleles results in an excess of homozygous genotypes within a population as compared to the expected proportion under Hardy Weinberg Equilibrium (HWE) and Linkage Equilibrium (LE) [Callen et al., 1993, Paetkau et al., 1995] while the model in Geneland is based on HWE and LE within the sought clusters.

In Geneland, the putative presence of null allele(s) can be explicitly taken into account for diploid data through an optionnal computing step. When this option is used, genotype ambiguity (homozigotes) is accounted for and null alleles frequencies is estimated along the clustering algorithm. With the option `filter.NA=TRUE` in function `MCMC`, null allele frequencies at each locus are estimated. They can be viwed with function `EstimateFreqNA`. This is also available through the GUI.

Note that if this option is used, all double missing genotypes will be interpreted as double null alleles. This can induce a large over-estimation of null allele frequencies if some of the missing data
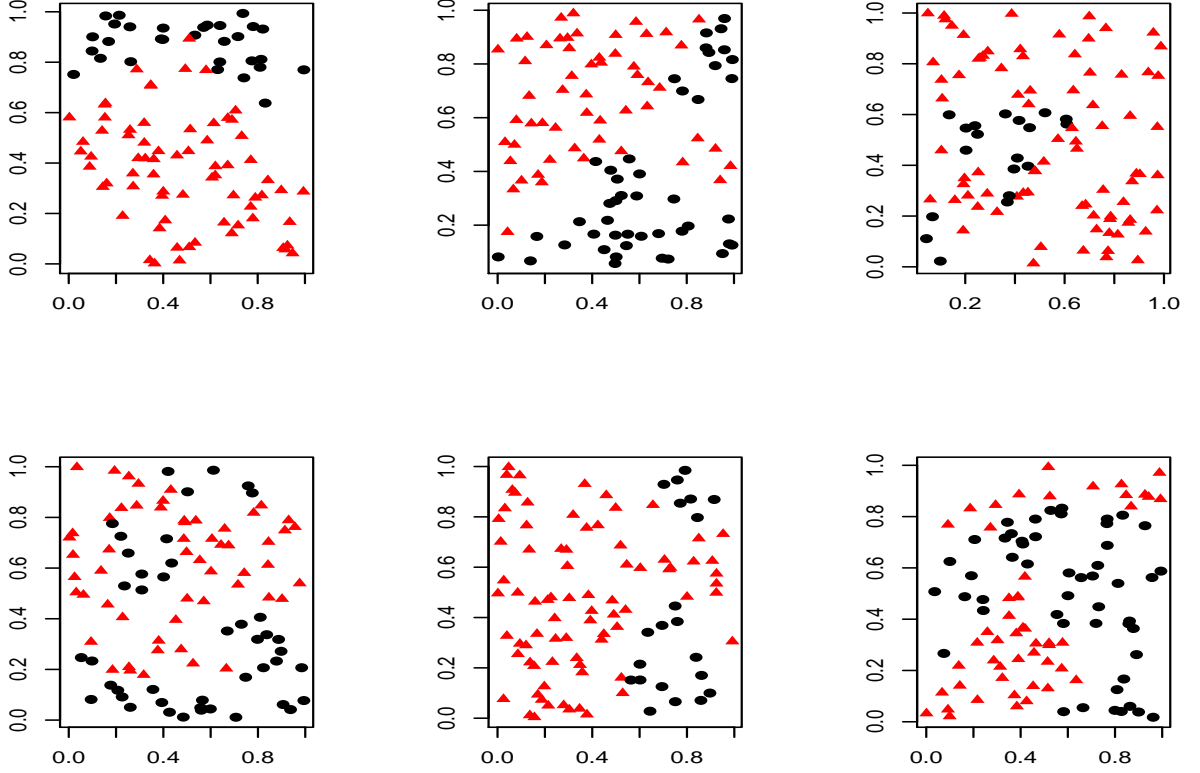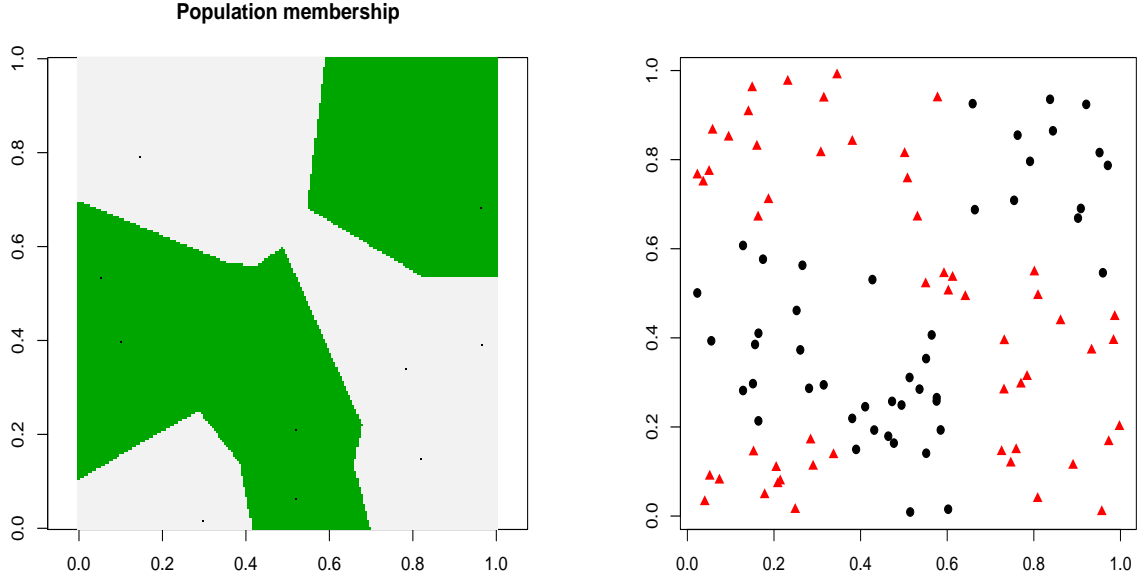
Figure 4: An example of 100 individuals belonging to two populations where the spatial domain of each population can be approximated by the union of a few polygonal domains. Left: spatial spread of individuals, right: corresponding polygons. In this example, $n = 100$, $K = 2$ and the number of polygons $m = 10$. The points u_i are depicted as tiny black dots on the left hand side panel. They are mot depicted on the right hand side for clarity.

are not null alleles (i.e. not due to amplification problems in the PCR), for example if some loci are missing for all individuals of certain sampling units.

2.3.2 Coordinates uncertainty

You may want to treat the spatial coordinates as uncertain for any or a combination of the following reasons:

1. The individuals under study are non mobile (plants, animals species with very limited vagility as compared to the scale of the study domain) but the coordinates have been recorded with an error or they have been recorded with a limited precision only (e.g. each individual has been affected to a small administrative unit).

2. The individuals under study are normally non mobile but a displacement might have been induced by the observation (e.g. hounding by hunters).

3. The individuals under study are mobile and they have a home range whose characteristic scale is non negligible as compared to the size of the study domain.

4. Even if none of the previous conditions holds, but if your dataset has samples sharing the same coordinates, then allowing some uncertainty in the coordinates will allow to have samples with the same coordinates to be assigned to different populations. It can be usefull to detect migrants.

Under conditions (1) and (2), the notion of "true" coordinates makes sense and refers to the location where each individual normally leaves, but these locations have not been observed. Under condition (3), there is no "true coordinates", i.e. there is no particular location where each individual could be considered to leave with certainty. In any of the previous conditions, the use of the recorded coordinates as locations where the individuals leave is innacurate and can be misleading in the inferences. It is recommended in this case to treat the observed coordinates as uncertain. The way Geneland does it is to consider that the observed coordinate of each individual is the sum of the true coordinate and of a random noise of small magnitude. This random noise can be interpreted as the movement induced by capture in case of hounding (condition b), in the other cases it says that an individual has been observed somewhere but that it could have been observed anywhere around as well.

3 Data format

3.1 Genotypes file

Assuming that you have data for n diploid individuals genotyped at L loci, the data must be arranged in

- a plain ascii file

- without any extra invisible characters (like in MS-Word .doc files)

- with one line and $2 \times L$ columns per individual

- each allele must be coded by an integer

- the number of digits of each field is arbitrary and can vary across loci

- extra header lines are allowed, the way these lines are handled when the data are loaded is prescribed through the arguments of the R function `read.table`.

- missing data are allowed. Users of the GUI have to specify the character string coding missing value in the `missing data symbol` sub-menu of the main menu. For users of the R command-line interface loading their data with the R function `read.table`, missing values can be coded in the raw text data file by any arbitrary character string, (e.g. `000`, `00`, `NA` or `-999`). The character string used to encode missing data must be specified through the argument `na.strings` of function `read.table`. By default, when data are loaded through `read.table`, it is assumed that missing data are coded as `NA`.

- for haploid organisms with L loci, the genotpye file must have L columns.

For $L = 10$, the two first lines of the genotype file should look like:

```
198 000 358 362 141 141 179 000 208 224 243 243 278 284 86 88 120 124 238 244
200 202 000 358 141 141 183 183 218 224 237 243 276 278 88 88 120 124 240 244
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .  .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .  .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .  .   .   .   .   .
```

In the example above, missing values are coded as `000`.

3.2 Coordinates file

If you want to use the spatial model, you should also have a file containing spatial coordinates of the individuals sampled. The coordinate file must be such that

- there is one line per individual and two columns (x-axis and y-axis coordinate),

- the units do not matter

- it is assumed that the coordinates are planar coordinates such as Lambert coordinates [3]

- whether your coordinates are genuine planar coordinates or spherical coordinates does not appear in the file format so that you can actually also input coordinates given as spherical coordinates. Unless you pre-process them in R, they will interpreted as planar coordinates by the Geneland functions. The quality of this approximation varies from very good (small domain, close to the equator) to very bad (large domain, close to a pole). Spherical coordinates (lon, lat) can be converted into planar coordinates using R function `mapproject` of package `mapproj`.

- extra header lines are allowed, the way these lines are handled when the data are loaded is prescribed through the arguments of the R function `read.table`.

- missing data are not allowed in this file.

- if some coordinates are missing, you can either subsitute an estimated value or do the analysis with Geneland without spatial coordinates at all using the non spatial model.

The two first lines of a coordinate file should look like:

```
25.6 745.2
54.1 827.8
  .    .
  .    .
  .    .
  .    .
```

---

[3]See `en.wikipedia.org/wiki/Lambert_azimuthal_equal-area_projection`

4 Example of data analysis using the graphical user interface (GUI)

4.1 Launching the GUI

- launch R

- load Geneland by typing `Geneland()` in the R prompt

- launch the Geneland GUI by typing `Geneland.GUI()` in the R prompt.

This will open open a window as below:
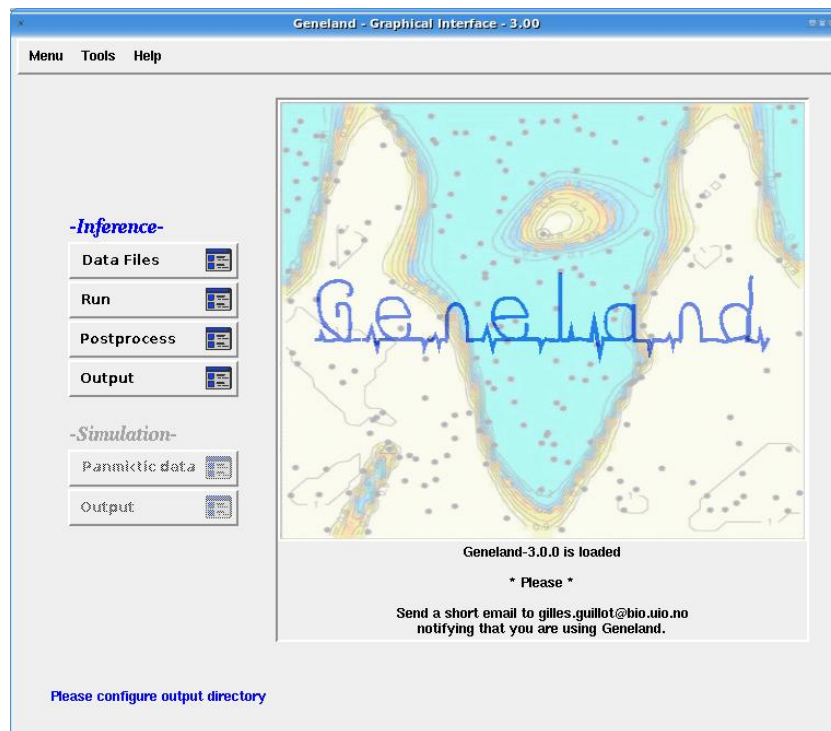
4.2 Selecting input and output files

Select the fields separator in your data files:
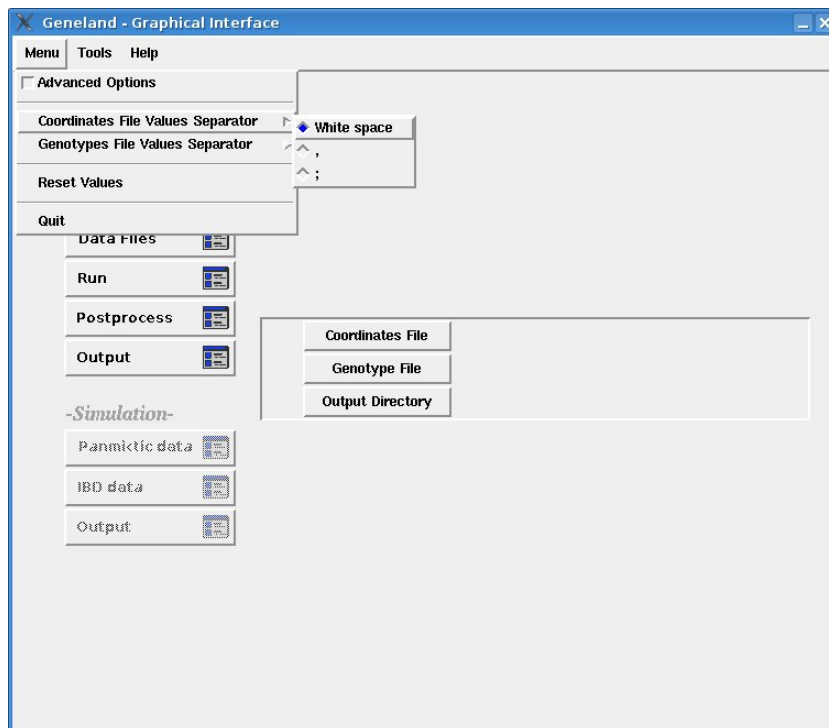
Select your data files and the directory containing your output files
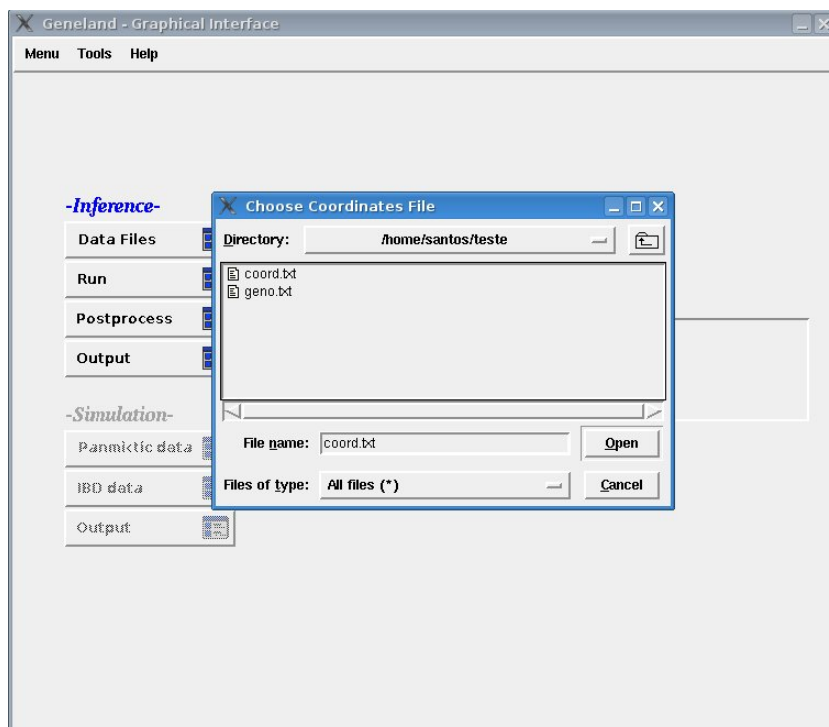
4.3 Inference

We now carry out an analysis to infer the number of populations and their spatial boundaries for this dataset. This will can be performed by selecting MCMC simulation parameters as follows:
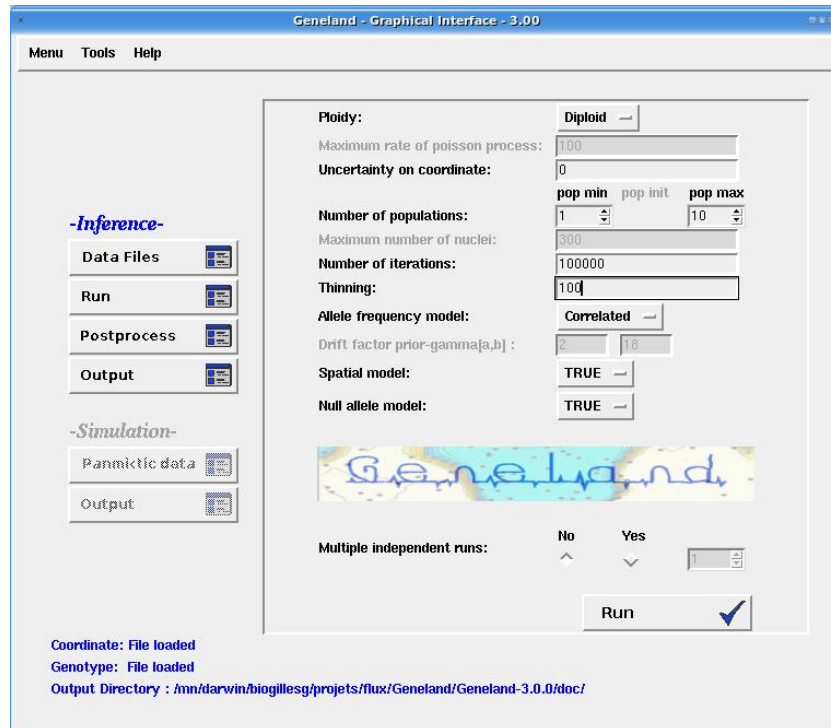
In the above example we select

- diploid organism

- no uncertainty attached to spatial coordinates

- the number of HWLE populations is unknown and hence treated as simulated variable along the MCMC simulations allowed to variy between 1 and 10

- the number of MCMC iterations will be 100000 (`nit=100000`)

- and only each 100th iteration will be saved on the disk (in total, 1000 iterations will be saved)

- combined with the correlated frequency model (`freq.model="Correlated"`).

- using the spatial model (`spatial TRUE`),

- in the current working directory (`path.mcmc="./"`).

4.4 Post-processing MCMC outputs

The call to function `MCMC` generates different files in the directory specified by the argument `path.mcmc`. Information is extracted from this file through a call to function `Post.Process.Chain` as follows:
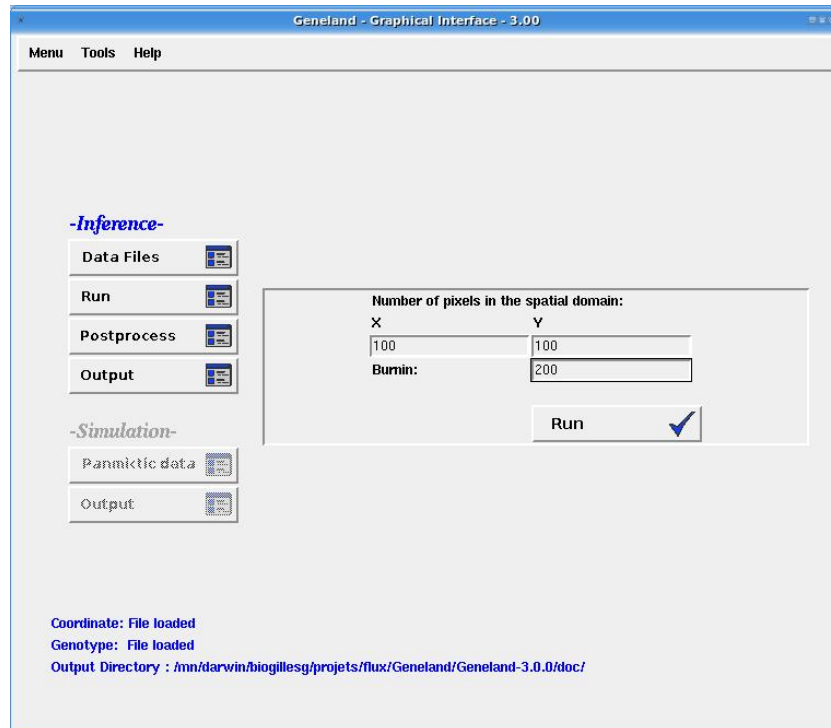
This will extract information from MCMC simulation with

- an horizontal discretization of the study domain in 100 pixels (`nxdom=100`)

- a vertical discretization of the study domain in 100 pixels (`nydom=100`)

- a burnin of 200 saved iterations, i.e. discarding the 200 first saved iterations (`burnin=200`)

4.5 Generating graphical and numerical outputs

Clik on menu output:
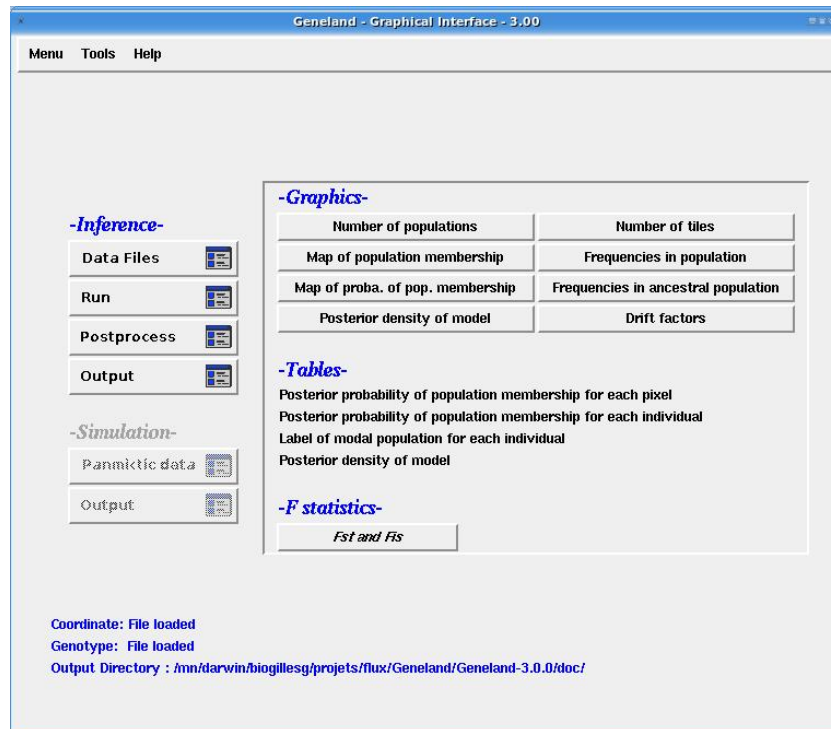
4.5.1 Estimated number of HWLE populations

The number of population simulated from the posterio distribution can be visualized by cliking on `Number of populations`

This produces a plot like:
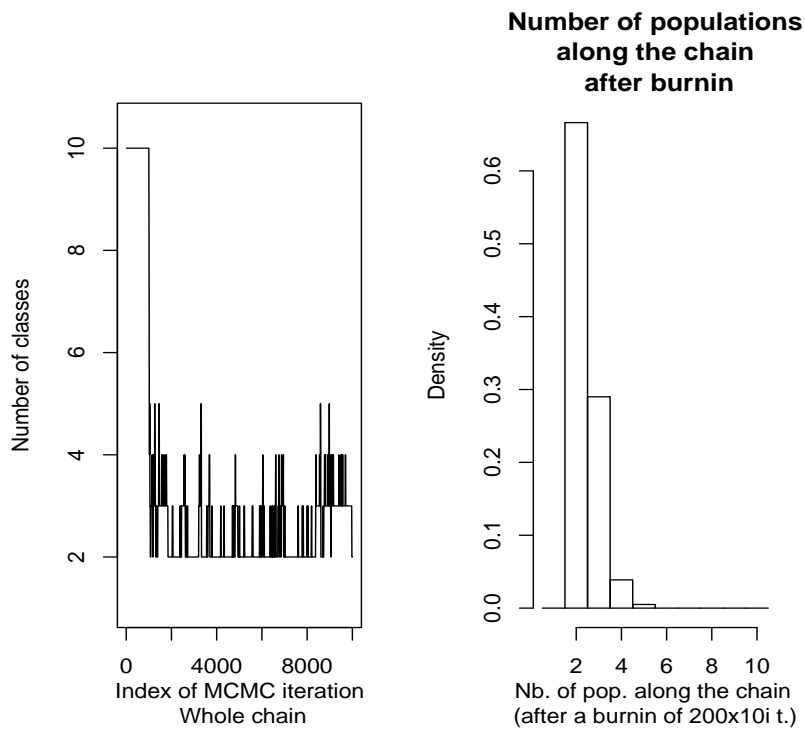
This run displays a clear mode at $K = 2$ which is hence the *maximum a posteriori* estimate of K.

4.5.2 Map of estimated population membership

A map of estimated population membership (by posterior mode) can be obtained by clicking `Map of population membership`

This produces a plot like:
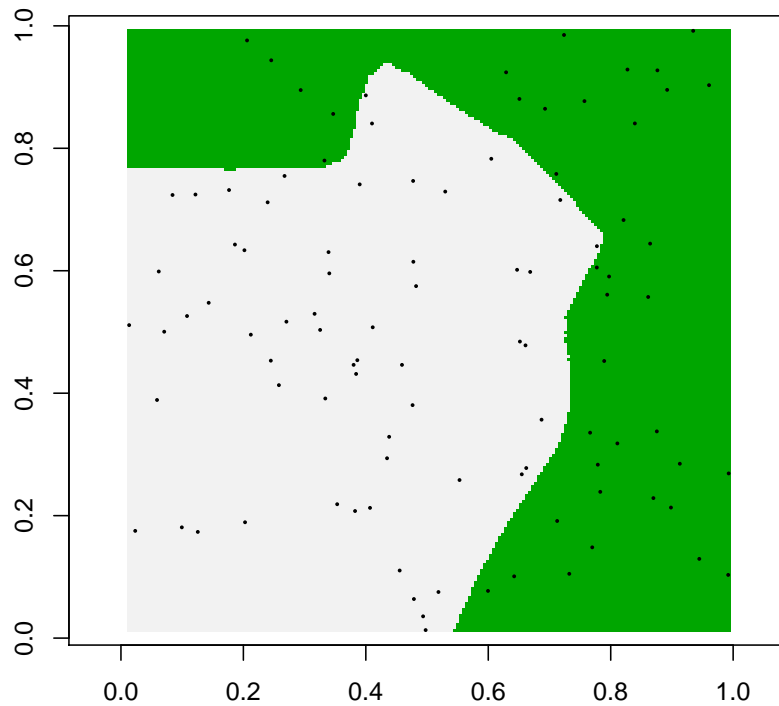
Posterior mode of population membership

4.5.3 Saving graphics

4.5.4 F statistics

F statistics Weir and Cockerham [1984] relative to estimated clusters are obtained through

```
Fstat.output(genotypes=geno,path.mcmc="./")
```

which returns:

```
$Fis
[1] 0.1427044 0.1512689

$Fst
           [,1]       [,2]
[1,] 0.00000000 0.03562966
[2,] 0.03562966 0.00000000
```

namely, individual F_{IS} and pairwise F_{ST} for estimated clusters.
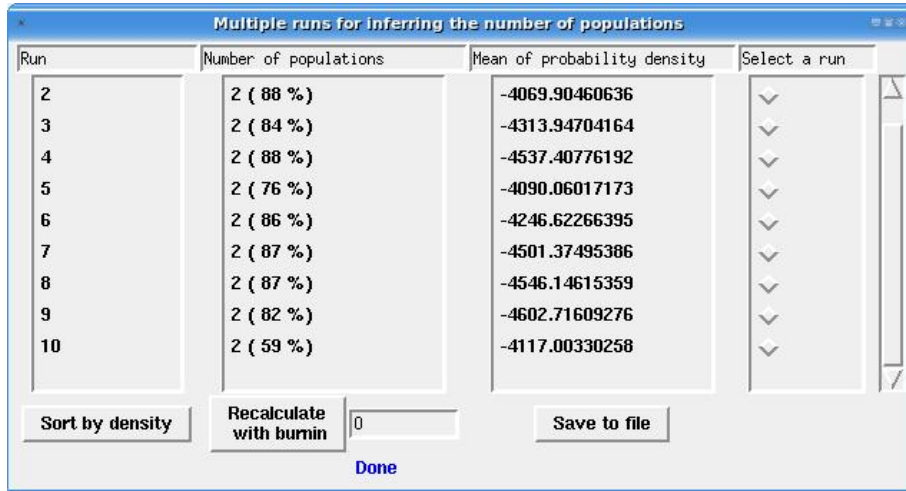
4.6 Launching several independent runs

Inference is based on a stochastic method, i.e. for a given dataset, the values of estimated parameters are random and depend on what happened during the run. In theory, different runs should give approximately the same estimates provided they are long enough. A good way to check that a run was long enough is to launch different runs and check that they provide approximately the same parameter estimates (K, individual population membership, maps).

This can be done automatically by clicking on the `Multiple independent runs` options:
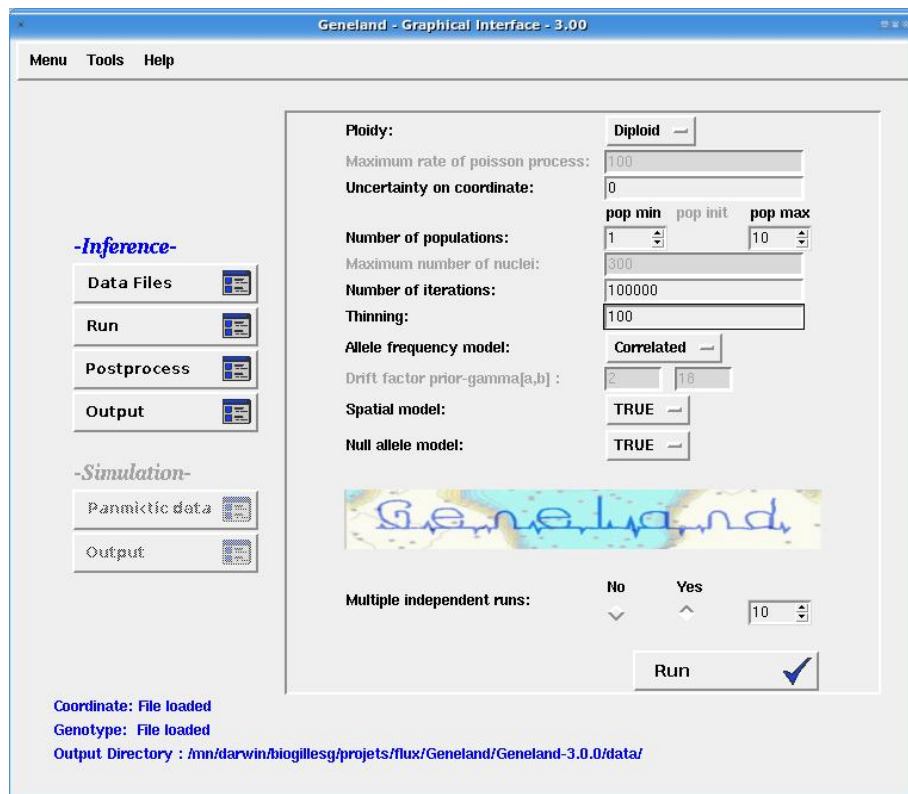
Runs are launched sequentially and computations are monitored in a new window:

Multiple runs for inferring the number of populations

Run	Number of populations	Mean of probability density	Select a run
2	2 (88 %)	-4069.90460636	∨
3	2 (84 %)	-4313.94704164	∨
4	2 (88 %)	-4537.40776192	∨
5	2 (76 %)	-4090.06017173	∨
6	2 (86 %)	-4246.62266395	∨
7	2 (87 %)	-4501.37495386	∨
8	2 (87 %)	-4546.14615359	∨
9	2 (82 %)	-4602.71609276	∨
10	2 (59 %)	-4117.00330258	∨

Sort by density | Recalculate with burnin | 0 | Save to file

Done

Here, the results of computations are consistent across runs in terms of estimated number of populations K. If different runs give different results, it is recomended to base conclusion on the run giving the highest average posterior probability (run # 2 above).

The MCMC output of these run is saved in the output directory. The output of each run is saved in a separate directory (named 1,2,...,). The global ranking of the runs can also be saved in a file.

5 Example of data analysis using the R command-line

5.1 Preliminary steps

5.1.1 Organising your session

If you plan to work through the R command-line, do not type directly the command in the prompt. Type them first in a data editor (the built-in commad line editor under Windows or emacs under Linux). This allows you to keep a trace of your work not only as numerical output but also as something looking like a computer program that you can re-use, correct, modify, share later. In addition, storing R code often takes far less disk space than storing the numerical output.

5.1.2 Launching Geneland

Assuming R and Geneland are installed,

- launch R

- you can launch the on-line help by typing `help.start()` in the R prompt (optionnal but very useful)

- load Geneland by typing `library(Geneland)` in the R prompt

- under Mac-OS, make sure that X11 is launched (see section 1.3.2).

5.1.3 Loading the data

Let us assume that the data file(s) are named `genotypes.txt` and `coordinates.txt` and stored in a directory called `data`. Then type in the R prompt:

```
geno <- read.table("../data/genotypes.txt")
```

... the genotypes are loaded and stored in an R object called `geno`.

Then type again in the R prompt:

```
coord <- read.table("../data/coordinates.txt")
```

... the coordinates are loaded and stored in an R object called `coord`.

Generally you can replace `../data` by any string `path_to_my_data` giving the path to the data relatively to the working directory. Under Windows this working directory is specified through the menu file and sub-menu preferences. Under Linux, the R working directory is the Linux working directory of the terminal from which R was launched.

5.1.4 Checking the data

You can control that you have correctly loaded your data by typing the names in the prompt, e.g. :

```
> coord[1:10,]
```

will print the ten first line of the coordinates.

The objects are a bit too large to be visalized in the R shell, it is more convenient to watch them throught the buil-in data editor:

```
fix(coord)
```

or

```
fix(geno)
```

you can check the dimension of the object:

```
dim(coord)
```

...we have indeed one line per individual and two columns, and...

```
nrow(geno)
```

...one line per individual and ...

we have indeed one line per individual and two columns, and...

```
ncol(geno)
```

two columns per locus for dipoloid data.

You can also plot the coordinates by

```
plot(coord,xlab="Eastings",ylab="Northings",asp=1)
```

...which opens a new window with the desired plot.

5.2   Inference

We now carry out an analysis to infer the number of populations and their spatial boundaries for this dataset. This is done with the main Geneland function named MCMC. A possible call of this function is as follows:

```
MCMC(coordinates=coord,
     genotypes=geno,
     varnpop=TRUE,
     npopmax=10,
     spatial=TRUE,
     freq.model="Correlated"
     nit=100000,
     thinning=100
     path.mcmc="./")
```

This will perform parameter inference by MCMC simulation assuming that

- the number of HWLE populations is unknown and hence treated as simulated variable along the MCMC simulations (`varnpop=TRUE`)

- but smaller than 10 (`npopmax=10`),

28

- using the spatial model (`spatial=TRUE`),

- combined with the correlated frequency model (`freq.model="Correlated"`).

- the number of MCMC iterations will be 100000 (`nit=100000`)

- and only each 100th iteration will be saved on the disk (in total, 1000 iterations will be saved)

- in the current working directory (`path.mcmc="./"`).

This function takes many more arguments, most of them being optionnals (i.e. with default values). An on-line help for function `MCMC` is given by typing `?  MCMC`. See section 7.

5.3 Post-processing MCMC outputs

The call to function `MCMC` generates different files in the directory specified by the argument `path.mcmc`. Information is extracted from this file through a call to function `Post.Process.Chain` as follows:

```
PostProcessChain(coordinates=coord,
                 genotypes=geno,
                 path.mcmc="./",
                 nxdom=100,
                 nydom=100,
                 burnin=200)
```

This will extract information from MCMC simulation with

- an horizontal discretization of the study domain in 100 pixels (`nxdom=100`)

- a vertical discretization of the study domain in 100 pixels (`nydom=100`)

- a burnin of 200 saved iterations, i.e. discarding the 200 first saved iterations (`burnin=200`)

5.4 Generating graphical and numerical outputs

5.4.1 Estimated number of HWLE populations

The number of population simulated from the posterio distribution can be visualized by:

```
Plotnpop(path.mcmc="./",
         burnin=200)
```

This produces a plot like

Number of populations along the chain after burnin
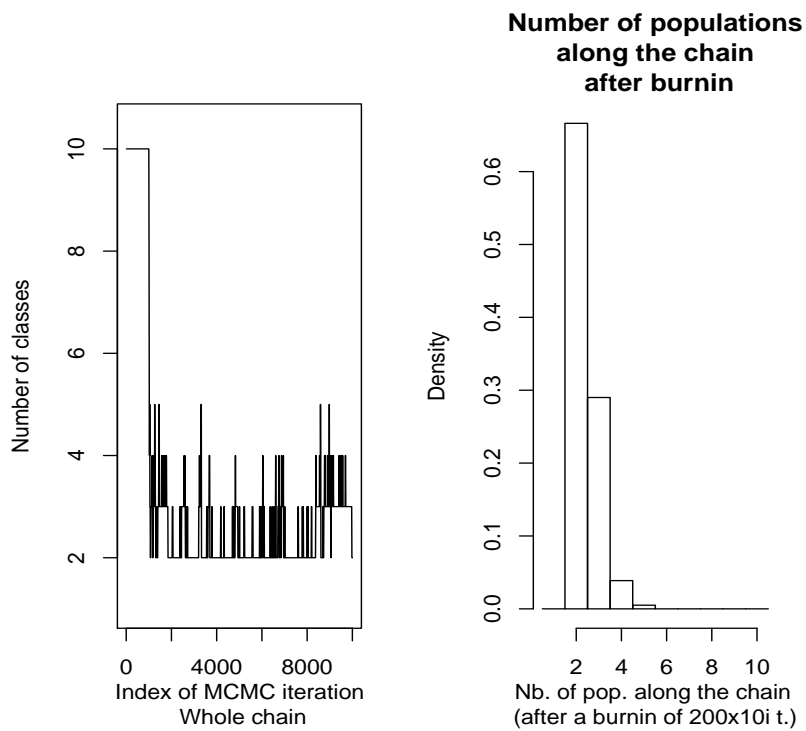
this run displays a clear mode at $K = 2$ and a relatively good mixing around this value.

5.4.2 Map of posterior probability of population membership

A call to function `PosterioMode` like:

```
PosteriorMode(coordinates=coord,
              path.mcmc="./",
              file="map.pdf")
```

will produce a plot like:
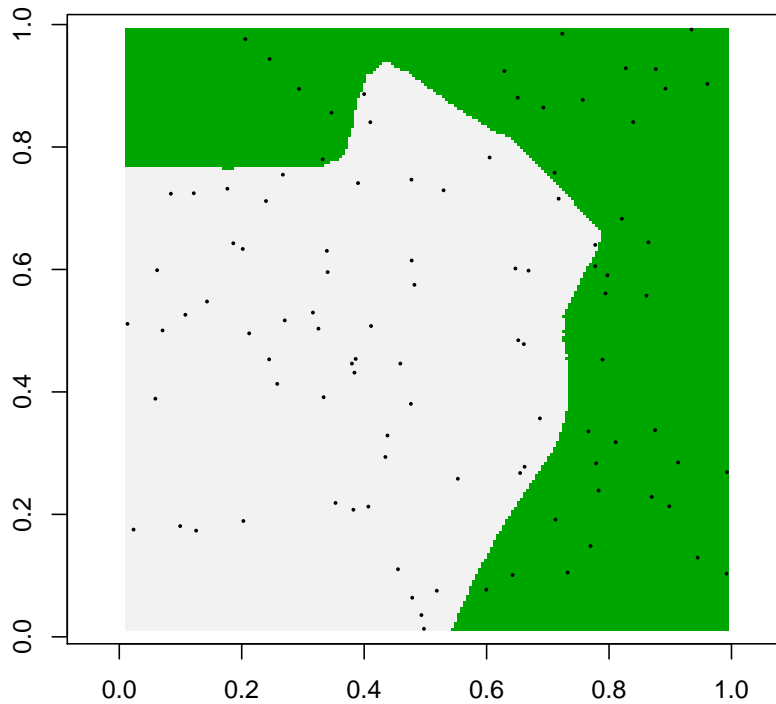
Posterior mode of population membership

5.4.3 F statistics

F statistics relative to estimated clusters are obtained through

```
Fstat.output(genotypes=geno,path.mcmc="./")
```

which returns:

```
$Fis
[1] 0.1427044 0.1512689

$Fst
           [,1]       [,2]
[1,] 0.00000000 0.03562966
[2,] 0.03562966 0.00000000
```

namely, individual F_{IS} and pairwise F_{ST} for estmated clusters.

5.5 MCMC convergence assessment

5.5.1 Checking MCMC convergence, what does that mean?

When it comes to Markov chains, convergence means that after enough iterations, the simulated vector is sampled from the desired target distribution. Since this target distribution is highly multi-dimensionnal and not much is known about it, checking convergence is not something easy to do

31

and is actually most often impossible. The best that can be done is to check that there are no obvious clues indicating a lack of convergence. MCMC behaviors indicating a lack of convergence includes:

- a single chain displaying a transient behavior, for example a clear decreasing or increasing trend from its initial value

- multiple chains leading to different estimations (number of population K, population memberships)

- a single chain stuck at a particular value or in a particular interval. Note however that in the case of a single integer parameters (for instance, the number of populations), a chain stuck at a particular value can be either a genuine feature of the posterior distribution or convergence flaw.

5.5.2 Factors affecting MCMC convergence

We have observed that mixing properties and hence convergence are affected by

- the number of individuals n, with poorer mixing properties as n increases

- the number of loci L, with poorer mixing properties as L increases

- departure from model assumptions. In case the dataset does not consist of genuine HWLE groups, it seems that the posterior distributions exhibits a ore complex pattern of local mode in which MCMC simulations get more easily trapped.

5.5.3 Example of R code as an aid to convergence diagnostic

As explained above about the multiple runs optiom in the GUI, MCMC convergence should be checked by comparing the output of several independent runs.

This can be done manually through a loop as follows:

```
## Loop for multiple runs
nrun <- 10
burnin <- 200
for(irun in 1:nrun)
  {
    ## define path to MCMC directory
    path.mcmc <- paste("./",irun,"/",sep="")
    MCMC(coordinates=coord,
         genotypes=geno,
         varnpop=TRUE,
         npopmax=10,
         spatial=TRUE,
         freq.model="Correlated"
         nit=100000,
         thinning=100
         path.mcmc=path.mcmc)
```

```
    ## MCMC postprocessing
    PostProcessChain(coordinates=coord,
                     genotypes=geno,
                     path.mcmc=path.mcmc,
                     nxdom=200,
                     nydom=200,
                     burnin=burnin)
  }

## Computing average posterior probability
## with a burnin of 200 (* 100) iterations
lpd <- rep(nrun,NA)
for(irun in 1:nrun)
  {
    path.mcmc <- paste("./",irun,"/",sep="")
    path.lpd <- paste(path.mcmc,"log.posterior.density.txt",sep="")
    lpd[irun] <- mean(scan(path.lpd)[-(1:burnin)])
  }

## Runs sorted by decreasing average posterior probability:
order(lpd,decreasing=TRUE)
```

6 Assessing influence of modelling assumptions

6.1 Chosing a model to perform MCMC simulations

Computations are obtained under specific assumptions regarding allele frequencies (correlated/ non correlated) and population membership (spatial / non spatial). First you have to chose which combination of models is the most suitable for your data. Roughly, if you expect differentiation due to the presence of simple shaped landscape features, the spatial model is presumably well suited. And if you are looking for low differentiation due to recent ecological events, the correlated allele frequencies model is more suitable.

6.2 Comparing outputs from MCMC runs under different models

The estimates under your preferred model should be compared to estimates under other models. Note that even if a model is more realistic than others from the biological point of view, the results of analysis under this "best" model can be tricked by poor MCMC mixing. This can be observed on large datasets (> 1000 individuals, $> 100 loci$) and/or due to departure from modelling assumptions.

   The most confortable situation is when different models give similar answer. In this case, there is presumably a strong signal in the data and the inferred pattern does not depend on the particular way information is extracted (model+algorithm).

In case results differ across models, our recommendations are as follows:

- check convergence under the different models

- give preference to models that fits better with the organism under study

    - *a priori*: in the sense of prior knowledege about dispersal, potential barriers to gene flow...
    - *a posteriori*: in the sense where estimated K and maps complies best with what is known about the organism

- do not attempt to compare different models on the basis of the average posterior probability. Indeed, they are defined on different parameter spaces and such a comparison do not make sense mathematically.

6.3 Objective criterions to perform model selection [TODO]

7   More examples using the R command-line

7.1   Estimating frequency of null alleles

If the option for filtering null alleles is chosen for simulation with function MCMC, the estimated frequency of null alleles at each locus can be obtianed through function `EstimateFreqNA` as follows:

```
EstimateFreqNA(genotypes=geno,path.mcmc="./")
```

which returns a vector of length `nloc` (the number of loci) whose entries are estimated frequencies of null alleles.

7.2   Analysing georeferenced data with a non-spatial prior

A dataset consisting of georeferenced genetic data can be analyzed with a non-spatial prior for population membership. The coordinates are not used in the inference algorithm, they are just used for the graphical representations.

Similarly to the example of R code given in section 5.2, this can be done by:

```
MCMC(coordinates=coord,
     genotypes=geno,
     varnpop=TRUE,
     npopmax=10,
     spatial=FALSE,          ## the argument spatial is now set to FALSE
     freq.model="Correlated"
     nit=100000,
     thinning=100
     path.mcmc="./")
```

7.3   Analysing non spatial data

It is also possible to analyse datasets consisting only of genotypes (no spatial coordinates). This can be done by a call of function MCMC as:

```
nindiv <- nrow(genotypes)
dummy.coord <- matrix(nr=nindiv,ncol=2,runif(nindiv*2)) ## create dummy coordinates

MCMC(coordinates=dummy.coord,
     genotypes=geno,
     varnpop=TRUE,
     npopmax=10,
     spatial=FALSE,          ## the argument spatial set to FALSE
     freq.model="Correlated"
     nit=100000,
     thinning=100
     path.mcmc="./")
```

This allows to estimate the number HWLE populations and population memberships of individuals. Spatial graphical displays do not make sense in this context.

7.4 Getting improved graphics

You may find that the graphical features availbale through the GUI and the Geneland graphical functions are too limited. The possibilites to improve graphics by working directly though the command-line are almost unlimited.

7.4.1 Superimposing countries boundaries and coast lines on a Geneland map?

Let us assume that you have a map giving population memberships or posterior probabilities of population memberships over a given domain. You can superimpose countries boundaries and coast line by typing:

```
map(resolution=0,add=TRUE)
```

This will add the desired lines to the active graphic window (by default the last window opened). The active graphic window can be redefined by e.g. `dev.set(3)` (this will set window # 3 as active window). See on-line help of function `map` for details (`? map`).

7.5 Using MCMC outputs to better check convergence [TODO]

7.6 Interpretation of posterior probabilities of population memberships [TODO]

8 Simulation of data under the spatially organised HWLE populations model

Datasets can be simulated under the statistical models described in section 2. This can be done via the R shell by calling function simFmodel as follows:

```
simdata <-  simFmodel(nindiv=100,
                      coord.lim=c(0,1,0,1),       ## simulations on the unit square
                      number.nuclei=15,           ## tessellation driven by 15 polygons
                      nall=rep(10,20),            ## 20 loci with 10 alleles each
                      npop=3,                     ## 3 populations
                      freq.model="Correlated",    ## Correlated frequency model
                      drift=rep(0.04,3),          ## drift (or Fst) parameters
                      dominance="Codominant"      ## codominant-like genotypes
                      )                           ## (two columns per locus)
```

Some of the arguments are optional. See on-line help (`?  simFmodel`) for details.

The R object `simdata` is a list whose components can be checked by `summary(simdata)`. The genotypes are stored as `simdata$genotypes` and the coordinates as `simdata$coordinates` and are suitable for simulations studies, e.g. to assess the effect of number of loci, number of individuals, allele diversity ... on accuracy of inferences.

9 Simulation of data with spatially auto-correlated allele frequencies [TODO]

10   Using other softwares to analyse Geneland outputs

10.1   Population genetics softwares

10.1.1   Genepop

There is a Geneland function called `gl2gp` that writes coordinates and genotypes into an ascii file suitable for analysis with the GENEPOP program [Rousset, 2007]. See on-line help (`? gl2gp`) for details. The genotype file produced might require some extra hand editing.

10.2   MCMC post-processing softwares

10.2.1   Partitionview

`Partittionview` can be used to visualize the posterior probability distribution of the partition of individuals. The information required as input for this program is in the file `proba.pop.membership.indiv.txt` stored in the Geneland output directory. See section C.2.3.

10.2.2   Distruct

`Distruct` can be used to visualize the distribution of individual population membership. The information required as input for this program is in the file `proba.pop.membership.indiv.txt` stored in the Geneland output directory. See section C.2.3. The information displayed is similar to what is obtained with the Geneland function `PlotTessellation` disregarding the spatial aspect of the dataset.

10.3   Geographical information systems (GIS)

Output of Geneland can be combined with high quality maps obtained with the Geographic Resources Analysis Support System, commonly referred to as GRASS. See `http://grass.osgeo.org` and `http://grass.osgeo.org/wiki/Main_Page`. Some of the tasks can be performed directly under R through the R package spgrass6. See `cran.r-project.org/web/packages/spgrass6`.

11   Using Geneland to analyse other software outputs [TODO]

A Frequently asked questions

A.1 Can I use population data?

It is possible to analyse datasets where different individuals share the same spatial location. If such data are treated without *uncertainty on coordinate* in the GUI or `delta.coord` is set to 0 in function `MCMC` then individuals sharing the same coordinates will be assigned to the same inferred group.

A.2 Can I use SNPs?

It is possible to run Geneland with SNPs. The bases have to be recoded as if they were alleles (each base being coded by an integer in $\{1, 2, 3, 4\}$). Fixed alleles are not allowed. Loci carrying such alleles have to be removed from the dataset first.

A.3 Can I use sequence data?

It is possible to run Geneland with sequence data. The bases have to be recoded as if they were alleles (each base being coded by an integer in $\{1, 2, 3, 4\}$). See also section A.2.

A.4 Can I use haploid data?

Yes. Specific computing options are implemented in Geneland for haploid data.

A.5 Can I study organisms other than haploid and diploid?

No. The only ploidy handled as of today are haploid and diploid.

A.6 Can I use dominant markers?

Yes although this feature is not published nor documented yet.

A.7 Is there any way to account for the presence of special landscape features?

We have been often asked how to include information about the presence of ”an urban area not suitable for my organisms” or ”a land mass in the midle of sea water obviously not suitable for fish”. There is currently no obvious way to do that with Geneland. We recommend to analyse your data ignoring the presence of such lanscape feature and to try to take it into account at the post-processing stage.

A.8 Can I study an organism leaving in a linear habitat?

Yes. The neat way to do that in Geneland is to convert 2-dimensionnal coordinates into 1-dimensionnal curvilinear coordinates measuring distance of each individuals to an arbitrary origin in this habitat.

A.9 How should I choose $K_{\max}$?

Take it larger that the largest value that you can reasonnably expect for your dataset. A value $K_{\max}$ taken much larger than the true K (if any) does not make any problem but the larger $K_{\max}$ is the slower the computations. The value $K_{\max}$ is too small if the chain simulating the values of K get stuck at the maximum value. In such a case, MCMC have to be made with a larger value for $K_{\max}$.

A.10 Which value should I choose for the number of MCMC iterations?

There is no obvious answer to that. The value should be large enough to avoid any of the symptoms of lack of convergence described in section 5.5.2. A rough order of magnitude would be $n_{it} = 100000$ for a dataset of $n = 100 - 300$ individuals at $L = 10 - 30$ loci.

A.11 Which value should I choose for the thinning?

The thinning is defined as the proportion of MCMC iterations saved on the disk. This computing option has a limited effect on the accuracy of inferences. It is more a matter of how many disk space is available. For a run of $n_{it} = 100000$ iterations, we typically save 1000 iterations and set the thinning to 100.

A.12 I have launched 50 runs of 5000000 iterations with a thinning of 1 and my disk is full!!

You don't need to save each single iteration. MCMC produce correlated samples. Saving one iteration out of 100 (`thinning=100`) is usually enough. The thinning can be increased even more as long as the number of iterations saved (`nit/thinning`) remain large ($100 - 1000$). Remember that the amount of disk space required to store results of MCMC iterations increases approximately linearly with the number iterations.

B Algorithm

B.1 Simulation based inference [TODO]

B.1.1 Special aspects

K_{init} set to K_{max} during the first iterations

B.2 Post-processing MCMC outputs [TODO]

B.2.1 Estimating K

B.2.2 Dealing with label switching

B.2.3 Computing posterior probability of population memberships

B.3 F statistics [TODO]

- Estimation according to Weir and Cockerham [1984]

- Missing data allowed. Induce a small bias in the estimation of F_{ST} and a somehow larger bias in the estimation of F_{IS} .

- Not implemented for haploid data

C Description of MCMC output files

C.1 Files produced by MCMC simulations

C.1.1 parameters.txt

List of characteristics of the dataset and all arguments passed to function *MCMC* (via the GUI or directly).

C.1.2 populations.numbers.txt

Simulated values of number of populations K.

C.1.3 nuclei.numbers.txt

Simulated values of number of Voronoi cell m coding for population membership. (Set to n under the non-spatial option).

C.1.4 coord.nuclei.txt

Simulated coordinates of the nuclei of Voronoi cells in the tessellation uses to parameterize population membership. (Set to the coordinates sof sampled individuals under the non-spatial option).

C.1.5 color.nuclei.txt

Simulated population membership of Voronoi cells. It is coded as an integer and displayed as a color.

C.1.6 `ancestral.frequencies.txt`

Simulated allele frequencies of ancestral population in the correlated allele frequencies model.

C.1.7 `drifts.txt`

Simulated drift parameter (or F_{ST}) in the correlated allele frequencies model.

C.1.8 `frequencies.txt`

Simulated allele frequencies of present time populations in the correlated allele frequencies model.

C.1.9 `log.likelihood.txt`

Log-likelihood along MCMC simulation.

C.1.10 `log.posterior.density.txt`

Log of posterior density of simulated parameters along MCMC simulation.

C.2 Files produced when post-processing MCMC simulations

C.2.1 `postprocess.parameters.txt`

List of all arguments passed to function *PostProcessChain* (via the GUI or directly).

C.2.2 `proba.pop.membership.txt`

Posterior probability of population membership for pixels of a discretization of the domain.

C.2.3 `proba.pop.membership.indiv.txt`

Posterior probability of population membership for sampled individuals.

C.2.4 `modal.pop.txt`

Estimated population membership for pixels of a discretization of the domain.

C.2.5 `modal.pop.indiv.txt`

Estimated population membership for sampled individuals.

C.2.6 `perm.txt`

Permutation of population labels allowing to get rid of the label switching issue.

References

D. F. Callen, A.D. Thompson, Y. Shen, H. A. Phillips, R. I. Richards, and J. C. Mulley. Incidence and origin of 'null' alleles in the (ac)n microsatellite markers. *American Journal of Human Genetics*, 52:922–927, 1993.

A. Coulon, G. Guillot, J.F. Cosson, J.M.A. Angibault, S. Aulagnier, B. Cargnelutti, M. Galan, and A.J.M Hewison. Genetics structure is influenced by lansdcape features. Empirical evidence from a roe deer population. *Molecular Ecology*, 15:1669–1679, 2006.

E.E Dakin and J.C. Avise. Microsatellite null alleles in parentage analysis. *Heredity*, 93(5):504–509, 2004.

M. Foll, M.A. Beaumont, and O. Gaggiotti. An approximate Bayesian computation approach to overcome biases that arise when using AFLP markers to study population structure. *Genetics*, 179:927–939, 2008.

M. Fontaine, S.J.E. Baird, S. Piry, N. Ray, K. Tolley, S. Duke, A. Birkun, M. Ferreira, T. Jauniaux, A. Llavona, B. Östürk, A.A. Östürk, V. Ridoux, E. Rogan, M. Sequeira, U. Siebert, G.A. Vikingson, J.M. Bouquegneau, and J.R. Michaux. Rise of oceanographic barriers in continuous populations of a cetacean: the genetic structure of harbour porpoises in old world waters. *BMC Biology*, 5(30), 2007.

G. Guillot. Inference of structure in subdivided populations at low levels of genetic differentiation.The correlated allele frequencies model revisited. *Bionformatics*, 2008. doi: 10.1093/bioinformatics/btn419.

G. Guillot, A. Estoup, F. Mortier, and J.F. Cosson. A spatial statistical model for landscape genetics. *Genetics*, 170(3):1261–1280, 2005a.

G. Guillot, F. Mortier, and A. Estoup. Geneland: A computer package for landscape genetics. *Molecular Ecology Notes*, 5(3):708–711, 2005b.

G. Guillot, F. Santos, and A. Estoup. Analysing georeferenced population genetics data with Geneland: a new algorithm to deal with null alleles and a friendly graphical user interface. *Bioinformatics*, 24(11):1406–1407, 2008.

U. Hannelius, E. Salmela, T. Lappalainen, G. Guillot, C.M. Lindgren, U. von Döbeln, P. Lahermo, and J. Kere. Population substructure in Finland and Sweden revealed by a small number of unlinked autosomal SNPs. *Under revision*, 2008.

G. Nicholson, A.V. Smith, F. Jónsson, Ó. Gústafsson, K. Stefánsson, and P. Donnelly. Assessing population differentiation and isolation from single-nucleotide polymorphism data. *Journal of the Royal Statistical Society, series B*, 64(4):695–715, 2002.

D. Paetkau, W. Calvert, I. Stirling, and C. Strobeck. Microsatellite analysis of population structure in canadian polar bears. *Molecular Ecology*, 4:347–354, 1995.

E. Paradis. R for beginers. `http://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf`, 2005.

E. Paradis. *Analysis of Phylogenetics and Evolution with R*. Springer Verlag, 2006.

F. Pompanon, A. Bonin, E. Bellemain, and P. Taberlet. Genotyping errors: causes, consequences and solutions. *Nature Review Genetics*, 6(11):847–859, 2005.

F. Rousset. Genepop'007: a complete re-implementation of the Genepop software for windows and linux. *Molecular Ecology Notes*, 8(1):103–106, 2007.

B.S. Weir and C.C. Cockerham. Estimating F-statistics for the analysis of population structure. *Evolution*, 38(6):1358–1370, 1984.

Index