WARNING:

This document is not the final one for gMetapop upcoming release 1.0.0
The complete version will be uploaded soon on the website.

gMetapop User Manual

version 1.0.0

October, 2022

Authors

Garnier-Géré Pauline (pauline.garnier-gere@inrae.fr)

Austerlitz Frédéric (austerlitz@mnhn.fr)

Mariette Stéphanie (stephanie.mariette@inrae.fr)

Raspail Frédéric (Frederic.Raspail@inrae.fr).

Website

Download and more information at the gMetapop website: https://github.com/gMetapop/gMetapop/tree/master/3-User.Manual-ver.1.0.0

gMetapop User Manual contents

5. CHAPTER 5 Evolutionary scenarios: testing gMetapop by comparing simulated results with theory or published results.

1. CHAPTER 1 Introduction and overview

1.1 Scope and aim of the simulation program with an interface

gMetapop is a flexible forward-in-time and individual-based simulation program, which allows one to model the evolution of diversity at genes and quantitative traits in subdivided populations of species with complex demographic life cycles. It is an individual-based stochastic simulation program, in which individuals can be characterized by their multilocus genotype and optionally by their phenotype at a given trait coded by quantitative trait loci (QTL). The software allows to simulate different types of genes (neutral or under selection), populations connected by gene flow, either via zygote migration (e.g. seeds migration) or male gamete migration (e.g. pollen flow). It also permits demographic models with overlapping generations (for example, perennial species such as trees or animals with an age class structure), and mating systems including clonal reproduction and/or selfing. One can apply selection within and among populations with different intensities, either on a phenotypic trait determined by theoretical allele additive values, or directly by assigning fitness values to each genotype. The total number of loci, individuals, populations and demographic or reproductive events during simulations is only limited by the users' hardware and memory capacities. As explained hereafter, we took a special care to provide help to the users for preparing the configuration of simulations, for launching them in batch mode, and for easily processing output files with the R software (www.r-project.org).

One main benefit of this modelling tool is its user-friendly, flexible and interactive Graphical User Interface (**GUI**). We developped the GUI to replace the cumbersome process of building the large configuration parameter files needed for elaborate evolutionary scenarios. The GUI thus allows the users to focus on their biological model. It also makes the program a great learning tool for basic evolutionary processes in population or quantitative genetics practical courses. This tool also permits, in a very efficient way, to get a graphical overview of the possible outcomes of a range of demographic or selective scenarios, and thus allows refining parameter values. To help achieve this goal, we designed several contrasted and detailed examples of simulation scenarios that serve both as scientific tests of particular predictions in evolutionary genetics, and as tutorials. These tests also give a more intuitive and efficient "getting started" with the software. The short **Chapter 1** and **2** give an overview of gMetapop installation, features and how to get help. **Chapter 3** describes GUI Tabs that allow choosing genetic features and parameter values for evolutionary processes in simulation scenarios. **Chapter 4** details input files needed and output files produced by the software. Chapters **3** and **4** can be skipped for users familiar with population genetics simulation softwares, thus one can go to **Chapter 5** to follow or redo a couple of tutorials, and/or start directly from the GUI. *Html* links throughout **Chapter 5** allow getting more information on processes and parameter definitions from previous chapters if needed.

The second principal benefit of gMetapop is that it allows simulating a range of complex demographic scenarios with overlapping generations or reproductive events (with or without selection) in a very straightforward manner. This is possible by simply choosing this type of scenario in the initial conditions of a configuration setup (see **Figure 3** and **3.1**), and then by further describing demographic parameter values across age or development classes with the help of the specifically designed GUI tools for that purpose (*Demography Tab* **3.5.3;** see also *3.5.5.3* to *3.5.5.7* for examples of the flexibility of the model). To our knowledge, these features are not available in recently published forward-in-time simulation tools, which propose nethertheless many other levels of complexity in evolutionary scenarios (e.g. Metapop, Soularue *et al.* 2019; QuantiNemo 2.0, Neuenschwander *et al*. 2018). Two recent flexible programs also encompass a large range of demographic and genetic processes and models, including some with age or class structure: SLIM3 (Haller and Messer 2019) and

CDMETAPOP (Landguth *et al*. 2017, 2019). SLIM3 greatly expands the SLIM original Wright-Fisher simulation framework (Messer 2013), in order to allow for more realistic scenarios (Haller and Messer 2019). However, such complex scenarios still require the user to develop their own scripts with the associated scripting language provided (*Eidos*, Haller 2018). CDMETAPOP is a spatially explicit, individual-based eco-genetic model of meta-population dynamics, initially developed in the particular biological context and demography of marine and river species. Although in theory applicable to many other biological models, an important amount of input data and matrices of various demographic and spatial parameters values is necessary to set up and run simulations (CDPOP, Landguth and Cushman 2010, Day et al. 2018). It appears thus that both SLIM3 and CDMETAPOP modelling frameworks would require a significant investment in time and training to master simulations with complex features such as overlapping generations with an age class structure, without the insurance that they would be suitable enough for the questions addressed by the users. In contrast, gMetapop provides an intuitive GUI, and step-by-step tutorials that allow the user to efficiently pick up different ways to model complex demographic settings, thus greatly reducing the time needed for this "getting started" phase. Additionally, the demographic model with an age class structure and overlapping generations that is developed in gMetapop offer a large flexibility of particular conditions (see *3.5.5.3* to *3.5.5.7*).

The gMetapop software is thus constituted by two main components (Raspail *et al.* 2022, **Figure 1.4**):

- **gMetapop_GUI**, the interface program which guides the user for designing simulations, launching them, and getting results graphical overview.
- **gMetapop_CORE**, the program that takes all these files as input to perform the simulations. gMetapop_CORE stems from earlier programs that included separate and complementary features (Le Corre *et al.* 1997; Austerlitz *et al*. 2000; Machon *et al*. 2003; Austerlitz and Garnier-Géré 2003; Le Corre and Kremer 2003). The first task of the gMetapop project was to combine these programs.

Overall, we have developed, nevertheless, an original tool, with ~90% of the whole program code that is new (72%), or has been largely modified and extended (18%).

1.2 Technical features – Operating systems

gMetapop _GUI is developed in C++ with Qt libraries. Qt was chosen because it is free and compatible with various systems, the Qt project being under open-source governance (https://www.qt.io). gMetapop_CORE has been developed in C and increasingly in C++ for recent modules. Both programs (GUI and CORE) can be compiled and run independently. However, it might be of interest for developers to know that they share some code in common, for example for verifying steps of initial file formats for allele frequency and genotypes. Executables are available for both Linux (Ubuntu distributions) and Windows Operating Systems (OS).

NB.1.2-a: It is possible to run simulations in one OS with *.xml or text files files that have been created in another. We just suggest to run simple checks for text format compatibility between OS (by running the dos2unix tool under Linux on text files for example). See **Chapter 4** for an exhaustive list of those files and how to manage them when running simulations.

NB.1.2-b: In principle, the GUI parameter files (*param* *.xml) created by the current release of gMetapop_GUI will be at least partially re-usable in future GUI versions if the main structure of the different Tabs is conserved. However, it is still recommended to check that parameter values have been loaded correctly.

1.3 Access and License

gMetapop executables and source code are freely available at https://github.com/gMetapop. For both gMetapop_GUI and gMetapop_CORE programs, they are distributed under the terms of version 3 of the GNU General Public license (GPL at https://www.gnu.org/licenses/gpl-3.0.html).

1.4 gMetapop: how does it work? Overview and main features

gMetapop_GUI launches a graphical interface that allows the user to set up all the initial conditions and parameter values needed to simulate evolutionary scenarios (See **Figure 1.4**). Some of those conditions and values can also be loaded directly into the GUI from user-defined files, or using paths that link to these files (e.g. allele frequencies, genotype files, migration rates, etc…). All this information is then saved in a *.xml parameter file, named *param* in the rest of the document. From the *Run Tab* of the GUI, the user finally chooses the working folder, and is guided to create or select all the files needed by gMetapop_CORE to launch the simulations (see **part 3.8** for detailed steps):

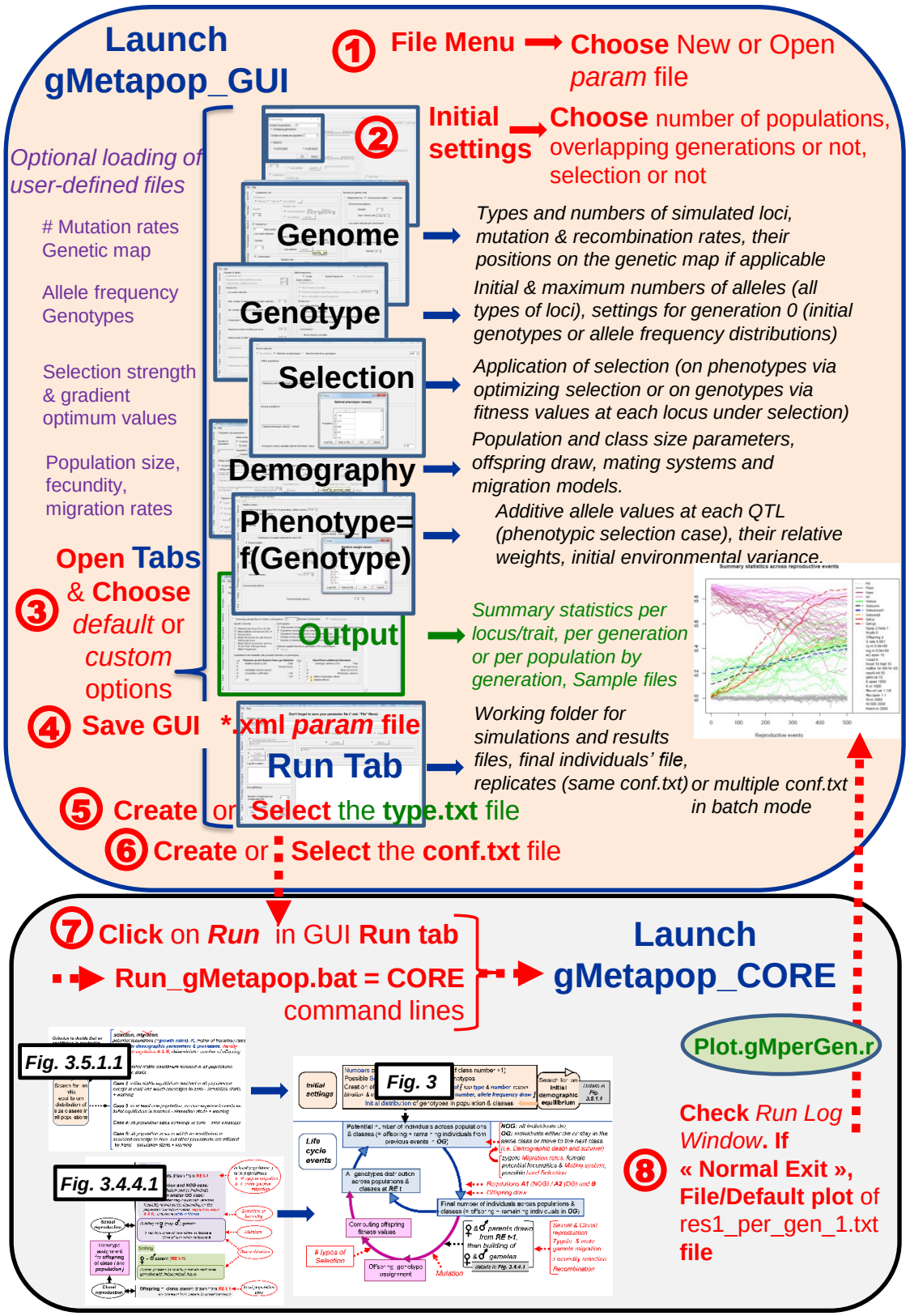
- the configuration file (default name **conf.txt**), which creation also generates the initial allele frequency file, unless an allele frequency or a genotype file was previously loaded,
- the “type of results” file (default name **type.txt**) that includes the information on all requested summary statistics,
- a file with command lines (*.bat/*.sh) needed to launch gMetapop_CORE either from the GUI, or directly from a Linux terminal or a Windows shell (see *3.8.7.3*).

All files are located in the working folder.

The GUI was designed so that a large range of scenarios is already available without loading any particular files. Moreover, the GUI serves to create the correct formats for initial frequency or genotype files needed for simulation scenarios. **These two main features allow saving a considerable amount of time in preparing simulation runs, and largely increase the range of possible scenarios available in an efficient way.** We provide step-by-step tutorials in **Chapter 5** for contrasted scenarios and for performing post-treatment analyses, all tutorials being available **online**. Many user-defined parameter values of simple formats can also be loaded into the GUI, which increases the range of possible scenarios, in addition to those proposed in the default options (see also **NB.3.2-a** for more possibilities by editing the conf.txt file).

Additionally, proposed default plots of output files allow obtaining a graphical overview of simulated scenarios with summary statistics. These plots can be used first to better adjust initial conditions, and second to better understand and interpret simulation results. The provided R functions are easily modifiable by users, and can be re-used for custom plots directly from the GUI (see **3.9** for more information on plots from the GUI). Examples of custom scripts are provided in most tutorials to further explore outputs from simulations, for example in tests **5.1**, **5.4**, **5.5**, **5.6**.

Figure 1.4 Main steps for running a simulation, main Tabs description, input and output or associated files (see more details in **Figure 3** for the typical life-cycle and processes modelled in this program, and ***Figure 3.4.4.1*** and ***Figure 3.5.1.1*** for more details on specific parts of this life-cycle).



2. CHAPTER 2 How to get started?

2.1 Installing and starting gMetapop_GUI

gMetapop is available as standalone executables for Linux and Windows OS with all files needed without any installation required. Simply download the compressed files corresponding to your OS (either **Windows 64 bits=Win64**, or **Linux 64 bits (Linux64)**) at the gMetapop website, and extract them to a folder of your choice:

Under a 64-bit Windows OS, simply click on the gMetapop_GUI_Win64.exe for launching gMetapop_GUI, from which gMetapop_CORE_Win64.exe can be launched.

Under a 64-bit Linux OS, open a Terminal window, unzip the linux version of the program then check that both GUI (i.e. gMetapop_GUI_Linux64) and CORE (i.e. gMetapop_CORE_Linux64) *.exe, *.sh and associated library (i.e. lib*) files have executable rights, type chmod +x *filename* to allow for executable rights to ***.sh** or ***.exe** files. Since the GUI is developed in C++ with open-source Qt libraries (https://www.qt.io), those are provided in the current distribution, but additional libraries might be needed if they are not already available from your OS, so please type first (once only):

sudo apt-get install libxcb-xinerama0

Then in the folder where the files have been unzipped, the GUI application is launched by typing:

sh run_gM_GUI_Linux64.sh in a terminal window.

The gMetapop_CORE source programs can also be compiled with standard C/C++ compilers (e.g. g++ under Linux, MinGW under Windows).

In case of installation problems, in particular for gMetapop_GUI under some Linux OS, please contact Frédéric Raspail (frederic.raspail@inrae.fr) or Pauline Garnier-Géré (pauline.garnier-gere@inrae.fr).

NB.2.1*:* If needed, we can also provide *.exe for a 32-bit Windows OS.

2.2 gMetapop_CORE for Linux OS

Linux users who would like to directly use gMetapop_CORE can either work with the compiled version provided on the website, or compile the program in their own Linux distribution if needed (source code available here). If you encounter any problems, please contact Frédéric Raspail at frederic.raspail@inrae.fr for more details on the compilation procedure.

2.3 Launching gMetapop_CORE simulations

Simulations can be run either directly from the GUI or in command line mode in both Linux and Windows OS (see **Figure 1.4** above and **3.8.7** below for more details).

2.4 How to get help?

Various sources of help are available to users:

- *Default parameter values* are proposed and automatically loaded in the GUI when making a new *param* file (see also **4.2**). We chose them so that they are plausible in scenarios corresponding to non-extreme biological conditions, based also on our own experience across different biological models. Therefore, these default values can be kept as starting points for exploratory tests, allowing the users to only focus at the particular factors or parameters they are interested in. However, before or when analyzing the results, we recommend that they carefully check if those default values

are compatible with the organisms that they have in mind or the scenarios they want to simulate.

- *Examples of *.xml param* files are given online at the gMetapop website **here** for all tests described in details in **Chapter 5**. Some of these *param* files might be better starting points than default parameters in some cases. They are available for loading into the GUI and can be further changed.
- *Tooltips* appear on screen when moving the “mouse” on the each sub-window/group-box part of the GUI in a Tab. Tooltips are small texts added to help the user directly from the GUI, in order to get a rapid and intuitive appraisal of particular aspects of the modelled processes. These tooltips provide definitions and main uses of the corresponding GUI options, and usually correspond in this Manual to the first sentences describing the GUI options in **Chapter 3** below.
- Finally, more explanations can be found in the present *User Manual* on different GUI features and processes modelled in gMetapop_CORE (**Chapter 3)**, on all input and output files (**Chapter 4**), and on building *param* configuration files with detailed step-by-step procedures (**Chapter 5**).

2.5 Input and Output files

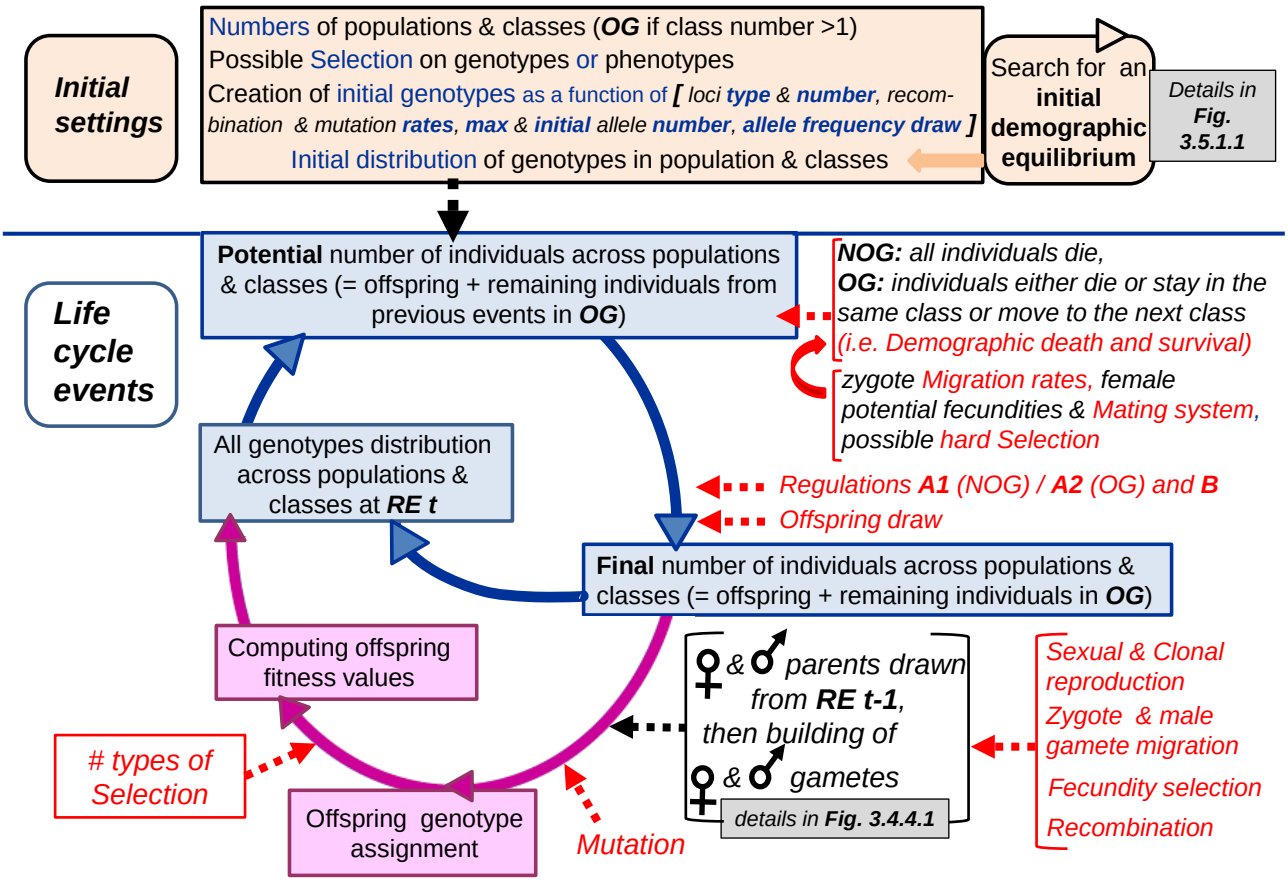
Input files are those created by the GUI, which are needed to run simulations. These files are based on the chosen initial conditions and the provided default parameter values. For most parameters, the user can also load text **input files** into the GUI, in order to fit particular scenarios and biological model characteristics. This is easily done since the GUI has been designed to avoid the burden of struggling with initial format definitions. First, as a general principle, **formats of these input files** can be produced for any configuration **in preliminary runs, by saving or requesting them as output files of the same type in the GUI** (see **Table 4.4.1**). These files will have the correct input format, or a very similar one, which can be used as an example. Second, these files can be edited while keeping the same structure, and loaded back into the GUI for particular scenarios. Examples of such files include allele frequency and genotype files, or parameter value files, with a few exceptions. See **4.2** and **Table 4.2** in **Chapter 4** for more information on input files, and many examples from the tutorials to download at the gMetapop website.

Output files are either allele frequency, genotype, or result files that include the summary statistics requested from the *Output Tab* (see **4.4**) or directly from the type.txt file when you are in command line mode (*3.8.7.4*). See also **3.7** for more on creating the type.txt file, and other features defined in the *Run Tab* (**3.8**).

3. CHAPTER 3 Building the *param* *.xml file with gMetapop_GUI and modelling demographic and selection processes

According to the scenarios simulated, evolutionary processes will act at different stages during the life cycle of individuals (see **Figure 3** for a generic life cycle as modelled in gMetapop).

Figure 3: Typical life cycle in gMetapop with possible overlapping generations. Density-dependent regulations **A1**, **A2** and **B**, for population and class sizes, are further described in *3.5.1.2*, *3.5.5.4* and *3.5.5.5*. The potential number of offspring (and of possible remaining individuals from previous events) is a function of (i) the number of fertile individuals across populations and classes and their potential fecundities, (ii) their class demographic survival and transition rates, (iii) the number of potential migrants, and (iv) the mean fitness of each class by population in the case of hard selection. The main cases when searching for an initial demographic equilibrium or initial demographic conditions are provided in **Figure 3.5.1.1**, and more information for algorithms leading to the attribution of diploid genotypes are in **Figure 3.4.4.1**. Parameter values chosen by the user are indicated **in blue** in the *Initial settings* box, and stochastic evolutionary processes thoughout *life cycle events* are **in red**. **Blue arrows** in the main cycle of events correspond to the demographic part (individuals' numbers), and **purple arrows** to the genetic part (genotypes' assignment). ***OG***: overlapping generations; ***NOG***: non-***OG***; ***RE t***: set of reproductive events at generation or time ***t.***



This chapter describes GUI Tabs, and is structured according to titles of each Tab main window, sub-windows within Tabs (also called *group-box*), and different *buttons*, *check-boxes* or *radio-buttons* of the GUI. These titles, unless self-explanatory, are further defined by **tooltips** that appear while moving the computer mouse on each graphical object (see **Figures 3.2, 3.3** and **3.8** for tooltip examples). For each Tab that represents particular genetic features, stages or evolutionary processes, additional information is provided on the

underlying models, and on events to which individuals can be submitted throughout their life cycle (**Figure 3** above, and see parts **3.2.4**, **3.3.4**, **3.4.4**, **3.5.5**, **3.6.6**, **3.7.6**, **3.10** in this chapter below). Diploid hermaphrodite individuals are simulated in gMetapop, and they can undergo:

a) either one basic life cycle starting with their creation (i.e. as adults initiated at the start of a cycle, or as offspring from the mating between fertile individuals) until their first reproductive event as fertile individuals when they all die (case of non-overlapping generations),

b) or more than one cycle in the case of overlapping generations, where they belong to different age classes which can reproduce or not, depending on their demographic parameter values.

Due to the many possible interactions among stochastic processes and steps in the demographic model to which individuals can be submitted throughout this typical life cycle, many scenarios are possible (see parts *3.5.5.3* to *3.5.5.7* for illustrations)

3.0 GUI File menu

File/New… allows the creation of a new *param* *.xml file.

File/Open… allows loading a *param* file which was created before.

NB.3.0: When loading a *param* file that was previously saved, most of the text files with parameter values are already included, except those corresponding to allele and genotype frequencies, mutations rates or genetic maps (see **Chapter 4** for more on input files). For these files, only their paths are saved. The user thus needs to check and update these paths by simply reloading those files from its own folder organization. Warnings or Error messages are issued if correct paths are not provided (see more details in **NB.5.1-c** and throughout the Tutorials of **Chapter 5**).

The **File/Close Interface parameter file** option of the GUI menu allows closing a currently open *param* file.

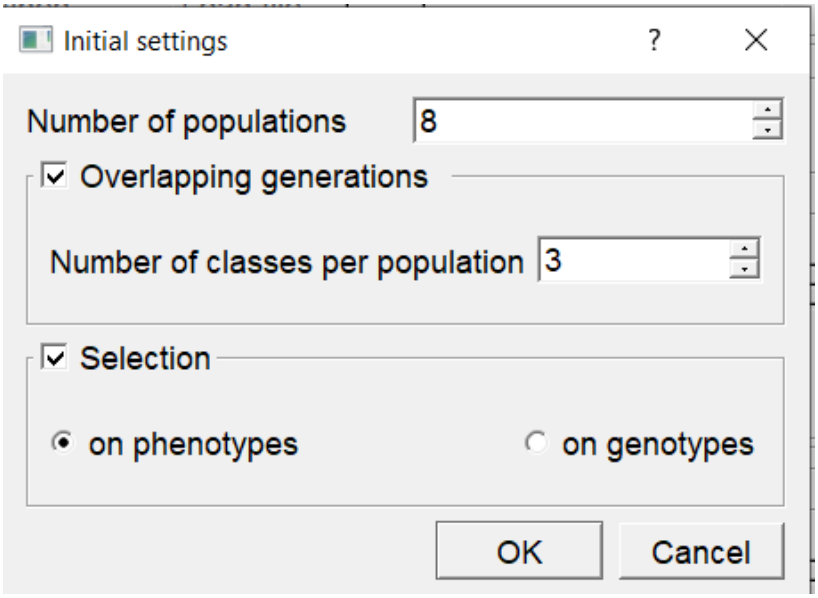
The **File/Save Interface parameter file** option allows saving a *param* file in the chosen folder.

For the options **File/Choose R path, File/Default plot, File/Custom plot** of the menu, see **3.9** below.

3.1 Initial settings…

…opens when choosing **File/New Interface parameter file** in the GUI menu (see **Figure 3.1).**

Figure 3.1 Example of Initial settings window.



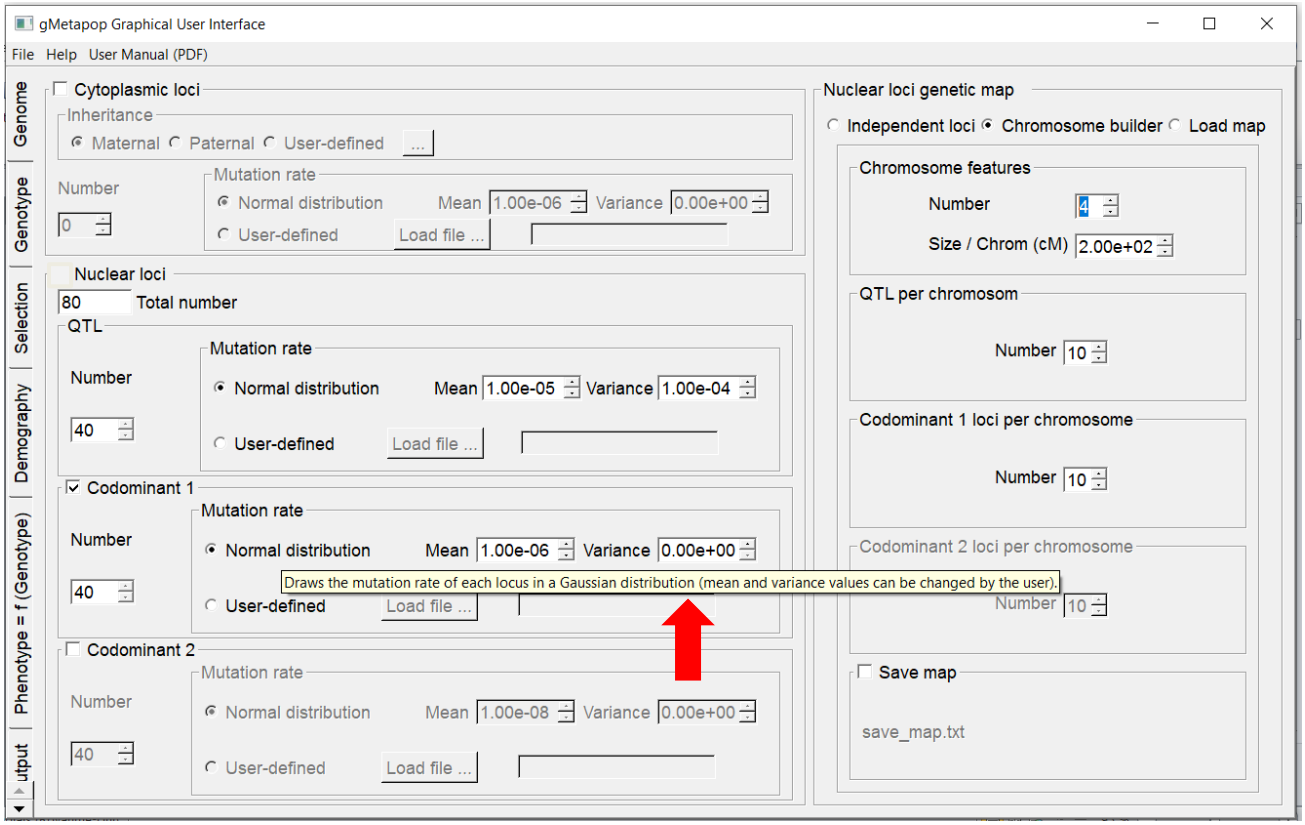
This window sets initial conditions that cannot be modified afterwards in the different GUI Tabs, unless a new *param* is created. These initial conditions are: the values for the maximum numbers of populations and classes, whether selection is **potentially** present or not on **at least one** locus, and whether it occurs on **phenotypes or genotypes**. Apart from these conditions, this window triggers the loading of many other default parameter values that *can* be modified afterwards across the GUI.

Number of classes per population: default is "1" (non-overlapping generations where all individuals die after reproduction). For organisms which can be modelled with overlapping generations or reproductive events, a class corresponds to a group of individuals sharing for example similar size, age or developmental stage (see **3.5 Demography Tab**).

3.2 Genome Tab

This Tab describes different types and numbers of simulated loci, their mutation and recombination rates, and their positions on a genetic map created by default, or one that is loaded by the user. Genomes of simulated individuals are set up as linear successions of such loci, possibly organized into chromosomes (see *3.2.3.1* and **Figure 3.2** below). Since different groups of loci (some neutral with possible different mutation rate ranges, and some under selection, see also **NB.5.2-a**) can be placed on the same genetic map, setting up varying recombination rates between them allows processes such as hitch-hiking to be studied, and statistics such as linkage disequilibrium to be computed from output genotype files (see **3.8.3 in the Run Tab**).

Figure 3.2: ***Genome Tab*** example with the use of the "**Chromosome builder**" option and a tooltip example (red arrow), in a scenario with selection on phenotypes (see **3.4**).



3.2.1 Cytoplasmic loci and mutation rates

Inheritance: sets the inheritance of cytoplasmic loci.

Maternal: all loci are maternally inherited.

Paternal: all loci are paternally inherited.

User-defined: proportion of cases in which each offspring inherits the paternal allele at each locus, either set by the user or loaded from a file. A value of "0" is equivalent to strict maternal inheritance, a value of "1" to strict paternal inheritance.

Mutation rate/normal distribution: draws the mutation rate of each locus in a Gaussian distribution (default mean and variance values can be changed by the user).

Mutation rate/User-defined: different rate values can be loaded from a file (see format examples in **Chapter 4** and online here, and see also **NB.3.2-a**).

3.2.2 Nuclear loci and mutation rates

Four classes of nuclear loci are allowed for more flexibility in building *param* files directly at the GUI level (see also *3.2.4.1*):

QTL: quantitative trait locus (or loci) in case of selection on phenotypes and their number (maximum number: 9999).

Loci under selection: in case of selection on genotypes and their number.

Codominant 1: group of neutral loci sharing the same mutation rate and variance (maximum number: 9999).

Codominant 2: another group of neutral loci possibly sharing different mutation characteristics (maximum number: 9999).

Mutation rate/normal distribution: draws the mutation rate of each locus in a Gaussian distribution (mean and variance values can be changed by the user, see **NB.3.2-a** below).

Mutation rate/User-defined: different values can be loaded from a file (see format examples in **4.2**, or at the website here**).**

NB.3.2-a: In addition to the different options available in the GUI above, any value of mutation rate can be chosen across loci (either cytoplasmic or nuclear) by editing them directly in the corresponding lines of the conf.txt file. This possibility exists for any parameter line of the conf.txt, as long as the original format and structure of the line remain unchanged. **Error messages will be generated the conf.txt format is changed**. Ideally, these modifications should be made with a script (see such an example in test **5.3** where migration rates are changed in batch across multiple conf.txt files), and more information is provided on possible problems encountered with editing the **conf.txt** file in the online Appendix (search for "edit").

3.2.3 Nuclear loci genetic map and recombination rates

Nuclear loci are located on a genetic map. They can either be all independent (physically unlinked), or organized randomly on chromosomes using the "**Chromosome builder**" option, or positioned according to a user-defined genetic map that can be loaded into the GUI as a simple text file, using a format compatible with the conf.txt file (see below).

Save map: prepares the path for saving the genetic map file in the chosen folder (see Load map in *3.2.3.2*.

3.2.3.1 Chromosome builder

Size/Chrom (cM): Size per chromosome in centiMorgan. Pairwise recombination rates among loci are drawn in a uniform distribution, and they are standardized according to the total chromosome size provided in the GUI divided by the sum of all drawn values, while using an upper bound value of 0.5.

QTL/Loci under selection per chromosome: the number of loci potentially under selection, which positions are randomly drawn on each chromosome.

Codominant 1 or codominant 2 loci per chromosome: the number of Codominant 1 (or 2) loci on each chromosome, which positions are randomly drawn among the positions not occupied by **QTL/Loci under selection**.

NB.3.2-b: Please note that once the **Chromosome builder** option is activated, numbers for the different types of loci are controlled from the small "Number" windows within that option, thus it is not possible anymore to change the number of loci from the main "**Nuclear loci**" window.

NB.3.2-c: For ver 1.0.0, the maximum number of loci that can be chosen in the GUI is 9999 for each class (2 neutral classes and 1 class under selection) i.e. 29997 in total. More loci could be simulated by editing the conf.txt file, but large numbers of loci increase the runtime, and are rarely needed nor recommended in the preliminary steps when designing scenarios. Consistently, a similar total number is reached (29970 loci) when using the Chromosom builder with maximums of 10 chromosoms and 999 loci per class per chromosome.

3.2.3.2 Load map

A genetic map is produced automatically according to the rules detailed above when creating the conf.txt file. In scenarios where either frequencies or genotypes are loaded by the user, such a map is required beforehand if more than one type of locus are considered. The map thus needs to be loaded first with this option. To check the correct format for a genetic map, just save one first that is consistent with the scenario, using the "Save map" option of the same box.

Then the user chooses to load:

- either an allele frequency file consistent with the genetic map in terms of possible populations, loci and allele numbers (option **Load Frequencies** in the **Allele Frequencies** window of the *Genotype Tab*)

- or a genotype file consistent with the genetic map (option **Load Genotypes** in the **Allele Frequencies** window of the *Genotype Tab*).

For scenarios with more than one type of locus, the **Load Frequencies** or **Load Genotypes** radio buttons are thus available only after a genetic map has been previously loaded. Some consistency among loaded files is further checked when the conf.txt file is created in the *Run Tab*. More details and format examples of various genetic map files are available from the **tutorials** described in **Chapter 5** (see for example *5.5.1.2* and **NB.5.5.f**, and also **Table 4.2** for a full list of possible input files).

3.2.4 More information on underlying models

3.2.4.1 Mutation rates

Once genotypes have been assigned to offspring after one particular reproductive event ***t*** (***RE t***), the total number of mutations at each locus follows a Poisson distribution using the chosen values for mutation rates and variances in the GUI (see **test 5.2.1**). The mutant allele, corresponding individual and population are then randomly drawn at ***RE t*** (**Figure 3**). When a mutation occurs, the allele is replaced by any of the ***k***-1 other possible alleles (***k*** being the maximum number of alleles defined by the user), drawn at random. This corresponds to a ***k***-allele mutation model (Crow and Kimura, 1970). Regarding the potential allele additive values of QTL, this ***k***-allele mutation model corresponds to a "house-of-cards" mutational model (Kingman 1978), which assumes that "*The fitness of the mutant is chosen at random from a fixed fitness (effects) distribution*" (Zeng and Cockerham 1993). This model has been shown to be a good approximation for a variety of mutation models (Turelli 1984; Barton 1986; Roff 1997; Johnson and Barton 2005), for example in the case of alleles with small effects for most individuals and a mutation variance coming essentially from rarer individuals with alleles of larger effects (i.e. a leptokurtic distribution of allelic effects). Practically, allele fitness effects are either derived directly from chosen genotypes' fitness values in case of selection on genotypes, or indirectly and dynamically from individuals' fitness values in the case of selection on phenotypes, see also *3.4.4.2*, **3.6.6** and *3.7.6.3*). No stepwise mutation model is implemented yet, but arbitrary classes of co-dominant loci are possible in the GUI to easily model a class of loci with low mutation rates (e.g. 10^{-9}) such as SNPs, and another class of loci with contrasted and higher mutation rates (e.g. 10^{-5}) such as QTL or microsatellites.

3.2.4.2 Recombination rates

Neutral and potentially selected nuclear loci are placed on a given genetic map according to recombination rates chosen between adjacent loci. These rates are used when building the gametes for assigning genotypes to newly recruited offspring during the life-cycle (**Figure 3** and **Figure 3.4.4.1**). Recombination rates range between 0 (full linkage) and 0.5 (independent loci). gMetapop assumes no interference between loci, *i.e.* if you have 3 loci A, B and C in this order on the map, the recombination rate (r_{AC}) expected between A and C can be obtained from r_{AB} and r_{BC} with the formula $r_{AC} = r_{AB} + r_{BC} - 2r_{AB}r_{BC}$.

As explained above, if a genetic map is loaded, it needs to be compatible with the values that the users enters into the GUI, or with possible loaded genotype or allele frequency files (see **3.3 Genotype Tab**), in regards to the number of loci and/or alleles, and/or with the definition of loci as being under selection or not (see **3.4 Selection Tab**).

Although more than one cytoplasmic locus can be modelled with different mutation rates, those with either maternal or paternal inheritance are considered to be completely linked physically. However, at a particular generation, a locus with paternal inheritance and another locus with maternal inheritance could be considered independent.

3.2.4.3 Simulating neutral dominant marker loci

This feature was available in one of the historical METAPOP versions and has not been kept in gMetapop. However if needed, dominant marker loci could still be produced by first simulating co-dominant loci. Genotypes can then be saved either using sample files across generations (in the *Output Tab*) or using the final individuals' file at the end of the simulation (in the *Run Tab*). These files can then be loaded in R (//cran.r-project.org) with the read.table() function and genotype classes can be pooled according to which is the dominant chosen allele, in order to build two marker phenotypic classes. These data can further be post-treated in R, or in another software that deals with dominant marker data.

3.3 Genotype Tab

This Tab documents the maximum and **initial** number of alleles at all types of loci, and describes **how genotypes are generated at generation 0** based on different possible initial conditions or allele frequency distributions.

3.3.1 Number of alleles

Maximum number of alleles per locus: the values range from 1 to 256.

Initial number of alleles per locus: the values range from 1 to the maximum number of alleles per locus

NB.3.3-a: The reason for 256 is that alleles' possible indices/numbers are stored using one byte.

3.3.2 Allele frequencies

The genotypes at generation 0 are created by using the following options:

1) **Create**: the genotypes are generated from allele frequencies under the different GUI options proposed in each type of loci box (3.3.3 below). See also how initial allele frequencies are being treated with the options **"Compute"** or "**By hand**" in the "*Initial number of individuals*" window of the *Demography Tab*.

2) **Load frequencies:** the genotypes are generated from initial allele frequencies provided in a text file by the user, and loaded into the GUI (See **Table 4.2** for format examples provided in the various tutorials).

3) **Load genotypes:** the genotypes of all individuals present at the beginning of simulations are directly loaded from files prepared by the user (see examples in **tests 5.5.1**). For this option to be available, a genetic map needs to be loaded first in case of more than one type of locus (see in *Genome Tab*).

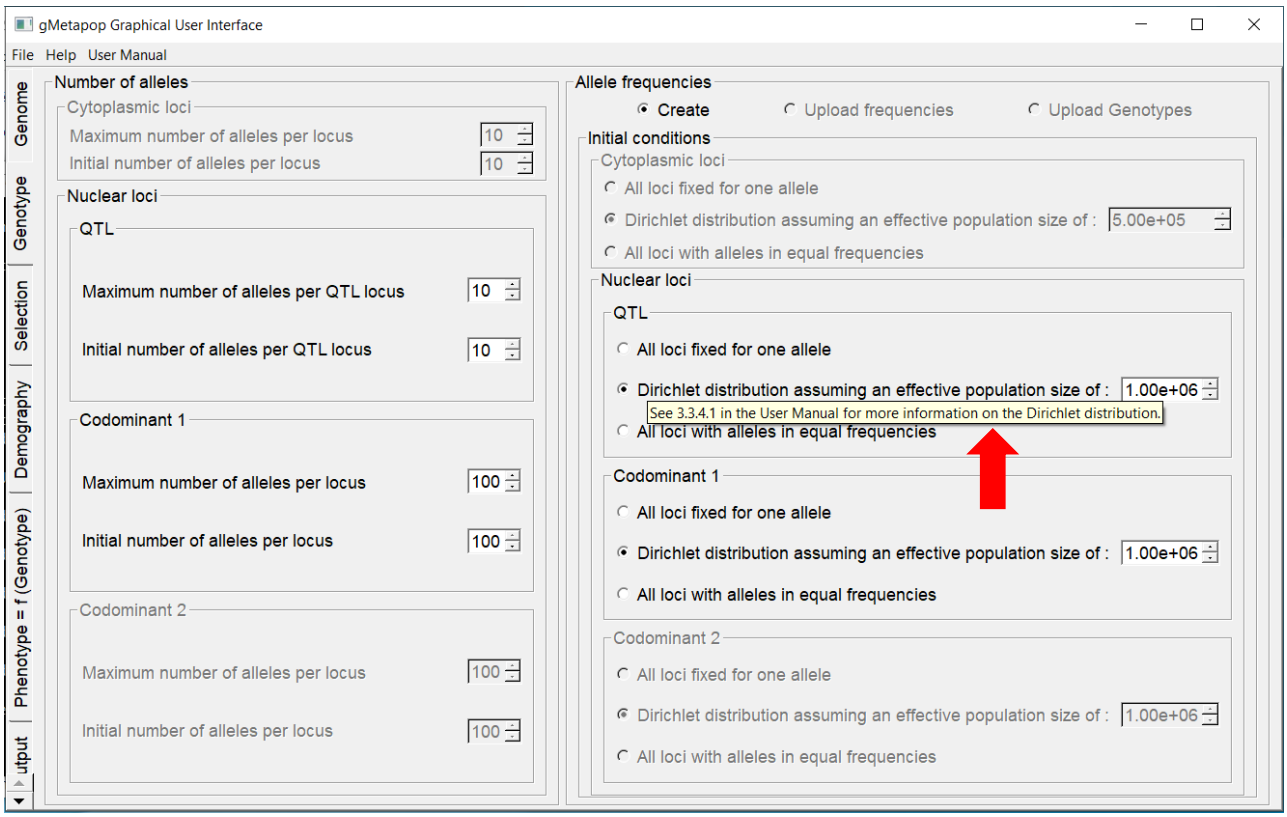
Additionally, initial number of individuals in one or more classes can be set in the *Demography Tab* and their genotypes will be created based on the conditions chosen above for 1) or 2) (see also **3.5.1** and **NB.5.5-i**).

3.3.3 Create/Initial conditions…

This allows choosing three pre-defined simple configurations across cytoplasmic or different categories of nuclear loci at the start of simulations from the GUI. Loci can either be fixed, or

their allele frequencies drawn from a Dirichlet distribution (see *3.3.4.1* below), or all their alleles can be in equal frequencies.

Figure 3.3: ***Genotype Tab*** example for a scenario including QTL (maximum allele number of 10) and 1 group of codominant loci (maximum allele number of 100) where initial genotypes are created based on allele frequency drawn from Dirichlet distributions. See also the tooltip example (red arrow).



3.3.4 More information

3.3.4.1 Initial allele frequencies from a Dirichlet distribution

Allele frequencies at the beginning of simulations are either set at given values by the user or drawn at random for each population in a Dirichlet distribution, in order to mimic a mutation-drift equilibrium in a *K*-alleles symmetric model (Ewens 2004). More precisely, denote **P** the vector of initial allelic frequencies $\mathbf{P} = (p_1, p_2, \ldots, p_K)$, these frequencies are drawn in the Dirichlet distribution with parameters $\boldsymbol{\alpha}=(\alpha_1, \alpha_2, ..., \alpha_K)$,

$$q(P;\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{K}\Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}\prod_{i=1}^{K} p_i^{\alpha_i - 1}$$

where

- Γ denotes the classical gamma function (see e.g. Abramowitz and Stegun, 1964)
- $\alpha_i = 4Ne\mu/(K\text{-}1)$, for $1 \le i \le K$, where μ is the mutation rate of the locus and *Ne* the effective population size of the population.

Note that since we assume here a *K*-allele symmetric model, all α_i parameters are set to the same value. **Test 5.2.2** was used to compare the simulated expected diversity values *He* to the theoretical ones across a range of mutation rates and *Ne* values with frequencies drawn

from Poisson-Dirichlet distributions, using the equilibrium formula **He = 4Neμ/(1+4Neμ*k/(k-1)),** where k is the number of alleles (p. 110 in Ewens 2004), and **μ** the mutation rate.

3.3.4.2 Format description for Input allele frequency file

Depending on the genome features chosen for a simulation scenario (numbers and types of loci, numbers of alleles and populations), the formats of allele frequency files can differ, so we recommend using the GUI to create them in a preliminary run (see more information in **Chapter 4** and various examples of preliminary runs in **Table 4.2** and tutorials of **Chapter 5**).

3.3.4.3 Format description of Input Genotype file

Similarly, genotype file formats can be obtained in preliminary runs using the GUI, according to the user's simulation settings (see as above for examples provided).

3.4 Selection Tab

Selection is applied either directly on phenotypes *via* the choice of optimizing selection strength values within populations (so indirectly on underlying QTL) and the choice of optimal phenotypic values across populations, or directly on genotypes *via* the choice of fitness values at each locus under selection. **Figure 3** shows where different types of selection can be applied in a typical life cycle.

3.4.1 No selection

All loci are neutral, but different classes of loci with various options for mutation and recombination rates are possible in the ***Genome Tab***.

NB.3.4-a: No selection can be applied on cytoplasmic loci but this may change in future versions of the program, depending on users' feedback.

3.4.2 Selection

Selection occurs *via* differences in the **realized** male and female fecundities among individuals, which are proportional to their genotypes' fitness values.

NB.3.4-b: In our framework, we define the **potential fecundity** of an individual (used as male or female) as that of its population or of its particular class within a population (e.g. this could correspond to demographic features of groups of individuals belonging to particular populations in a species) in a neutral model. The values for potential fecundity are set up in the **3.5 Demography Tab** below. The **realized fecundity** of one individual will depend first on its **potential fecundity** and different density-dependent regulation steps (e.g. stochastic demographic mortality in classes in the more complex age/class structure modelled when generations overlap, see **Figure 3**, and *3.5.1.2*, **3.5.3**, *3.5.5.4* to *3.5.5.6*), and second of its relative genotypic fitness value in the case of selection (see also **Figure 3.4.4.1** and *3.4.4.2*).

3.4.2.1 Selection on Phenotypes

Within populations:

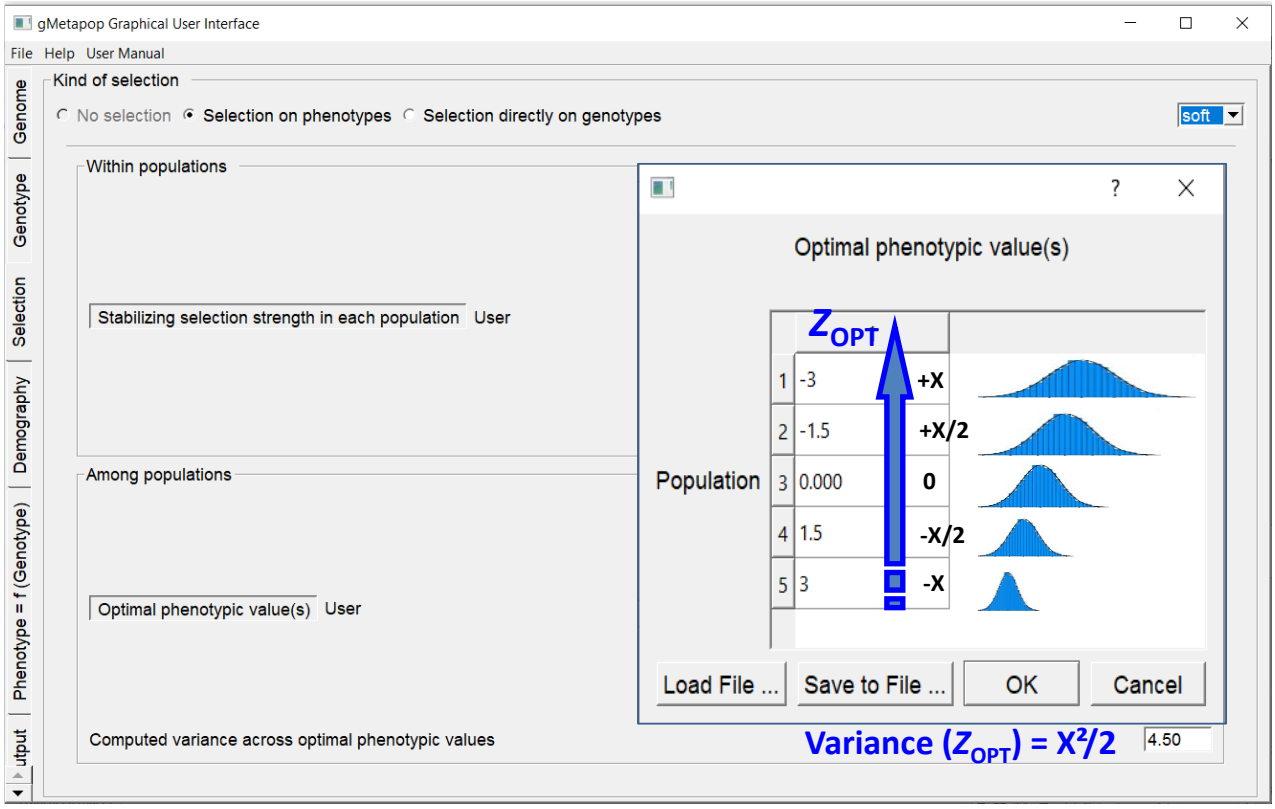
Stabilizing selection strength in each population: values indicate the intensity of stabilizing selection within populations. They can differ across populations and allow fitness values to be attributed to individuals based on their phenotypic values (see more information in *3.4.4.2*).

Among populations:

Optimal phenotypic value(s): …can be the same (i.e. homogenizing selection) or vary across populations (i.e. allowing for diversifying selection among populations, see **Figure 3** and *3.4.4.3*), mimicking for example environmental gradients (**Figure 3.4** below).

Computed variance across optimal phenotypic values: the default variance is 1, or it is computed *a posteriori* from chosen or loaded (optimal) phenotypic values across populations.

Figure 3.4: ***Selection Tab*** example for a scenario of clinal selection across 5 populations with different phenotypic optima ***Zopt*** (see the computed variance of 4.50 across optimum values), and different strengths of optimizing selection within each population (click on the **Stabilizing selection strength in each population** window to see corresponding values).



3.4.2.2 Selection on Genotypes

For each locus, fitness values are assigned by the user to all possible homozygote and heterozygote genotypes (**Figure 3**). The fitness of an individual is the product of its fitness values at all loci, inducing in particular a null fitness value for individuals with null fitness at one locus.

Matrix(ces) of fitness values for user editing: opens [genotype by population] tables, locus by locus. A **recurrence rule** is used when clicking "OK" to go to the next locus, or when loading a text file for one or more loci. Default values are "1" for all genotypes (neutrality).

NB.3.4-c: Recurrence rule: the program uses the last edited, checked or loaded set of values to fill matrices for all other loci that are potentially under selection. This allows to rapidly filling a set of variable fitness values across different populations for a large number of loci with similar coefficients. For more complex cases, we suggest that you first check the configuration file (conf.txt) in the working folder, after it has been created in the *Run Tab*, and then adjust the fitness values either from the GUI before creating a new conf.txt file, or directly by editing the conf.txt, using a small R script (see **NB.3.2-a**).

What is described above holds before saving the *param* file. However, once a *param* file has been saved, it deactivates the **recurrence rule** in such a way that fitness values can only be edited or loaded for one locus at the time. You need to start a new *param* file to benefit again from this rule (or alternatively edit the conf.txt file separately, see **NB.3.2-a**).

3.4.3 Hard or soft selection

These terms refer to **the first step** of the selection process that deals with the number of offspring produced. In gMetapop, these processes have been modelled in the following manner (following Wallace 1968, 1975; see also Lowe *et al*. 2017; Reznick 2016):

Soft selection (default): The total number of offspring from each parental population depends **only** on the potential female fecundity per class and population, and how it may further be affected by the demographic model and regulation steps (**Figure 3**).

Hard selection: The total number of offspring from each parent population (or class per population) depends on the potential female fecundity per class and population, on the demographic model and its regulation steps, and **is also** multiplied by the average fitness of individuals in this population (non-overlapping generations) or of those belonging to this class from this population (overlapping generations).

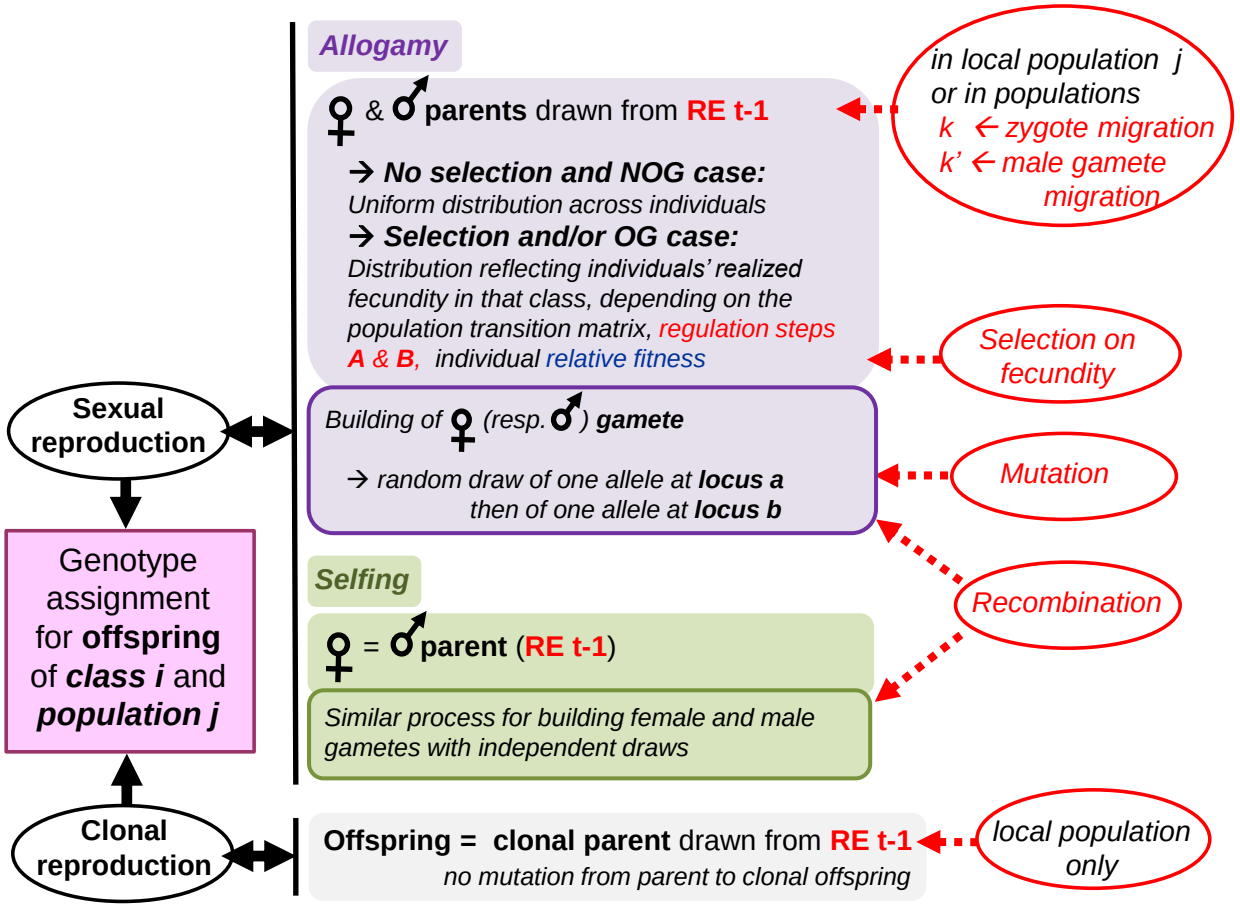
3.4.4. More information on selection processes

3.4.4.1 Application of selection and building offspring genotypes

Practically, selection on genotypes operates in two steps. First, it affects the number of offsprings produced in the case of **hard selection** only (**Figure 3**; see also above and *Demography Tab 3.5.1.2* below for the description of how offspring are drawn). Second, it affects the offspring genotypes in the sense that potential parents with higher fitness values will be more likely to be drawn since the probability of a parent to be drawn as a mother or as a father is proportional to its fitness (**Figure 3.4.4.1** below). More precisely, once the final number of offspring is set in each population after the regulation steps, the male and female parents of each offspring issued from sexual reproduction are drawn according to their **realized fecundity** from the previous set of reproductive events in their life cycle, and thus according to their relative proportions across populations and classes. The genotype of a newly recruited offspring is built by drawing first its female parent, the maternal gamete being chosen by randomly drawing alleles across loci in that parent individual, accounting for recombination rates between loci. The male parent is then drawn either from the same population than the female parent or from another population according to the male gamete matrix flow (see **3.5.4** and **Figure 3.4.4.1** below). The paternal gamete of the new offspring is built following the same algorithm than for the female gamete. Once gametes have been associated to a new individual, mutant alleles are drawn at random for each locus, based on chosen mutation rates and variances (see **Figure 3**, **Figure 3.4.4.1**, *Genome Tab* and test **5.2.1**). For offspring coming from clonal reproduction, their genotypes is the same than that of their parent drawn at random in the same population, based on previously applied clonal fecundity values across classes.

Figure 3.4.4.1: Schematic algorithm for the assignment of diploid genotypes to offspring individuals, in the genetic part of the cycle of events described in **Figure 3**. Stochastic evolutionary processes are indicated in red. Please note that in the program, the mutation process occurs once the gametes have been formed, independently to the recombination process. ***RE t***: set of reproductive events at generation or time ***t*.** Density-dependent regulations **A** (**A1** and **A2)** and **B** of population and class sizes are further described in *3.5.1.2*,

3.5.5.4 and *3.5.5.5*. Populations *k* and *k'* are the source of migrant parents via zygotes or the male gametes, and they can be the same. ***OG***: overlapping generations. ***NOG***: non-***OG***.



The fitness values of offspring genotypes are then either computed from their corresponding phenotypic value (selection on phenotypes, see *3.4.2.1* above), or by multiplying the fitness values corresponding to their genotype at each selected locus for scenarios of selection directly on genotypes (see below).

NB.3.4-d: It is not yet possible to run a simulation that combines both selection on genotypes and selection on phenotypes, or to directly include dominance or epistasis in phenotypic values. These options might be included in a future release.

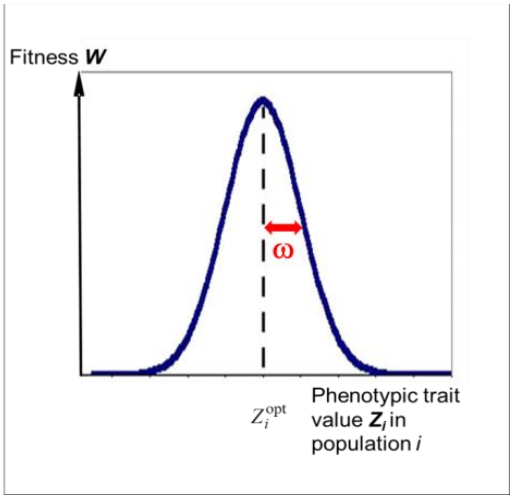
3.4.4.2 Fitness values and modelling optimizing selection

In scenarios of selection on a phenotypic trait within populations, gMetapop assumes optimizing selection, also called stabilizing selection, used here with the same meaning, but more types of selection may be added in the future. It means that individuals with intermediate phenotypic values have the highest fitness. This type of selection is modelled using a convenient and widely used fitness function in evolutionary genetics (Johnson and Barton 2005; Walsh 2007; Charlesworth and Charlesworth 2010), called the *nor-optimal* or normalizing fitness function: the fitness value $W_i(Z)$ of an individual with phenotype Z in a given population i is given by:

$$W_i(Z) = \exp\left(-\frac{\left(Z - Z_i^{opt}\right)^2}{2\omega^2}\right)$$

where Z_i^{opt} is the optimal phenotype in population *i,* and $1/\omega^2$ denotes the intensity of selection (**Figure 3.4.4.2**). Thus $W_i(Z)$ decreases with increasing differences between the phenotypic value Z and the phenotypic optimum Z_i^{opt}, and depends also on the selection strength $1/\omega^2$ (see also **3.6.6** for more on the phenotypic value decomposition).

Figure 3.4.4.2: The *nor-optimal* fitness model. The equation is plotted for fitness with an intensity of stabilizing selection of $1/\omega^2$.



This fitness function is thus proportional to a Normal density function. To set up parameter values for ω and Z_i^{opt} that would be consistent with biological data or natural conditions, both theoretical results and available meta-analyses of experimental results can be used. Early reviews of experimental data using mutation-selection models and simulations showed that for most traits ω^2/V_E falls between 5 and 20 (Roff 1997 p. 358-361; Turelli 1984). Also, based on Kingsolver *et al*.'s (2001) review of estimated selection gradients γ (and assuming *nor-optimal* stabilizing selection so that $\omega^2/V_E = -1/[2*\gamma*(1-h^2)]$), Johnson and Barton (2005) showed that many significant estimates of γ translate into strong stabilizing selection strength, based on previous reviews (e.g. Lande 1975). Therefore ω^2/V_E may show large ranges of values, it may be far below 5 in natural populations for traits with low heritabilities, and quite difficult to estimate on a single trait in natural populations for various statistical and biological reasons (see Walsh and Lynch 2018, Chapter 30 for a review of Kingsolver *et al.* (2001) meta-analysis and following works). In general, values of 5 and below for ω^2 would transfer into strong selection, while 100 and above would correspond to a weak selection pressure.

For scenarios with selection on genotypes, fitness values are defined across genotypes for each locus, allowing for example deleterious alleles, recessivity and over- or under-dominance to be simulated. More flexibility is also provided by being able to directly set up those fitness values across populations in the GUI, and thus simulate various spatial fitness landscapes. The fitness of an individual is the product of the fitness values of his genotype at each locus.

3.4.4.3 Applying divergent selection among populations

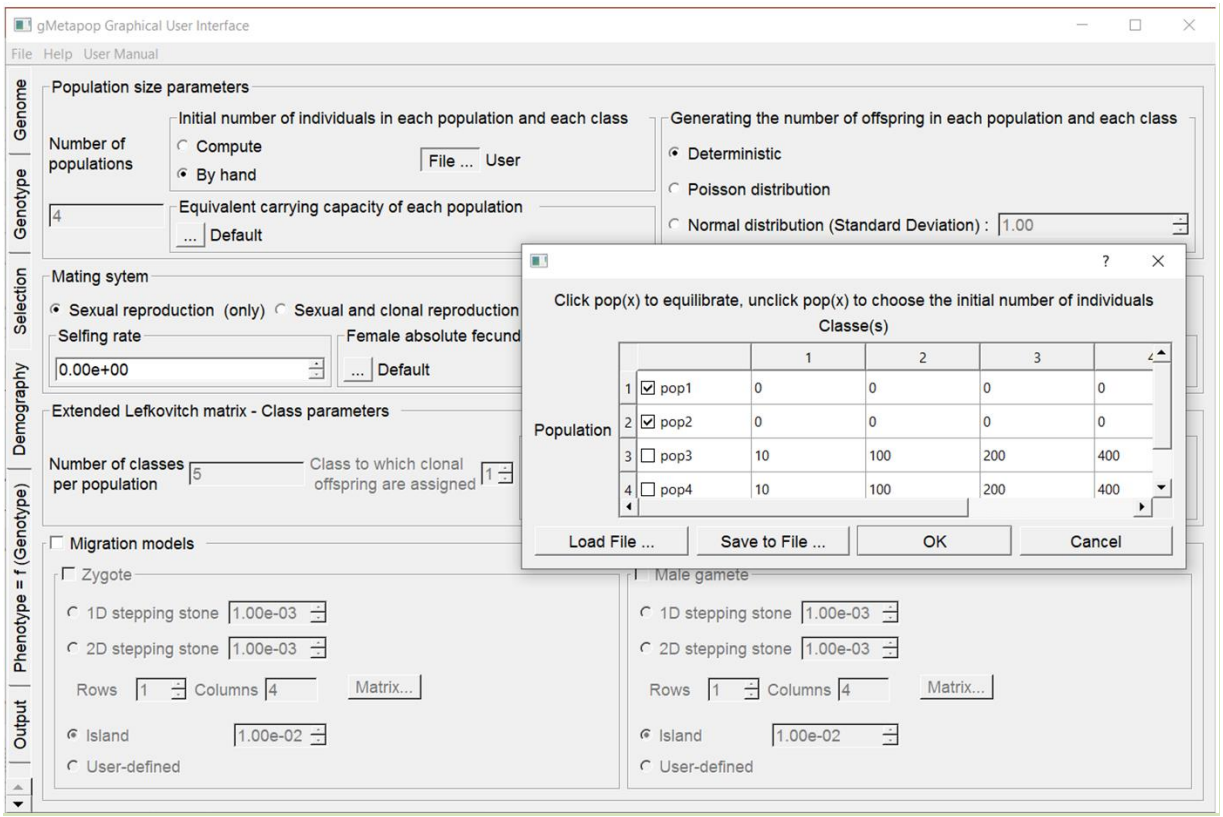
In a metapopulation context, selection among populations can be applied by setting up different phenotypic optima values, Z_i^{opt} for each population *i*. The variance V_{Zopt} between

optima characterizes the strength of the selection among populations, from uniform (zero variance) to diversifying (high variance among optima). By default, optimal phenotypic values correspond to a linear gradient of optima across populations for a variance among optima of 1, whatever the number of populations (but see other examples in **Tests 7** and **Tests 10** of **Chapter 5**). These values can be easily modified in the GUI manually, or loaded from a simple one column text file. Individual phenotypic values are initially all centered on zero. They will be centered around their population optimum after a variable rounds of selection, depending on the strength of selection, and on whether a relatively stable equibrium can be reached during simulations. Phenotypic values can be requested in the *Output Tab* with the "**Extracting sample files…**" option (see **3.7.4**) The strength of within population optimizing selection intensities can vary across populations, which allows a great level of flexibility in terms of possible selection scenarios and spatial landscapes in which populations could be set up, with or without gene flow (see also **Figure 3.4** and *3.5.5.3*).

3.5 Demography Tab

This Tab displays all demographic features available in the program, including the size class structure for simulating overlapping generations (*OG*), and different types of mating systems and various migration models (Figures **3.5.1**, **3.5.2** and **4.2**).

Figure 3.5.1: ***Demography Tab*** example in a scenario including 4 populations and 5 "age" classes. The **"by hand"** option allows the user to choose the initial distribution of individuals' in at least one population. The populations which are clicked still undergo the search for an initial demographic equilibrium based on the demographic parameter values entered in this Tab (see **3.5.1** for a description of all different cases).



3.5.1 Population size parameters

3.5.1.1 Initial number of individuals in each population and each class

Compute: the program computes and searches for an initial equilibrium distribution of size classes in all populations, based on mating system values and class parameters but using deterministic offspring numbers, and assuming no selection, no migration, and no stochasticity. Practically, this step is useful in the case of overlapping generations (*OG*, see **Figure 3** and **Figure 3.5.1.1**). It is performed before starting a simulation following the demographic model detailed in *3.5.5.5*. For non-*OG*, the initial number of individuals in each population is close to its carrying capacity.

NB.3.5-a: Please note, therefore, that for a neutral scenario with no selection, no migration and a deterministic drawing of offspring, the expected demographic equilibrium should be similar to the initial deterministic equilibrium, even if a small amount of stochasticity is added. On the other hand, if migration, selection, or different ways of drawing offspring are being used, the associated stochasticity may be higher, and it might lead to a different equilibrium situation than the initial one (given that an equilibrium situation is reached). Also, due to one of the density dependent regulation added to the demographic model (regulation **B**, see in **Figure 3** and in part ***3.5.5.4***), the initial number of individuals at equilibrium is often around K-0.001*K. This is more detectable when larger population sizes are used. In such cases, the value of K can be adjusted if needed to start with a more precise number of initial individuals.

Convergence criterion: a maximum of 10000 iterations in each population are performed for the search of an initial demographic equilibrium. This is also done for populations which are clicked when using the By Hand option. One iteration corresponds to one reproductive event in *non-OG*, and to all demographic events (changes of class sizes) during one time step in the case of *OG*.

Different cases can occur:

Case 1: a stable initial demographic equilibrium is reached and all populations are nonempty, using within each population **the criterion** that the mean of relative differences between class sizes in each population should be less than 10^{-4} between two iterations. This criterion has to be reached before iteration 10000 for each population.

Case 2: a stable initial demographic equilibrium is reached following the same criterion, but with at least one population size converging towards zero.

In **Cases 1** and **2**, the initial equilibrium values for population and class sizes are taken as starting points of the simulations, and can be obtained with demography statistics (see **3.7**).

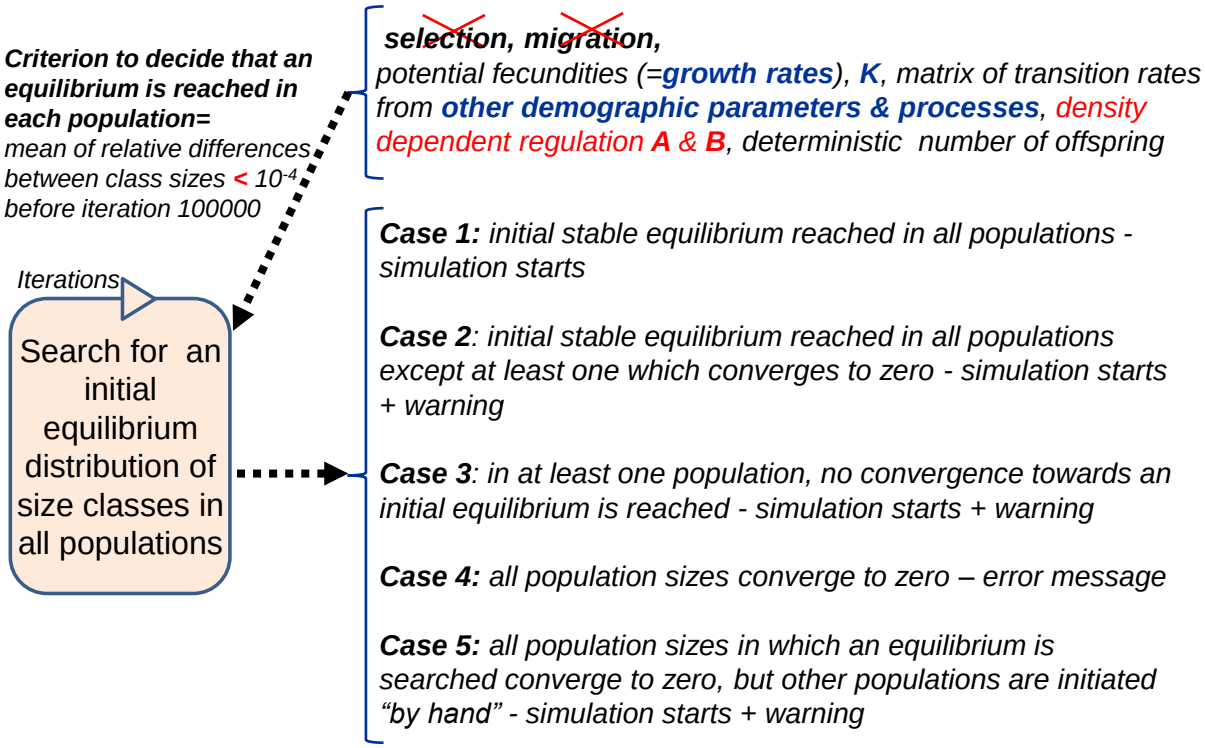
In **Case 1**, no warning is issued.

In **Case 2** the following warning is issued (and the simulation can proceed):

"**WARNING**: In the populations undergoing the search for an initial demographic equilibrium distribution, numbers of individuals have converged to zero in at least one population (before iteration 10000), if considered in isolation from other populations. The initial distribution can be checked by requesting demography statistics in the *Output Tab*, and demographic parameter values can be changed if needed. Alternatively, with the "By Hand" option, more or all populations can be unclicked, and new conditions can be tested."

NB.3.5-b: Even if the previous warning is issued, remember that neither migration nor selection occur in the search for an initial demographic equilibrium. Therefore, with different conditions, the actual simulation might produce different outcomes. For example, populations that would have gone extinct in isolation, might receive migrants from the other populations if the migration rates are different from zero during the simulation.

Figure 3.5.1.1: Summary of the main cases of initial demographic conditions (see also **Figure 3**).



Case 3: in at least one population, no convergence towards an initial equilibrium is reached. In such population(s), class size values are chosen from an iteration that is drawn at random among the 100 last iterations. The following warning is issued:

"**WARNING**: In the populations undergoing the search for an initial demographic equilibrium distribution, no convergence has been reached in at least one population (before iteration 10000), using a criterion of less than 10^{-4} for the mean of relative differences between class sizes, within each population, when going from one iteration to the next. This maybe due to cyclic demographic fluctuations of individuals' distribution in classes across iterations. The initial distribution can be checked by requesting demography statistics in the *Output Tab*, and demographic parameter values can be changed if needed. Alternatively, with the "**By Hand**" option, more or all populations can be unclicked, and new conditions can be tested (see examples of Case 3 in **Tests 5.10.1** in Chapter 5)."

Case 4: all population sizes converge to zero, and the following error is issued:

"**ERR**: During the search for an initial demographic equilibrium distribution, numbers of individuals have converged to zero in all populations and classes (before iteration 10000). This is likely due to demographic values which lead to extinction of populations (e.g. too strong demographic regulation, too low female fecundities). Please review those values and try again until a non-zero initial equilibrium is reached in at least one population, or until the population and class sizes reached before iteration 10000 allow simulations to proceed. Alternatively, with the "**By Hand**" option, more or all populations can be unclicked, and new transitory conditions can be tested."

Case 5: all population sizes in which the initial demographic equilibrium is searched converge to zero, but other populations are initiated "**by hand"** (see below), thus the following Warning is issued:

"**WARNING**: In the populations undergoing the search for an initial demographic equilibrium distribution, numbers of individuals have converged to zero in all populations and classes (before iteration 10000), if considered in isolation from other populations. This is likely due to demographic values which lead to extinction of populations (e.g. too strong demographic regulation, too low female fecundities). Please review those values if needed, and try again if you are looking for an initial demographic equilibrium in at least one of those populations, or for a non-zero size before iteration 10000. However, the simulation is launched since the "By Hand" option is used for other populations."

By hand(/File): the user sets the initial number of individuals in each class of each unclicked population (/File option), while an initial demographic equilibrium is searched as with the **Compute** option above for the populations which are clicked.

NB.3.5-c: With the "**By hand/File**" option, the user either fills the table in the GUI, or loads a matrix file (*.csv/txt file **with ";" delimiters**) with the chosen numbers of individuals for population and classes. Also, with this option, if allele frequencies are loaded in the GUI, they are used for drawing individual genotypes. If a population is empty, non-zero allele frequencies are ignored. In general, an initial frequency file that is loaded in the *Genotype Tab* needs to be compatible with the number of populations and individuals chosen in the "By hand" option. Otherwise error messages are issued either in the GUI, or when running the CORE, depending on the configuration (see the Appendix online for more on error description). For example, this will be the case if allele frequencies are null or missing in loaded files while populations are defined as being non-empty.

NB.3.5-d: The "**By hand**" option is not activated if a genotype file is loaded in the *Genotype Tab*.

Equivalent carrying capacity of each population (*K*): Equilibrium number of individuals (non-overlapping generations), or maximum deme occupancy. In scenarios with overlapping generations, equivalent class sizes may vary across individual classes and are accounted for when searching for an initial equilibrium distribution (see the **Compute** option above and the different cases described).

NB.3.5-e: Be aware that in the case of overlapping generations combined with a search for an initial demographic equilibrium (see **Compute** above), initial population sizes could be lower than the equilibrium carrying capacity ***K*** values. They will depend on the combination of demographic conditions, which are chosen at the start of simulations.

3.5.1.2 Generating the number of offspring in each population and each class

The expected number of offspring in a new life cycle in each population is first computed based on the available number of parents, their potential female fecundity values, and migration rates towards that population (see **Figure 3** for a typical life cycle, and **Figure 3.5.2** below). In case of **hard selection**, the number of offspring also depends on the fitness value of their class (more details in *3.4.4.1*). Second, density-dependent **regulation** processes occur in different ways, referred to as **A1** (non-overlapping generations, see *3.5.5.4*) or **A2** (overlapping generations, see *3.5.5.5*) and **B** (both non-overlapping and overlapping generations, *3.5.5.4* and *3.5.5.5*), depending on the initial settings and demographic models chosen. Briefly, regulation coefficients are computed based on current population sizes and potential growth, demographic cycles and events, and possible class structure. Third, the amount of stochasticity is also differently applied depending on the following cases:

Deterministic: It corresponds to the case with the least possible stochasticity once the equilibrium carrying capacity is reached. The realized number of offspring is computed by simply adding a value drawn from a uniform distribution between 0 and 1 to the potential number having undergone regulation. This would be needed in scenarios with very low growth and/or migration rates. When the demographic equilibrium has been reached, the population size is regulated so that it remains stable from one set of reproductive events to the next ones, and the expected theoretical drift applies on allele frequencies (see **5.1** for an example).

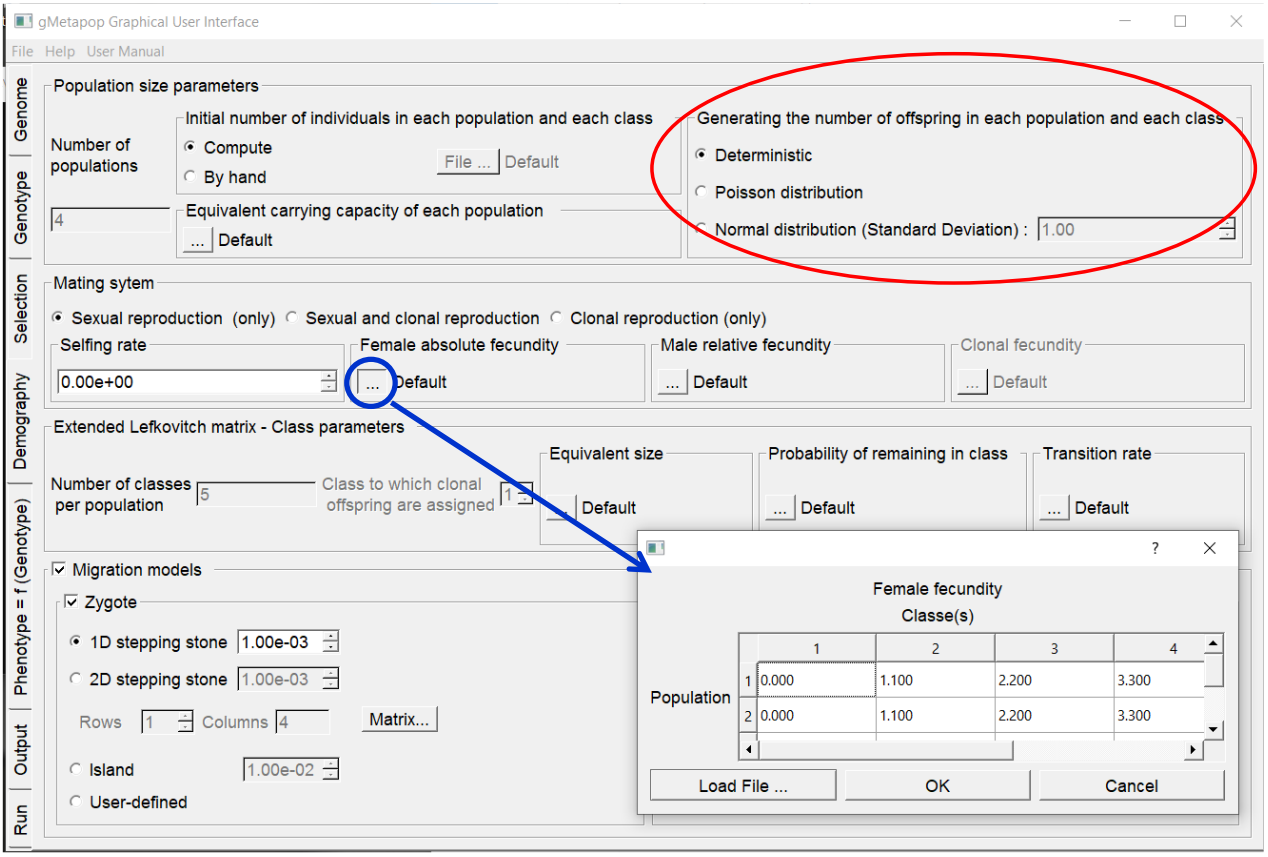
Poisson distribution: here the realized number of offspring is drawn from a Poisson distribution, which parameter is the number of offspring having undergone the different rounds of regulation. The Poisson distribution can be chosen in population models where a stable demographic equilibrium is not particularly expected, or where a larger variance in the number of individuals from one reproductive event to the next is needed. Therefore, even if populations start from initial chosen ***K*** equilibrium values, the drift and stochasticity will likely be higher than in the deterministic situation (see for example the comparison between scenarios in **5.10.1**). Another possible use of such distribution is when low to very low

numbers of individuals need to be simulated, the variance of the Poisson distribution corresponding also to the expected number of offspring, and thus being potentially very large.

Normal distribution (standard deviation): the realized number of offspring is obtained by adding to its expected value after regulation, a number drawn from a zero-centered normal distribution with standard deviation chosen by the user, which allows controlling the extent of stochasticity in the number of offspring produced. A very large variance can lead some populations to stochastic extinction (see one example in test **5.10.3**).

Population and class sizes then follow the logistic growth model towards the equivalent carrying capacity (***K***) chosen (see *3.5.5.4*), or remain at around ***K*** when ***K*** has been reached (mimicking a demographic equilibrium).

Figure 3.5.2: ***Demography Tab*** example in a scenario including 4 populations and 5 "age" classes. The secondary window shows here population by classes female fecundity values for 2 among the 4 populations. It is obtained by clicking on the "Female absolute fecundity" Default square (circled **in blue**). The GUI options for offspring generation are circled **in red**.



The number of individuals remaining at each generation (and of adults and offsprings produced in case of overlapping generations) can also be followed with some demography summary statistics in result files (see **3.7.5** and **3.7.6**, *Output Tab* below).

3.5.2. Mating system

Selfing rate: proportion of offsprings produced by self-fertilization, added to the proportion of selfed offspring produced at random, given the population size.

Female absolute fecundity: female potential fecundity of individuals (mean offspring number) of a given class in each population.

Male relative fecundity: male potential fecundity of individuals of a given class in each population, which indicates the relative contribution of that class in each population when drawing male parents.

Clonal fecundity: absolute clonal fecundity of individuals (mean offspring number) of a given class across all populations. Note that offspring issued from clonal reproduction only come from the local population, and that they do not present any new mutation compared to their parent (see **Figure 3.4.4.1** and more details in part *3.5.5.7*).

3.5.3. Extended Lefkovitch matrix - Class parameters

Equivalent size: Equivalent size of an individual of a given class, or relative area occupied by perennial organisms in a class, values can be above or below one, see tests **5.9** and **5.10** for examples with overlapping generations in **Chapter 5**.

Probability of remaining in class: proportion of individuals that survive and stay in the same class from one reproductive/demographic event to another, in the absence of density dependence (see **NB.3.5-f** below).

Transition rate: proportion of individuals that survive and move to the next class from one (set of) demographic event(s) during the life cycle to another.

NB.3.5-f: In our framework and in the default case of **soft selection**, mortality and survival of individuals are treated only in a context of demographic stochasticity and regulation, and thus independently of their genotypes and of the application of selection. In the case of non-overlapping generations, all individuals die simultaneously. In the case of overlapping generations, they die with a probability that depends on their size class and other demographic parameter values in the more complex age-structured model implemented (see *3.5.5.5* below). It is only in the case of **hard selection** that the classes' mean fitness can impact the overall class mortality. See also **3.4.3** and *3.4.4.1* and *3.5.5.6*).

3.5.4 Migration models

In the "Zygote" window:

1D stepping-stone: populations are organized on a line and exchange migrants only with their two neighboring populations.

2D stepping-stone: populations are organized on a two-dimensional area and exchange migrants only with their four neighboring populations.

NB.3.5.g: See also **Tests 5.4** in Chapter 5, which illustrate the predictions of both stepping-stone migration models above for estimating dispersal variances using simulations.

Island: migrants are exchanged at the same rate among all populations (example in **Figure 4.2**).

User-defined: the user either sets all migration rates one by one from the GUI, or loads them from a file, which allows much flexibility in tested scenarios. The matrix can also be saved and used in other *param* files.

In the "Male gamete" window: Male gametes from different populations can contribute to offspring of a new reproduction event *via* the choice of male parents according to the different proposed or user-defined migration models (see **Figure 3.4.4.1**).

3.5.5. More information on demography, mating systems and migration models

3.5.5.1. Zygote migration models

Zygote dispersal is set by a matrix of migration rates ***M*** = (m_{ij}) between populations, where m_{ij} denotes the proportion of zygotes (e.g. seeds) that potentially migrates from population *i* to population *j*, as female parents for new reproduction events. This migration matrix can be set up from real estimates of zygote flow if available (*e.g*. estimated proportion of seeds coming from the neighbouring populations estimated in maternity analysis or from direct observations of seed dispersal). For instance, if we assume a stepping stone migration model with four populations and a rate of 0.02, the zygote dispersal matrix is:

$$M = \begin{pmatrix} 0.99 & 0.01 & 0 & 0 \\ 0.01 & 0.98 & 0.01 & 0 \\ 0 & 0.01 & 0.98 & 0.01 \\ 0 & 0 & 0.01 & 0.99 \end{pmatrix}$$

The ***M*** matrix can also refer to any theoretical model or be defined by the user from observed data and imported into the GUI.

3.5.5.2. Male gamete migration models

Male gamete (e.g. pollen) dispersal is set by a migration matrix ***P*** = (p_{ij}) between populations, where p_{ij} denotes the proportion of male gamete that potentially migrates from population *i* to population *j*, via the choice of male parents for new reproduction events. This migration matrix can be set up similarly to the zygote dispersal matrix from a theoretical migration model (*e.g.* a stepping stone migration model available in the GUI), from real estimates of male gamete flow if available, or from a user-defined migration matrix that can be imported.

3.5.5.3 Flexibility of the demographic model and metapopulation dynamics

One strength of this program is to allow for overlapping reproductive events or generations, using density-dependent regulation processes in the life cycle which is modelled (**Figure 3**). In combination with the other options available for mating systems, offspring generation and user-defined migration rates among populations, this opens a very large range of possible evolutionary scenarios in a metapopulation context, with more or less complex demographic characteristics.

We provide a few examples below:

Although there is no individual spatial structuring within populations (*i.e.* individuals are not identified by spatial coordinates), each population can be considered as a particular location (cell) in a spatial landscape, offering different metapopulation dynamics. Practically, there is the possibility to use a combination of:

1) Particular migration models implemented though gene flow matrices, and

2) A large number of populations (which can be structured in age classes, see *3.5.5.5* below) with user-defined initial/maximum numbers of individuals. This easily allows mimicking spatial landscapes and positioning of populations with a large amount of flexibility, where different types of selection can also be assigned to each population (or groups of populations, see *3.5.5.1* and *3.5.5.2* above) according to realistic or particular environmental landscapes.

3) Different manners in which some populations can go to stochastic extinction, either using a very large variance in the case of the "Normal distribution" option for generating offspring (see one example in test **5.10.3**), or using a strong selection in some populations to mimick particular habitats, or via different combinations of demographic parameter values of the class parameters (see **3.5.3**)**.** In the case of isolated populations and no possibility of migrants to recolonize empty populations, warnings are issued, depending on different types of initial conditions (**Figure 3.5.1.1**).

4) A total flexibility of setting up initial number if individuals in different populations (and classes), allowing population growth to change though time according to demographic parameters and regulation processes, in combination with possible types of selection. This allows to explore many transitory conditions of population dynamics, before reaching an equilibrium, even if all populations may finally reach extinction.

5) Variable mating landscapes across populations using the possibility to attribute different potential fecundities to both populations and (developmental) classes in the case of overlapping generations. Since the individuals that can mate are hermaphrodites, they can be drawn either as a male or as a female parent for each offspring (with possible random or additional selfing), but there is no genetic sex determinism in this version of the program. However, different populations and age classes can harbour null female and/or male fecundities values, as well as variable clonal fecundity values across different classes of individuals.

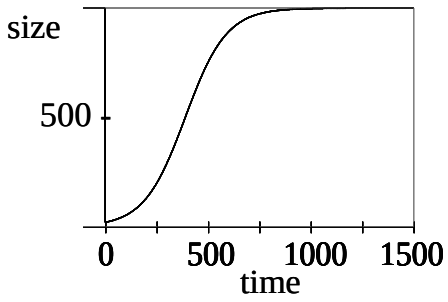
3.5.5.4 Demography with non-overlapping generations

In case of non-overlapping generations, all individuals are born and die simultaneously after each reproduction event, with possible clonal reproduction (e.g. vegetative reproduction, and see **Tests 8** in Chapter 5). In this case, populations grow at a logistic growth rate *f,* which is the value set for the female potential fecundity in each population, until they reach their equilibrium size (***K***) specified by the user in each population in the "**Equivalent carrying capacity of each population**" box of the *Demography Tab*. The size N_{t+1} of the population at time t+1 is deduced from the population size N_t at time t using the formula $N_{t+1} = N_t + fN_t(1-N_t/K)$. Fecundity values can be chosen in agreement with experimental data and the logistic growth function allows accounting for density (see **Figure 3.5.5.4** below and the **regulation** process occurring is referred to as **A1** in **Figure 3**).

In practice with logistic growth, the growth rates need to be relatively small from one generation to the next. Thus in case of large potential fecundity values, N_{t+1} values could rapidly and strongly exceed ***K*** values. This effect will be exacerbated in the case of high migration rates from other populations since the **regulation** process **A1** depends on the previous size (and potential growth) of the target population only. In those cases, an additional density dependent **regulation** process has been implemented (referred to as regulation **B** in **Figure 3** and throughout the User Manual), which prevents N_{t+1} values to exceed a value of 1.15*K (i.e. they cannot exceed K by more than 15% of K), allowing for a certain amount of stochasticity around the carrying capacity.

In cases of offspring generated using Poisson or Normal distributions (see *3.5.1.2*), the population growth trajectory will be similar with added stochasticity.

Figure 3.5.5.4: Example of logistic growth with density-dependence



3.5.5.5 Demography with overlapping generations

In cases with overlapping generations, the model implemented assumes that there are several size classes in all populations. This model belongs to a class of models developed originally by Lefkovitch (1965), who proposed that these classes could be stages of development, for example age classes or ontogenetic classes (Lefkovitch 1965, Caswell 2001). The model used here incorporates density-dependent growth regulation of each class (referred to as **regulation** process **A2** in this document, see also **Figure 3**), and the possibility of different relative sizes of each class (see Austerlitz *et al.*, 2000 for more details on rationale, model development and applications).

More precisely in gMetapop, the number of classes (k) is initially set by the user and does not vary during the simulation. Denote $\mathbf{N}(t)=(N_1(t), N_2(t),\ldots, N_k(t))$ the vector of the number of individuals in each size class after a reproductive event t. Parameter values set by the user in the **Extended Lefkovitch matrix - Class parameters** group-box of the *Demography Tab* allow building a transition matrix (**A**) from t to $t+1$ defined as follow:

$$ {}^{1}\mathbf{A} = \begin{pmatrix} P_{11}+f_1 & f_2 & f_3 & \cdots & f_k \\ P_{12} & P_{22} & 0 & \cdots & 0 \\ 0 & P_{23} & P_{33} & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & P_{k-1,k} & P_{kk} \end{pmatrix} $$

where $\mathbf{N}(t+1)=\mathbf{A}.\mathbf{N}(t)$ with :

- P_{ii} is the proportion of individuals that stay alive and remain in class i from t to $t+1$
- P_{ii+1} the proportion of individuals of class i that stay alive and move from the class of size i to $i+1$ during the time interval t to $t+1$
- f_i denotes the potential female fecundity of an individual of class i in a particular population.
- The proportion of individuals that die in class i during a given time step is $1 - P_{ii} - P_{ii+1}$

In order that the populations do not grow indefinitely, the density-dependent growth **regulation process** acts as follows: when density increases, both the probabilities P_{ii+1} of moving to the next class and the fecundities f_i decrease, and conversely mortality increases. This is performed by assigning to the individuals of each class *i* a relative equivalent size g_i (e.g. a given stand basal area, "stand" being a term used generally for a forest tree population, which is a measure of the space occupied by an individual, thus we use stand basal area in the rest of this part for "equivalent size" for it to be less theoretical, but formulas would be applicable to any organism with overlapping generations). The g_i values across classes are assumed to be the same for all populations, and the total stand basal area *G(t)* of a population is calculated as:

$$G(t) = \sum_{i=1}^{k} N_i(t).g_i$$

Then, at each new reproductive event, all P_{ii+1}'s and f_i's are multiplied by a reduction coefficient $\alpha(t)$ ($0 \leq \alpha(t) \leq 1$), which is dynamically computed by the program in order that $G(t)$ grows logistically to an equilibrium value (G_{eq}, i.e. equivalent carrying capacity) specified by the user for each population.

More precisely, denote $\mathbf{S}(t)$ the vector containing the numbers of individuals that remain in each size class and $\mathbf{Y}(t)$ the vector of the numbers of individuals that would be potentially recruited in each size class from *t* to *t*+1 if there was no regulation during the interval *t* to *t*+1. $\mathbf{S}(t) = (S_1(t), S_2(t), \ldots, S_k(t))$, where $S_i(t)$ is the number of potential surviving individuals in size class *i* during the interval *t* to *t*+1. The $S_i(t)$ values are computed with the formula $S_i(t) = P_{ii}.N_i(t)$. Then, the program computes the corresponding stand basal area of these individuals:

$G(\mathbf{S}(t)) = \sum_{i=1}^{k} G_i.S_i(t)$. For the $\mathbf{Y}(t) = (Y_1(t), Y_2(t), ..., Y_k(t))$, $Y_1(t) = \sum_{i=1}^{k} f_i.N_i(t)$ is the number of individuals that are potentially recruited in the first size class, $Y_{i+1}(t) = P_{ii+1}.N_i(t)$ is the number of individuals that potentially pass from size class *i* to size class *i*+1 and the corresponding stand basal area is $G(\mathbf{Y}(t)) = \sum_{i=1}^{k} G_i.Y_i(t)$, with $\mathbf{Y}(t) = (Y_1(t), Y_2(t), \ldots, Y_k(t))$.

$\mathbf{Y}'(t) = \alpha(t)$. $\mathbf{Y}(t)$ correspond to the vector of recruited individuals after density-dependent regulation. $\alpha(t)$ is a numerical reduction coefficient ($0 \leq \alpha(t) \leq 1$) computed for each population following this rationale:

if there was no density regulation, the increase in stand basal area would be $\Delta G(t) = G(\mathbf{S}(t)) + G(\mathbf{Y}(t)) - G(t)$. However, we introduce a density-dependent regulation by stating that the true increase of the stand basal area is $\Delta G'(t) = \Delta G(t).(1 - \frac{G(t)}{G_{eq}})$, with G_{eq} the equilibrium capacity for the population. This is performed by setting the $\alpha(t)$ value to $\alpha(t) = \frac{(G(t) + \Delta G'(t) - G(\mathrm{S}(t)))}{G(\mathrm{Y}(t))}$.

The same procedure as above is also applied to search for an initial demographic equilibrium at the beginning of the simulations when clicking on the "**Compute**" option of the *Demography Tab,* assuming that all populations are isolated, not submitted to selection, and with a deterministic offspring number (see various possible outcomes of this initial search in *3.5.1.1*). Once simulations have started, the same rules apply but accounting also for possible migration and selection effects on population sizes. Similarly, if the number of initial individuals is chosen with the "**By hand**" option, regulation steps will start acting based on the initial number of individuals in each population and in each class.

In practice, for the population growth to follow a logistic function, the growth rates (i.e. female potential fecundity values) need to be relatively small from one generation to the other. Thus, for a small number of classes with contrasted and fairly large fecundity values in some cases, the *G(t)* values could rapidly and strongly exceed ***K*** values. This effect is exacerbated in the case of high migration rates since the density-dependent **regulation process A2** only depends on the targeted adult population and class sizes and their potential fecundity. In those cases, the additional density dependent **regulation process B** applies as in non-overlapping generations (see **Figure 3**), and prevents *G(t)* values to exceed the ***K*** values by more than 15%, similarly to what is done for non-overlapping generations (see *3.5.5.4*).

For example, in Austerlitz *et al*. (2000), the following values for the same parameters in a similar demographic model were chosen : $k = 25$, $f = 250$, $G_{eq} = 91912$, $P_{11} = 0$, $P_{ii} = 0.54$ for $2 \leq i \leq 24$, $P_{25,25} = 0.98$, $P_{12} = 0.1$, $P_{i,i+1} = 0.4$ for $2 \leq i \leq 24$, $G_1 = 0.01$, $G_i = 0.5$ for $2 \leq i \leq 8$, $G_i = 1$ for $9 \leq i \leq 16$, $G_i = 2$ for $17 \leq i \leq 24$, $G_{25} = 3$. These values are arbitrary, they were adjusted so that the individuals had the following characteristics: the number of classes (k) is 25, so individuals can never reproduce before the age of 25, and potential fecundity is high in the absence of density dependence ($f = 250$). These parameters were chosen so that at equilibrium, an individual reaches the adult class on average at 50 years and that the average age of the individuals in the adult class is 100 years. An important point of this model is that, for a newly founded population, due to low density, individuals reaching the adult class are much younger.

3.5.5.6 Male and female fecundity and link to the application of selection

At each reproductive step, offspring numbers are produced following hard or soft selection rules which are first based on class and population potential female fecundity values (see **3.4.2** and **3.4.4**). Over many reproductive events in a neutral model, female realized fecundities are adjusted so that the populations grow logistically toward an equilibrium value, using the various density-dependence regulation steps detailed in parts *3.5.5.4 and 3.5.5.5* (see also **Figure 3**). If there is "**no selection**" (option in the *Selection Tab*), all individuals have the same fitness and offspring distribution is only affected by density-dependent regulations in the demographic model. In scenarios with selection, these female realized fecundities also depends on their fitness values, and for newly recruited individuals in each class and population, male and female parents (and thus gametes within them) are drawn from their relative contribution to the parental populations based on their realized fecundity (see *3.4.4.1*).

Regarding the male potential fecundity, we assumed that there is *a priori* no male gametes limitation overall, *i.e.* the dispersal rates and the fitness values of individuals only affect the relative proportions of male gametes originating from different possible populations and

classes. This might be realistic for many plant species or marine organisms with relatively large census sizes and gamete production “where the female fertility is rarely limited by the number of males in the populations” (Charlesworth and Charlesworth 2010), but might not be for other biological models with low numbers in age classes or populations. Thus the relative contribution in male gametes into population *j* from a male *k* from population *i* with a fitness f_k and belonging to a size class *c* of male potential fecundity f_c will be $p_{ij} \times f_k \times f_c$, where p_{ij} is the male migration rate from population *i* to population *j* (see *3.5.5.2* and **Figure 3.4.4.1**). Depending on male gamete flow and individuals’ fitness values, these proportions can be very low.

Male gamete limitation or absence will only occur if all individuals, in a given population and in all populations connected to this population through male-gamete flow, have a male fecundity equal to zero. In that case, no sexual offspring will be produced in this population, even if there are individuals with non-zero female fertility in it. A warning message indicating “No male gamete available in population *p*, [p being the population identifier in the run], so no sexual reproduction” will be put in the result file. This can occur for example in a model where age classes are simulated, and with individuals in younger classes being infertile so that they need to undergo several cycles before reaching a class where individuals can produce gametes.

3.5.5.7 Clonality

Clonal reproduction is modelled with the following assumptions (see also **Figure 3.4.4.1**):

- There is no new variants by mutation for individuals resulting from clonal reproduction.
- In a scenario with overlapping generations, individuals from clonal reproduction are attributed to a specific size class set by the user.
- The clonal offspring are always located in the populations of their parents (no clonal migration), which is a realistic assumption in many species. However, since clonal individuals mate with local individuals once they become fertile, their lineage can contribute to gene flow between populations through offspring produced via sexual reproduction in the following reproductive events.

The possibility of simulating scenarios with both overlapping generations and clonality allows for a large range of mating systems, including some form of clonality, which can vary across age classes for example. However, clonal fecundity is considered to be a characteristic of the biological system simulated in gMetapop, and thus does not vary across populations. Practically however, one way to make the proportion of clonal fecundity vary among populations is to allow different female potential fecundity among populations. Consequently, clonal reproductive rate would also vary for a given clonal fecundity, but the constraint here would be that the total fecundity would also be different across populations (see **tests 5.8** for examples of scenarios with clonal reproduction).

NB.3.5-h: Please note that in the case of clonal reproduction only, simulated scenarios need to account for the fact that no mutation is possible from parent to clonal offspring, which is likely to affect patterns of diversity and evolutionary potential in scenarios with large clonal fecundities and large number of generations.

3.6 Phenotype=f(Genotype) Tab

When selection acts on a phenotypic trait, this Tab allows setting phenotypic values as the sum of theoretical genotypic and environmental values. Genotypic values are computed as by the sum across QTL of initial (and theoretical) additive allele values at each QTL (see more details in **3.6.6**).

3.6.1 Cumulative variance across QTL for generating additive values

This cumulative variance value is set by default at 1 per QTL with equal weights. It can be changed in the GUI, and QTL weight values can be changed according to different proposed distributions, so that the genotypic values across QTL depend on the distribution of relative weights. These values, and the value chosen for the environmental variance, will condition initial heritability values (see also **3.6.5** below and **NB.5.7-a**)

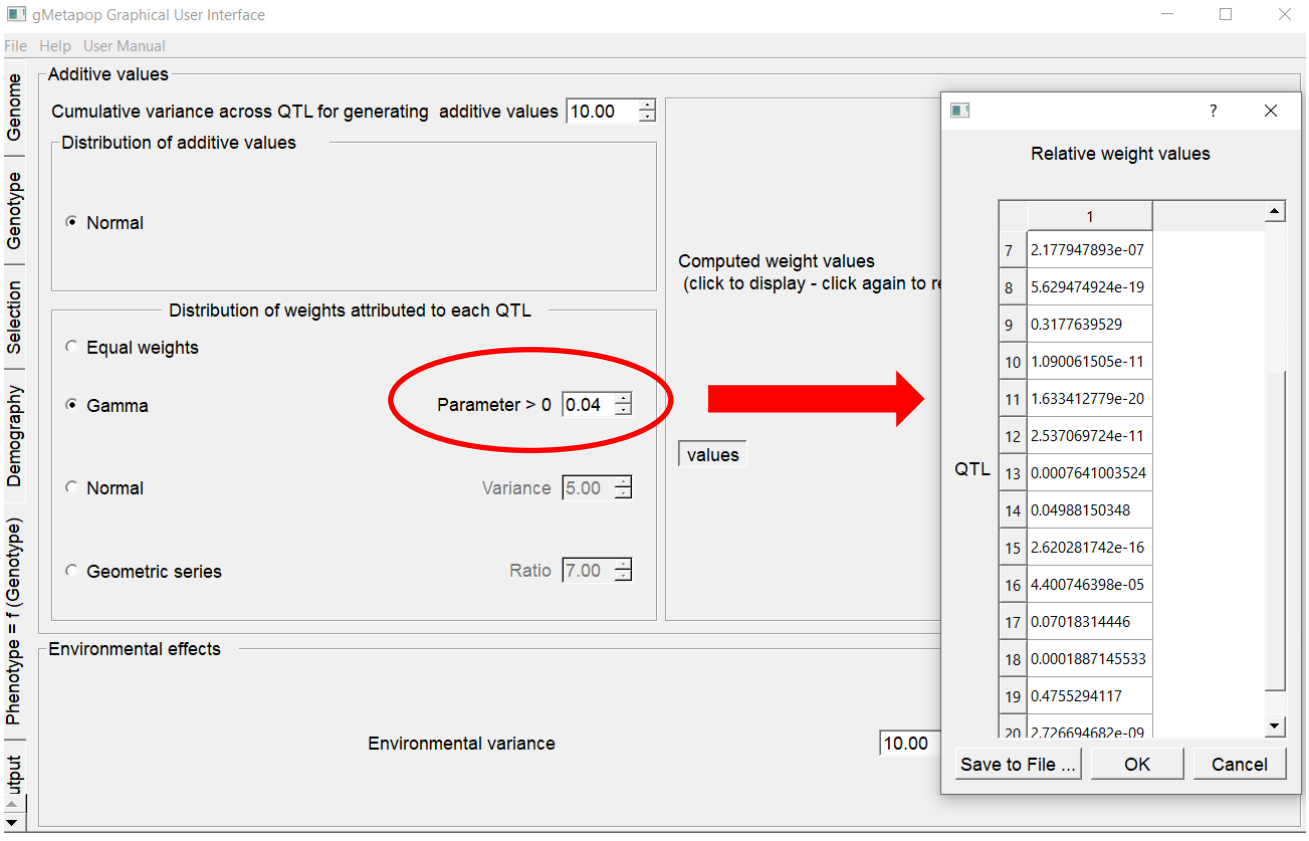
3.6.2 Distribution of additive values

Additive values for all potential alleles at each QTL are theoretical values drawn from zero-centered Gaussian distributions using the variances per QTL derived above (but the produced conf.txt can later be edited for any other case, see **NB.3.2-a**). Relative contributions of QTL can be modulated from the different weight options available (see after).

3.6.3 Distribution of weights attributed to each QTL

Weights can be equal across QTL (default), or drawn from various distributions (Gamma, Gaussian or Geometric Series) with parameter values set by the user (**Figure 3.6**). For example, drawing weights from a Gamma distribution with a parameter value below 0.5 will greatly increase the variance among QTL weights.

Figure 3.6: ***Phenotype=f(Genotype) Tab*** example in a scenario where QTL have unequal relative weights, which are drawn from a Gamma distribution.



3.6.4 Computed weight values (click to display – click again to redraw)

Relative weights (sum = 1) are automatically redrawn from the chosen distribution each time the "**values**" window is clicked.

It is not possible to edit these values in version 1.0.0 of the GUI, but it allows getting an overview when searching for a range of values with different initial distributions.

3.6.5 Environmental Effects

Environmental effects are added to the individuals' genotypic values to obtain phenotypic values. They are drawn from a zero-centered Gaussian distribution, with a variance which conditions, along with the initial cumulative additive variance chosen, the range of estimated heritability values across populations that will be computed at the start of simulations in the CORE (see more in **3.6.6** below, and also **Tests 7** examples and **NB.5.7-a**). By default, both cumulative initial additive variance and environmental variance are set to values based on the number of QTL. This yields, most of the time, a medium range (around 0.5) for the initial heritability values before applying selection. Those values can be changed if a very large number of loci is considered, or for a different initial heritability magnitude (see **3.6.1** above).

3.6.6 More information on the quantitative genetic model

Following the classical quantitative genetic approach (Fisher 1918; Walsh 2007), the phenotypic value ***Z*** of each individual is decomposed into a genotypic value ***G*** and an environmental value ***E*** (or effect, see also **3.6.5** above):

$$Z = G + E$$

Genotypic values are derived from a simple theoretical model of additive action of allele values at each QTL, across QTL contributing to the trait**. Additive values** can be considered as **theoretical allele values** attributed to all potential alleles at QTL, which can be chosen with different distributions in mind. These values should not be confused with statistical "**additive allele effects**" which can be computed *a posteriori* at each reproductive event, and depend on allele frequencies or genotypes conditional probabilities (see Lynch and Walsh 1998, and *3.7.6.3* Additive variances). Given previously drawn additive values and a number of QTL chosen in the *Genome Tab*, the genotypic value (***G***) of a given individual that carries the alleles j_1 and j_2 at locus i is then computed as: $\boldsymbol{G} = \sum_{i=1}^{n}(a_{ij1} + a_{ij2})$**,** for QTL with equal weights. The corresponding phenotypic value, ***Z***, is submitted to the normalizing fitness function that is used for modelling stabilizing selection (see *3.4.4.2*). Although theoretical, additive values and QTL weights can be calibrated based on available literature, for instance by using QTL analyses results, which provide effect size distributions across QTL (*e.g.* Johnson and Barton 2005; Albert *et al*. 2007; Feng *et al*. 2020). Similarly, values for initial cumulative additive variance and environmental variance can be set so that they are consistent with heritability or genetic basis information found in the scientific publications. It should be noted here that the range of theoretical additive values are initially set across the maximum number of potential alleles chosen for each QTL and will not change across simulations. However, the evolutionary stochasticity in the system can still be very large, and depends on this number of potential alleles, their initial number (which can be much lower), other initial conditions, and on the interaction between all processes simulated and their corresponding parameter values:

- mutation (e.g. what will be the effect of a new allele having this particular initial additive value compared to the optimum value?),
- selection (e.g. how will selection act on such new alleles?),
- migration (e.g. how the genotypic composition of migrants will be affected by selection parameters of their destination population?),
- genetic drift (e.g. how easily will those new alleles be lost?), etc.…

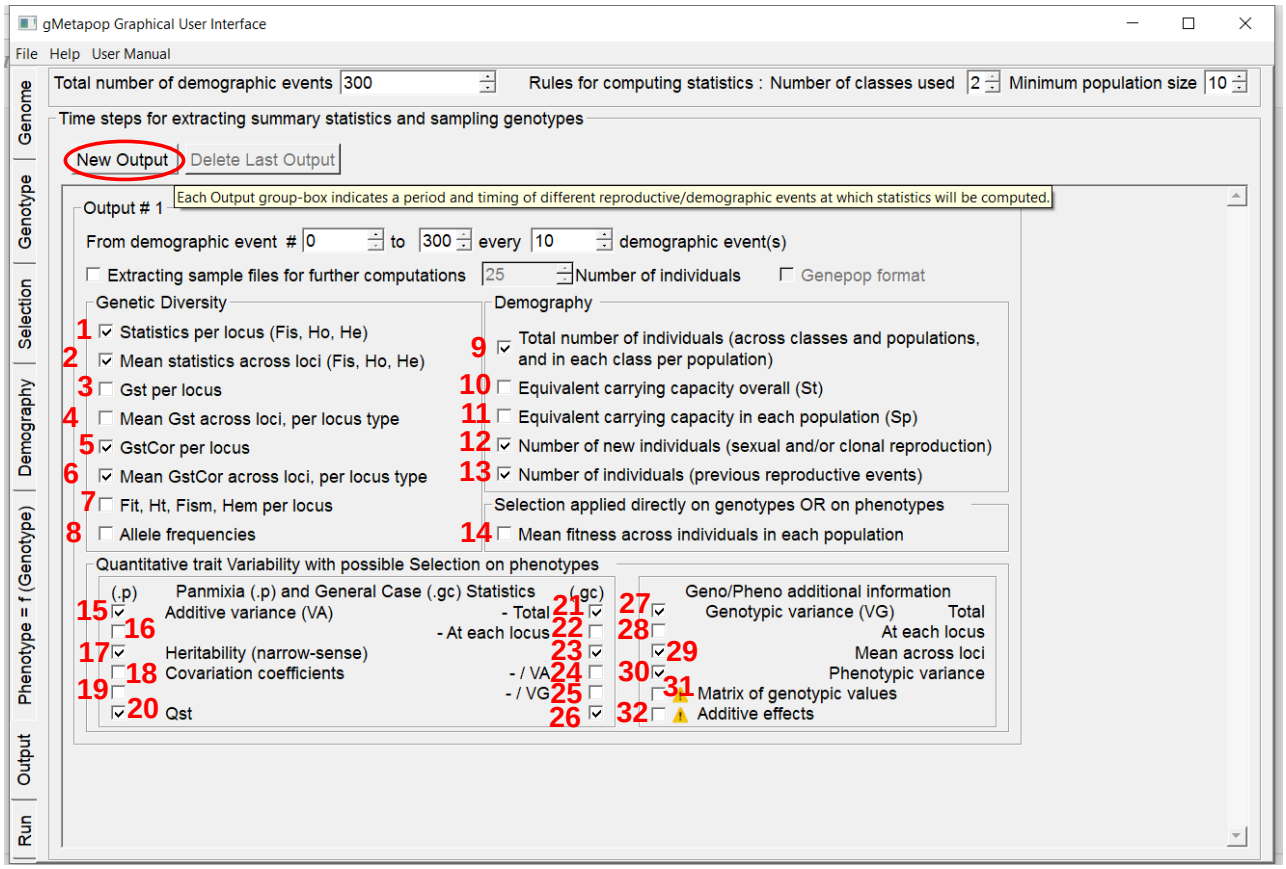
Dominance or epistasis effects are not implemented for building genotypic values in this version of the program. However, with the normalizing fitness function used for modelling stabilizing selection, allele contributions to the fitness values of individuals $W_i(Z)$ very much depend on additive values at other alleles and QTL, and can thus generate strong epistasis for fitness despite their additive mode of action when defining the genotypes (Whitlock *et al.* 1995, see their Figure 2).

3.7 Output Tab

This Tab creates the *type.txt* file that lists the summary statistics requested during the simulation. These statistics can be computed for all individuals present in each population after each set of reproductive or demographic events, and they are printed in different results output files in the working folder (see **3.8** *Run Tab* below, and **Chapter 4** for details on result files).

NB.3.7-a: Precisely, we use "reproductive event(s)" in the *Output Tab* in a life cycle with non-overlapping generations, and "Demographic events(s)" in the case of overlapping generations where both demographic events (i.e. survival or transition across classes) and reproductive events for individuals in different classes can occur (see more details in **Figure 3**, **Figure 3.4.4.1** and **Figure 3.5.1.1**). **For simplicity, we often use "reproductive event" in general for a set of demographic or reproductive event(s) in both type of life-cycles.**

Figure 3.7: Overview of *Output Tab*, numbers indicating various options (i.e. check boxes) and refer to in the first column of **Table 3.7.5** below.



NB.3.7-b: When requesting the computation of genotypic values or additive effects (checkbox **31** and **32** in **Figure 3.7**), please note that much additional computation time might be needed, and large output files will be produced depending for example on the initial number of alleles chosen (see various estimates of simulation runtimes across **Chapter 5** tutorials). We thus recommend performing exploratory tests, and accounting for the warning messages issued when requesting of genotypic values and additive effects matrices.

3.7.1 Total number of reproductive (or demographic) events

In gMetapop, the numbering of each set of reproductive or demographic events starts at 0 (“generation” counter) for each new simulation run.

3.7.2 Rules for computing statistics

Number of classes used: indicates how many classes are used when computing the statistics requested, starting from the class with the highest index then using decreasing indices.

Minimum population size: minimum number of individuals that need to be present in a population for statistics to be computed.

3.7.3 Time steps for extracting statistics and sampling genotypes

Clicking one Output group-box (**1** to **32** in **Figure 3.7**) activates the computation of various statistics during particular periods and time steps across the successive reproductive or demographic events during simulations. These time steps are indicated in a first column named “Generation” in all output files. Different Output Boxes (click “**New Output”** circled in **Figure 3.7**) will generate different result files corresponding to different sets of demographic or reproductive events (defined by the index **“Y”** in the files **resX_per_gen_Y.t**xt for example, “**X”** being the index of the replicate number, and see **4.4** for a list of all possible output files).

Please note that if overlapping time steps are requested in two different Output group-boxes, it is the second one that documents the corresponding “generations”.

3.7.4 Extracting sample files for further computation

Genotypes can be sampled at random across requested classes at the same time steps than those used for computing statistics (see examples in **Chapter 5** indicated in **Table 4.4.1**). They are saved into simple text files formatted for loading into R or other softwares, and if requested into the format for the Genepop software (Rousset 2008, https://genepop.curtin.edu.au/).

If the number of individuals is smaller than the one requested for extracting samples, all individuals are sampled. Therefore, when working with small population sizes, or when using a Poisson distribution for drawing offspring numbers (which might increase the genetic drift), we recommend that the user follows the number of individuals with the Demography statistics across generations (choose check boxes **9** to **13** in **Figure 3.7** and see **Table 3.7.5** below).

3.7.5 List of available statistics

Depending on the type of loci and scenario simulated, different statistics are available in the *Output Tab* (**Table 3.7.5** below**,** see also more details and definitions of statistics in **Table 4.4.2** in Chapter 4).

Table 3.7.5 List and abbreviations for summary statistics available from the *Output Tab*. See also **Table 4.4.2** for more information on the statistics definitions, and on their legend in Default plots.

Check-box clicked (as in Fig. 3.7)	**Nu-clear locus**	**Cyto-plasmic locus**	**Keywords in type.txt**	**Statistic name in result files*** **	**Printed in resX_per_gen_Y.txt**	**Printed in resX_per_pop_Y.txt**	**Default plots@**
Genetic Diversity							
1	**X** (any type)	**X***	Ho He Fis	Ho_l…, He_l…, He_cl…# Fis_l…,		**X**	
2	**X** (any type)	NA&	Div	Ho, He, Fis,		**X**	**X (Hom, Hem, Fism@@)**
2	**X** (at least 2 types of loci)	NA	Div	Ho_cod1/2, He_cod1/2 Ho_sel, He_sel Ho_qtl, He_qtl		**X**	
3	**X**	**X**	Gst	Gst_l…#, Gst_cl…#	**X**		**X**
4	**X****	NA	LocGst, cod1(/2)Gst, SelGst	Gstm$, Gstcod1/2, Gstsel$$ or Gstqtl$$	**X**		**X**
5	**X****	X	Gstcor	Gstcor_l…, Gstcor_cl…	**X**		**X**
6	**X****	NA	LocGstCor, cod1(/2)GstCor, SelGstCor	Gstcorm$, Gstcorcod1/2, Gstcorsel$$ or Gstcorqtl$$	**X**		**X**
7	**X**	**X***	Hem,Ht, Fit,Fism	Hem_l..., Hem_cl... Ht_l…, Ht_cl… Fit_l… , Fism_l…	**X**		**X**

8	**X**	**X**	Frequencies[§]	f.l1a1… f_cl1a1…	**§**	**§**	
Demography							
9	NA	NA	Nt_Np	Nt Np	**X**	**X**	
10	NA	NA	St	St	**X**		
11	NA	NA	Sp	Sp		**X**	
12	NA	NA	demo_new	Nsext, Nclot Nsexp, Nclop	**X**	**X**	
13	NA	NA	demo_prev	Nremt, Nremp	**X**	**X**	
Selection applied on phenotypes OR directly on genotypes [§§]							
14	**X**	NA	Fitness	mean_fitness		**X**	**X**
Quantitative trait Variablity with possible Selection on phenotypes							
15	NA	NA	VA.p.tot	VA.p.tot	**X**		**X**
16	NA	NA	VA.p.qtl	VA.p.qtl1…		**X**	
17	NA	NA	h2.p	h2.pan		**X**	**X**
18	NA	NA	thetaWp.g	mthetaW.p.g, thetaW.p.g	**X**	**X**	**X**
19	NA	NA	thetaWp.t	mthetaW.p.t, thetaW.p.t	**X**	**X**	**X**
20	NA	NA	Qst.p	Qst.p	**X**		**X**
21 to 26	NA	NA	Same than 15 to 20 with ".gc" instead of ".p"	Same than 15 to 20 with ".gc" instead of ".p" (/ ".pan" for h2)	**X**	**X**	**X**
27	X		VG	mVG, VG	**X**	**X**	**X**
28	X		Vg.qtl	Vg.qtl1…		**X**	
29	X		Vg	mVg, Vg	**X**	**X**	**X**
30	X		VP	VP		**X**	**X** (**mVp**[@@])
31	X		gVal	G.qtl1…	**&**	**&**	
32	X		$Ai^{\#\#}$	ai.q1.a1…, aei.q1.a1…, ai.gc.q1a1…	**§§**	**§§**	

[@] When indicated in this column, the statistic names used in the result files (column 5) are also used in the legend of default plots (see **3.9**).

[@@] In the **plot.perGen.png/pdf** plot, **Hom (resp. Hem**, **Fism**, and **mVP)** is the computed mean across populations of the per population statistics **Ho (resp. He**, **Fis**, and **VP),** which are printed in the **resX_per_pop_Y.txt** output file). In the case of only one population, the statistics **Ho, He, Fis** and **VP** are used in the plot for that population.

***** Except Fis and Ho statistics; ****** Gst and Gstcor estimates are provided for more than 2 populations.

******* Here the prefix in the names of the statistics indicates the type of statistic while the suffix is the type of locus.

& mean diversity statistics are only computed for nuclear loci.

$ mean Gst/Gstcor statistics across all nuclear loci considered.

$$ Gstsel and Gstcorsel are for loci potentially under selection on genotypes, Gstqtl and Gstcorqtl are for QTL in scenarios with possible phenotypic selection.

§ Allele frequencies are printed in files **resX_freq_Y.txt**, with X being the replicate number and Y being the set of reproductive events considered (i.e. one set with particular time steps for each Output window in the *Run Tab*).

"-cl" in a statistic indicates that it is for a cytoplasmic locus, "-l" that it is for a nuclear locus, "ax" is the allele number suffix for frequencies, additive and aei effects, x being the number.

"Ai" (in the type.txt file) generates the computation of additive allele effects either under the assumption of panmixia ("ai.") or in the general case ("ai.gc", but see **NB.3.7-c** below), and also of the average excess effects ("aei."), see **3.7.6.3.** In order to optimize computation time (in the general case only, and for more than one population in ver 1.0.0 of the software), if more than 1000*nb.pop individuals are present in total (with nb.pop being the number of populations), a random sampling of 1000*nb.pop individuals is used to decrease computation time. In the case of only one population, all individuals are used in computations.
& the genotypic values G.qtl1, G.qtl2 etc… are printed for the last time step of the last set of reproductive events in the **gVal.merge.pop.txt** (working file).
§§ "ai.", "ai.gc" and "aei." statistics are printed in the **resX_ai_Y.txt** file, X being the replicate number and Y the set of reproductive events.

NB.3.7-c: Please note that if additive effects are the only statistics requested in the quantitative window of the *Output Tab*, these are provided under the hypothesis of panmixia only.

3.7.6 More information on summary statistics

3.7.6.1 Genetic Diversity within and between populations

For more details on diversity statistics within populations and between populations, see also **Table 4.4.2** that includes formula and references. See also the formula of different estimates of differentiation statistics and correlations between them derived from simulations scenarios in chapter 5 (**Tests 5.3**)

3.7.6.2 Population dynamics and demography statistics

Both equivalent carrying capacity (***K***) and number of individuals' (N) statistics can be requested from the *Output Tab* either per generation or per population/generation. In the case of non-overlapping generations, K and N are equivalent, while in the case of overlapping generations, *K* depends on both the number of individuals and the equivalent size of the classes (see **3.5.3** in the *Demography Tab* and tests **5.9** and **5.10**). For example with 5 classes, individuals belonging to classes 1 and 2 have by default an equivalent class size below 1, which means that their weight (in terms of space occupancy for example) is relatively lower than those in classes with higher values, and thus these individuals contribute less to the total K.

In addition to the demography statistics printed in the **resX_per_gen_Y.txt** and **resX_per_pop_Y.txt** output files, more demography statistics are printed in separate **resX**.txt text files ("**X"** being the replicate number). These files can be useful to understand the warnings corresponding to the different cases of initial demographic conditions (see **Figure 3.5.1.1**). Intended initially as working files describing cycles, it is provided here as it prooved useful in models with complex age-class structure to better understand particular aspects of population and class dynamics. Note that the "**res**" prefix is fixed for all result files (see *3.8.7.1*).

3.7.6.3 Additive Variances

Additive variances depend upon statistical **additive allele effects** and average excess effects (which are the same in case of random mating among individuals), themselves depending on allele frequencies and on conditional probabilities within populations of each genotype, given one allele or the other. Following Lynch and Walsh (1998, p. 65-79 in their chapter 4, refer to formula 4.16a, 4.16b, 4.23a, and 4.23b), these effects and resulting additive variances can be estimated with the hypothesis that mating is random among individuals, whatever the number of alleles. However, with more than two alleles and when mating is not random among individuals (for example in case of strong selection, or when there is assortative mating), we need to use multivariate regressions, which have been used in gMetapop to estimate statistical additive effects in the general case (see equation 4.19, chapter 4 in Lynch and Walsh 1998). In the case that these regressions do not have a solution, missing values are returned for additive variance estimates, and the user needs to

decide whether random mating estimates can be considered as being informative in the tested scenarios, and/or as valid approximations. See examples provided in **tests 5.7** and **tests 5.10.2** in Chapter 5.

3.8 Run Tab

This tab creates (or selects from previous runs):

1) The working folder for all input and output files,

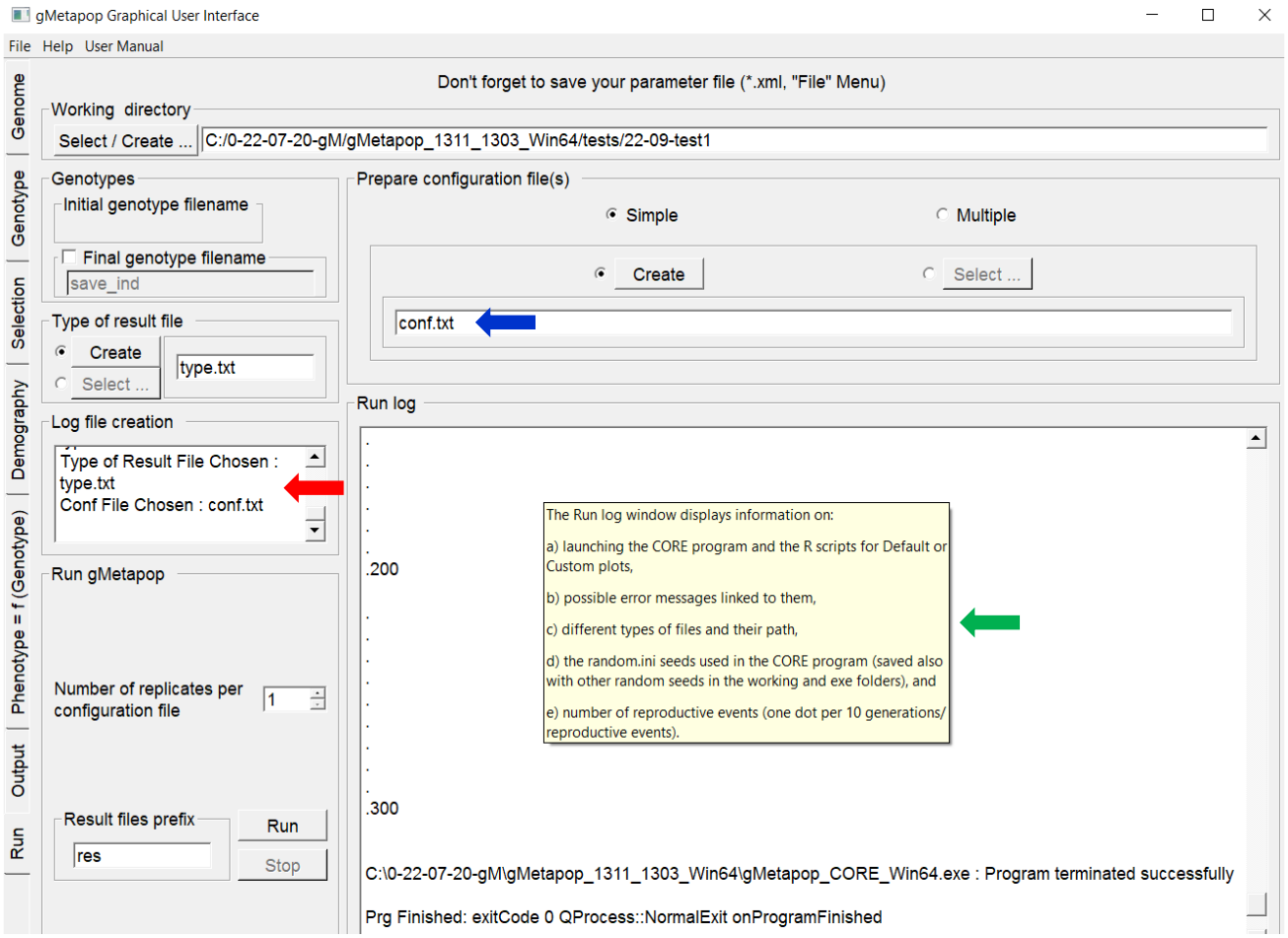
2) **The "type.txt" and "conf.txt" files needed to launch gMetapop** (see also **Figure 1.4** and part *3.8.7.3*).

This *Run Tab* also indicates the file with individual genotypes at the start of simulations (if loaded beforehand in the *Genotype Tab*). In this case, and in other cases where previous *param* files are being loaded into the GUI, all the saved paths may need to be checked before launching gMetapop since the folder organization and working folder may have been changed (see **NB.5.1-c**).

3.8.1 Working directory

The working directory is either created or selected from existing directories. It contains all necessary files to run the simulations (either previously created or loaded into the GUI and copied here). Under a Windows OS or a Linux OS with a graphical environment, working directories can be created directly in the OS graphical environment in the process of choosing them.

Figure 3.8: ***Run Tab*** example showing the default configuration name that can be changed (red arrow), the information in the log file window (red arrow), and the tooltip for the Run log window (green arrow).



3.8.2 Saving the parameter *.xml file

As indicated in the *Run Tab*, we recommend saving the *param* file before launching a simulation. This will allow re-loading all chosen values for the different parameters after closing the gMetapop_GUI. A warning will be issued otherwise. However, this is not necessary for exploratory runs, since these values are also saved when creating both conf.txt and type.txt default files, and these can be saved in different folders (or renamed in the same working folder) without having each time to save the corresponding *param* xml file.

3.8.3 Genotypes

Initial genotype filename: the name of the file loaded into the *Genotype Tab* is reported here with the corresponding path. Some of its format characteristics will be checked before starting the simulations.

Final genotype filename (**save_ind** by default) is the "prefix" name of the genotype file, which is printed at the last generation of each replicate of a simulation run. For example, for a simulation with 2 replicates, if the final genotype files are requested, save_ind**1**.txt and save_ind**2**.txt (i.e. suffix "**1**" and "**2**" for replicate "**1**" and "**2**" respectively, etc…) are created in the working folder.

3.8.4 Type of result file

Create: **a new 'type of result' file (« type.txt ») is created in the working directory. This file stores all information on the statistics chosen in the *Output Tab***. See multiple multiple examples of "type.txt" files across the tutorials of **Chapter 5**.

Select: **the type of result file is chosen from previously created files.** It is copied in the current working folder when the simulation starts (Run button), and it can have a different name. The name of the file is indicated in the corresponding *Run Tab* window.

If it is selected, it will be disconnected from checkboxes clicked in the *Output Tab* (see **NB.3.8.6**). Although "type.txt" is the default name, any other text file name can be used. See **Table 3.7.5** for an exhaustive list of possible statistics requests from the *Output Tab*.

NB.3.8.4: In the "Multiple" option for preparing Configuration Files, the "type.txt" file will be created above the different simu* folders (see **3.8.6** below). It can also be chosen from another folde

3.8.5 Log file creation window

Indicates the creation or copy of the "type.txt" and "conf.txt" files in the working folder (red arrow in **Figure 3.8**), with the time elapsed since the saving of the *param* xml file if needed.

3.8.6 Prepare configuration files

Simple: only one conf.txt file is used during simulations.

Multiple: Multiple configuration files are created in different *simu** folders with different initial random seeds (e.g. different draws used for initial allele frequencies, or initial additive effects). They can serve as replicates with more stochasticity due to different initial conf.txt files, and also as a basis for comparing scenarios across a range of some parameter values (see **Tests 3** for an example on migration rates).

Create: **a new configuration file (« conf.txt ») is created in the working directory, with all the parameter values chosen in the GUI.** This file can be further edited for particular values without changing its format (see **NB.3.2-a**). See also the many "conf.txt" examples in the tutorials provided in **Chapter 5**.

NB.3.8.6-a: WARNING: the name of the configuration file that appears in the window under the "**Select**" or "**Create**" buttons (blue arrow in **Figure 3.8**) can be changed by the user. It is the name used by the software to create the configuration file ("**conf.txt**" by default).

Select: **the configuration file is chosen from previously created files.** It is copied in the current working folder when the simulation starts (Run button), and it can have a different name. The name of the file is indicated in the corresponding *Run Tab* window.

NB.3.8.6-b: In the "Select" cases (for both the conf.txt and the type.txt files), when clicking on the "Run" button (see below), the file **run_gMetapopCore.bat (or *.sh** under linux) is using the selected "conf.txt" and/or "type.txt" files, which are thus disconnected from any default or modified configuration values previously entered into the GUI Tabs, see also 3.8.7.3 below.

3.8.7 Run gMetapop_CORE

3.8.7.1 Directly after creating a new param file

"Number of replicates per configuration file": several replicates using the same initial "conf.txt" can be performed. Simply choose a number of replicates different than the default of **"1"**.

NB.3.8.7-a: In version 1.0.0, the "**Result files prefix**" has been fixed to **"res"** for all result files, but could be easily modified by the user afterwards.

Then the user needs to click on the "**Run**" button to start simulations. This action generates the creation of a file called **run_gMetapopCore.bat** (or ***.sh** under linux), which includes the

command lines launching the CORE Program. Simulations can be easily terminated any time by clicking on the "**Stop"** button.

NB.3.8.7-b: If the gMetapop_GUI application is simply closed, gMetapop_CORE can still run in the background depending on your OS, and the process may have to be cancelled.

3.8.7.2 After re-opening a previous param file

Similarly, after launching the GUI application, a simulation can be run using a previous *param* xml file. We suggest in this case to quickly (re-)check parameter values, working folder (in case of changes in the folder organization), summary statistics requested in the *Output Tab*, number of configuration files requested, and number of replicates wanted.

3.8.7.3 From the Run Tab, using previous conf.txt and type.txt files

Simulations can be directly launched from the *Run Tab* after opening gMetapop_GUI. First follow or check steps in **3.8.1, 3.8.4,** and **3.8.6** for selecting both a "type of result" file and a configuration file. Choose the number of replicates and click **Run**.

In this last case, the user can go directly to the *Run Tab* to launch gMetapop from the GUI.

NB.3.8.7-c: Please note that in the first release of gMetapop_GUI, launching simulation from the GUI by selecting a previous conf.txt is only compatible with using the "Simple" option in the "Prepare Configuration Files" window. In the case of the "Multiple" option, you can directly re-use the **run_gMetapopCore.bat** (or ***.sh** under linux) prepared previously (see **3.8.6**), or use your own series of command lines (See *3.8.7.4 below*). This may change in future versions of gMetapop. On the other hand, type or result (i.e. type.txt) files can be selected for both Simple and Multiple conf.txt options.

NB.3.8.7-d: Please note also that in this case (*3.8.7.3*), consistency and/or availability need to be verified between (1) the number of classes in the type.txt and the conf.txt files, and (2) the conf.txt file and the initial allele frequency file that should be in the working directory. Various error messages should be issued otherwise.

3.8.7.4 Using command lines

gMetapop_CORE can also be launched from command lines once the following files are available: the conf.txt, the type.txt, and either the initial corresponding allele frequency file (previously created or imported) or the initial individual genotypes (see above **3.8.7.3** and **4.3.1**).

Under Windows:

"C:\g*metapop.exe.path\gMetapop_CORE_Win64.exe" -C " conf.txt" -D " type.txt" -R " res1 [-W "\working.folder.path\"] [-I "geno-file.txt"] [-S " save_ind1.txt"]*

Under linux:

/gmetapop.exe.path/gMetapop_CORE_Linux64 -C "conf.txt" -D "type.txt" -R "res1" [-W "/working.folder.path/"] [-I "geno-file.txt"] [-S " save_ind1.txt"]

The arguments in [brackets] are optional, they correspond to simulations that start with an initial file of genotypes [-I], and/or that save a final file of genotypes [-S]. If the **–W "working.folder.path"** is provided, there is no need to provide a path for the "geno-file.txt". Alternatively, if the working folder path is not provided, the full path for the initial genotype file should be given. Many examples of command lines can be found in the tutorials of **Chapter 5** as they appear a) in **the run_gMetapopCore.bat** (or **run_gMetapopCore.sh**) files, and b) in the **Run log** window of the GUI. Those files are being created when clicking on the **Run** button of the *Run Tab* (see particular examples for launching simulations in batch with different conf.txt files in **Tests 5.3**). The basic syntax is also obtained by typing:

gMetapop_CORE_Win64[/linux64].exe under their OS respective command line windows.

3.8.8 Run log

The **Run log** window displays information on (green arrow in **Figure 3.8**):

a) launching the CORE program and the R scripts for Default or Custom plots,

b) possible error messages linked to them,

c) different types of files and their path,

d) the random.ini seeds used in the CORE program (saved also with other random seeds in the working and exe folders), and

e) number of reproductive events (one dot per 10 generations or reproductive events).

Therefore, the runtime can be estimated approximatively from shorter runs. Particular repetitive tests or steps are documented in the **resX_simul.log** files (see **Table 4.4.1**). Some contents of the **Run log** window also appear in the Shell windows of the different OS with additional information on filling up the *param* file (which is intended for developers). See also more in the Appendix online for trouble shooting and more information on possible errors while running the programs.

3.9 Plotting simulation results with R

In order to use R scripts from the GUI, you need first to install R (https://www.r-project.org/) and indicate to the GUI where R executables are located:

File/Choose R path in the GUI menu: chooses the main R folder (usually mentioning the R version, for example …R/R 3.5.0 or later versions). This needs to be done once only, unless you click again on "Choose R path" and indicate a wrong path, which prints an Error message.

The GUI allows for two types of plots:

3.9.1 Default plots

File/Default plot from the GUI menu launches the default R script **plot.gM.perGen.r**, and allows visualizing most statistics printed per generation in the "**res1_per_gen_1.txt**" file, as requested in the *Output Tab.* Two plot files are obtained: **plot.perGen.pdf/png**, in the working folder, the *.png plot being used for visualization from the GUI. The script also runs a R function for an additional **plot.perGen.quanti.pdf** file that shows other requested quantitative trait variability statistics with different range values. The most recent version of those scripts are available online (with histories of changes).

NB.3.9-a: Precisely, the **File/Default plot** GUI menu option first copies the original **plot.gM.perGen.r** R script provided in the *.exe folder, into the working folder. Then the copied script is launched with the user's R version, and can be further modified. Re-asking for a default plot first checks whether the user wants to use the working folder version (if the same name has been kept, otherwise the **Custom plot** option needs to be used), or if the original script needs to be copied and used again before being launched.

NB.3.9-b: Please note that the default plot script requires to read the configuration text file. For the script to work, this file needs to keep its default name conf.txt otherwise an explicit error message is issued. If the conf.txt name has been changed, it can either be renamed "conf.txt" before launching the default plot, or the "conf…" parameter values in the script functions can be changed by R users. The new script can either be run as default from the working folder, or its name can be changed and it can be run with the **File/Custom plot** option in the GUI menu.

These plots are very useful for getting a first overview of the simulations, and help adjusting parameter values (see **Figure 3.9** below for an example, and many examples of default plots in the test tutorials in **Chapter 5**).

If requested, the mean (across generations) of the total number of individuals and other demography statistics are also indicated, but only in the plot legend due to scale differences.

An exhaustive list of abbreviations used in the default plots legends for the different statistics and summarized range of parameter values is provided in column 3 of **Table 4.4.2** in **Chapter 4**, with more information at the start of the **plot.gM.perGen.r** script file for the different ways of using it (please check for the most recent version online as indicated by the creation date at the top of the script, with a list of eventual changes since the Release 1.0).

Figure 3.9: Example of default plot for a scenario with selection on phenotypes and overlapping generations used in **Tests 5.10.2** (scenario 4).

Insert Figure here from Pheno test 7

All default plots and their legends can easily be changed at the end of the **plot.gM.perGen.r** script. This allows a great flexibility of use. For example, changes can be done for line colours and types in the legend options, for the minimum and maximum scale values on the Y-axis, or for only plotting a subset of the statistics chosen for a better visualization and/or for comparative purposes among simulations. See also more details at the start of the script file for the different arguments and examples in tutorials of **Chapter 5**.

If any changes are being made:

a) The script can either be used as the new default plotting script, by copying it in the *.exe folder, giving it the same name **plot.gM.perGen.r**,

b) Or be re-used with a "**Custom plot**" option from the working folder (see below). In b), we recommend that you change the script name in the working folder since another use of the "**Default plot**" option (with answer "yes" to the question) will copy again the default R script with its original name, thus erasing your modified file of the same name.

c) See also the **plot.gM.perGen.zoom.r** script for zooming in between two particular reproductive events across the range that has been simulated (**NB.3.9-e**), which can be used with the Custom plot option in the **File Menu**.

If "**Multiple**" configuration files were used for simulations, the plots need to be made within each working simulation sub-folder **(…/simu1**;**…/simu2** etc…) that can be used as a new working folder from which the default (or custom) plot will be made. Click again on

"**select/create**" int the *Run Tab,* go to each simulation folder (e.g. …/simuX, "X" being the number of one of the scenarios, in **Tests 3**) before asking for "**File/Default plot**".

If only allele frequencies are requested in the *Output Tab*, no "**res1_per_gen_1.txt**" file is created so a default plot cannot be performed (. If only Demography statistics are requested, a warning is issued in the plot where only the legend is printed.

NB.3.9-c: Please note that default plots can only be produced if the **res1_per_gen_1.txt** and the conf.txt files of the corresponding simulation are present in the working folder. The **res1_per_gen_1.txt** file might be missing if none of the corresponding statistics were requested from the *Output Tab* (see **Table 3.7.5** for an exhaustive list). Particular quantitative statistics also requires that the **res1_per_pop_1.txt** be present in the folder, otherwise they are not plotted. In a simulation with only one population, the Ho, He and Fis mean statistics across loci from the **res1_per_pop_1.txt** are also plotted by default

If default plots cannot be produced, see first the generic error message that lists potential problems (e.g. names of working folder may have been changed and thus do not exist anymore, special characters appearing in the folder names that have to be removed, results files having been removed, presence of a bug in the script…), and see additional R error messages in the log widows. See also the online Appendix for more on trouble shooting and error management.

Default plots can be performed after the GUI has been closed, independently from the simulations, by just re-opening the GUI and choosing directly the working folder in the *Run Tab*, or by using directly the **plot.gM.perGen.r** script in command lines in the folder where results files have been created with Rscript.exe, for example:

C:\path.Rscript.exe\Rscript.exe plot.gM.perGen.R res1_per_gen_1.txt plot.name

3.9.2 Custom plots

File/Custom plot in the menu: opens a window where you need to type a simple command line as follows:

userRscript.r result.file name.chosen.for.the.plot [any other additional arguments listed in the userRscript.r provided, or developed by the users].

The R script can be any R script making use of at least two arguments: the *result.file* first and the *name.chosen.for.the.plot* as the second argument. The first argument is by convention any result file that is present in the working folder and further processed by the R script (e.g. **res1_freq_1.txt),** or any file used in the script that is named that way**,** but it can also be any character string used by the script if needed (see the post-treatment of tests *5.5.1.2*, **5.4** or **5.5.1**). The second argument *name.chosen.for.the.plot* is the character string used for naming the plot file without extension. This string is then searched by the GUI in the file list (from the working folder) in order to be visualized (a ".png" extension will be added automatically). In order to make the Custom plot option work, the only constraint is to include the exact following lines in your script, which opens and closes a *.png plot device:

png(filename=paste(name.chosen.for.the.plot,".png",sep=""),unit = "cm", width = 20, height = 18 , res = 600) (with values that can be changed)

…

dev.off()

This *name.chosen.for.the.plot* refering to the second argument can also be used in any other way in the script.

Additional arguments from the R functions in the script can be added to the command line.

The **plot.gM.perGen.r** script, once copied in the working folder, can also be used with the "**File/Custom plot**" option of the menu, with **res1_per_gen_1.txt** as *result.file* and any plot name for the second argument, as follows:

plot.gM.perGen.r res1_per_gen_1.txt *name.chosen.for.the.plot*

or it can be further modified (and renamed) and used in the same way. Other examples are provided in Chapter 5 (see **Tests 1**, **Tests 4** and **Tests 9**). Custom plots can be very useful in exploratory work for fast and direct visualisation of simulation results in different replicates, or when using files other than the "**res1_per_gen_1.txt**" file. Any additional necessary files for the custom plot need to be copied into the default working directory (e.g. popfinit2.txt for **Tests 5.1** custom plots).

NB.3.9-d: Being able to use the default plot script with the "Custom plot" option means that one can plot and visualize any other **resX_per_gen_Y.txt** file corresponding to a different replicate with different time steps (see **4.4** for more details on output files) using the command line indicated above.

NB.3.9-e: The default plot script has been extended to zoom between reproductive event **x** and **y** in a simulation using a total number of events higher than **y**: the corresponding script **plot.gM.perGen.zoom.r** is also located in the ***.exe** folder, and can be copied into working folders and used with the **"Custom plot**" option (see here, and procedure also explained at the start of the script file).

Custom plots can also be performed afterwards, independently to simulations, by just opening the GUI and choosing directly previous working folders in the *Run Tab*.

3.10 More information on the use of random number generators

3.10.1 Pseudorandom Number Generators (PRNG)

Two pseudorandom number generators (PRNG) are used in gMetapop:

The **PRNG 1** is the Mersenne Twister (Matsumoto and Nishimura 1998), that is used both in gMetapop_GUI (e.g. when drawing allele frequencies in a Dirichlet distribution), and in gMetapop_CORE (e.g. for all the sample files requested from the *Output Tab*). The PRNG 1 is also implemented once for the GUI from the GNU Scientific Library at https://www.gnu.org/software/gsl (referred to as **PRNG 1A**), and twice for the CORE from the C++ Standard Template Library (**PRNG 1B** and **PRNG 1C**). This type of PRNG has a pseudoperiod of $(2^{19937}-1)$.

The **PRNG 2** is used for most random processes in gMetapop_CORE. It is the PRNG that has been used historically in previous versions of the program (e.g. Machon *et al*. 2003), designed by L'Ecuyer (1988). This PRNG has a pseudo period of $\sim 2^{61}$ (L'Ecuyer 1988; L'Ecuyer 2017), which makes it still suitable for very large numbers of random draws.

When a simulation run is designed with the GUI and then launched and performed until the end with the CORE, the updated random seeds are stored in four files, in order to be used in new simulations.

- **Random.bin:** binary file, from **PRNG 1A**, used for all random draws needed in the GUI. When opening the GUI for the first time, with no Random.bin file present in the *.exe folder, the same initial random seeds will always be used. The updated Random.bin file is saved in both the *.exe and working folders *only when closing the GUI*.

NB.3.10-a: Please note that as a consequence, if more than one conf.txt file is created during a working session with the GUI, the only **Random.bin** accessible (that can be saved for replicating simulations, see below 3.10.2, optio b)), is the one present in the *.exe folder before the first conf.txt creation. If there is no previous Random.bin in this folder, the same initial random seeds used will be used for the first conf.txt creation (i.e.just after the opening of the GUI). Please note also that some conf.txt files do not call the PRNG 1A as they do not need any random draws.

NB.3.10-b: It is not possible to copy the binary Random.bin file from one OS to another, unlike the **Random.ini** file.

- **SeedRng.ini**: text file, from **PRNG 1B**, used by gMetapop_CORE when producing new offsprings. As for the Random.bin, the same seeds are used at the start of a new simulation, if the updated ones are not available in the *.exe folder. The **SeedRng_init.ini** file stores the initial seeds in the working folder.
- **SeedSampling.ini**: text file, from **PRNG 1C**, used if any samples are requested in the *Output Tab*. As for the Random.bin and the SeedRng.ini, the same initial seeds are used, if the updated ones are not available in the *.exe folder. The **SeedSampling_init.ini** file stores the initial seeds in the working folder.
- **Random.ini**: text file, from PRNG 2, is updated each time PRNG 2 is needed during the gMetapop_CORE simulated processes. The starting Random.ini seeds are also copied at the beginning of the **res1.txt** file in the working folder, printed in the "Run log" window at the start of a run, and saved in the **Random_init.ini** file in the working folder.

3.10.2 Replicating simulations or redoing the same simulations

Replicates can be launched from the GUI with the same conf.txt file, either using more than one replicate (see *3.8.7.1*), or with different conf.txt files by using the **"Multiple"** option in the *Run Tab* (see **3.8.6**), or by running the same simulation from the same conf.txt file. Since the simulations are stochastic, the results of each replicate will differ from each other, the variation among replicates depending on the relative importance of the stochastic and deterministic processes in the scenario chosen. Replicates are usually needed for drawing more general conclusions on the tested scenarios. Different random seed files are automatically used in the GUI when replicating simulations.

However, re-running the exact same simulation might be useful in complex demographic scenarios, if one wants to explore more deeply a particular series of events during the simulations, which has been identified from a default plot from exploratory runs for example. To that end, more summary statistics, or statistics at more frequent time steps, can be requested in the *Output Tab*. There are basically two main ways to redo the exact same simulation (i.e. with similar random draws):

a) Either simply re-use i) the same initial files, and ii) the same random seeds files needed for a simulation run by the CORE. For i), use the files previously created by the GUI in the working folder (i.e. a conf.txt, a type.txt, and an allele_frequency.txt file) or those corresponding to users' initial allele frequency or genotype files to be loaded. These files can be re-used from the same working folder directly from the *Run Tab* (or can be copied into a new working folder for redoing the simulation). For ii), use the **SeedRng_init.ini** and **Random_init.ini** files (and the **SeedSampling_init.ini** file if samples are extracted) that were generated in the working folder during the first CORE run, copy them in the *.exe folder and rename them **SeedRng.ini** and **Random.ini** (and **SeedSampling.ini**).

b) Or re-create the same conf.txt and type.txt files from the GUI before launching the CORE run. In order to do this, the same **Random.bin** file needs first to be used. This means that either the same initial seeds are used twice by the GUI (i.e. no Random.bin present both times before opening the GUI for creating the conf.txt file once), or that the Random.bin used for the first conf.txt creation (i.e. just after opening the GUI) is copied beforehand and is re-used the second time when reopening the GUI. Also, all *_init.ini files from the CORE run needs to be renamed as *.ini files in the *.exe folder the second time, as in a).

NB.3.10-c: please note that in cases a) and b), if the second simulation is performed in a different folder, the folder indicating the allele_frequency.txt path must be updated in the conf.txt file.

NB.3.10-d: Usually, if the time steps at which summary statistics are requested are different when redoing the simulations, most sample files will however be different, since a different number of those files will be generated. However as explained above, sample files are generated using a specific PRNG (PRNG-1C), so this will not affect the other processes.

NB.3.10-e: In scenarios with a quantitative trait under potential selection, the lines corresponding to the theoretical additive values for all possible alleles across QTL in the **conf.txt** file (which are randomly drawn) are stored in a separate "QTL_initial_value.txt" file, and so can be re-used in a new scenario where the same additive allelic values are chosen (see *4.3.1.2*).

4. CHAPTER 4 Input and Output files

4.1 Initial genotype files and conditions

All simulations start with initial genotypes which are drawn or chosen:

- Either from default GUI options for allele frequencies (see **3.3.2**) using *"***Create**" "**Allele frequencies***"*,
- Or loading an allele frequency file with *"***Load frequencies***"*,
- Or directly loading genotypes using *"***Load Genotypes***"* (see *Genotype Tab* for more details).

Correct formats for allele frequency files can be obtained in preliminary runs, either using the *allele_frequency.txt* file produced when creating the conf.txt file (i.e. "**Create"** option of the *Run Tab*), or by clicking "**Allele frequencies**" in the *Output Tab.* These files are text files, with ";" as separators.

For genotype files, simply ask for "**Final genotype**…" in the *Run Tab*. See **NB.5.1.a** and the preliminary runs of tests **5.1**, **5.5, 5.6** and **5.9** described in Chapter 5 for examples.

4.2 Default values and input parameter files loaded into the GUI

Simulating scenarios also requires setting a given value for each parameter. To help the user when a new *param* file is created, we provide a range of default values in the GUI for all parameters and options. These values are based on our previous tests, practice of the application, and own experience on a range of different biological models (see also **2.4**). Alternative options proposed might however be much contrasted (e.g. the initial conditions for allele frequencies) so we recommend to minimally check those default values, and compare them in exploratory runs if needed.

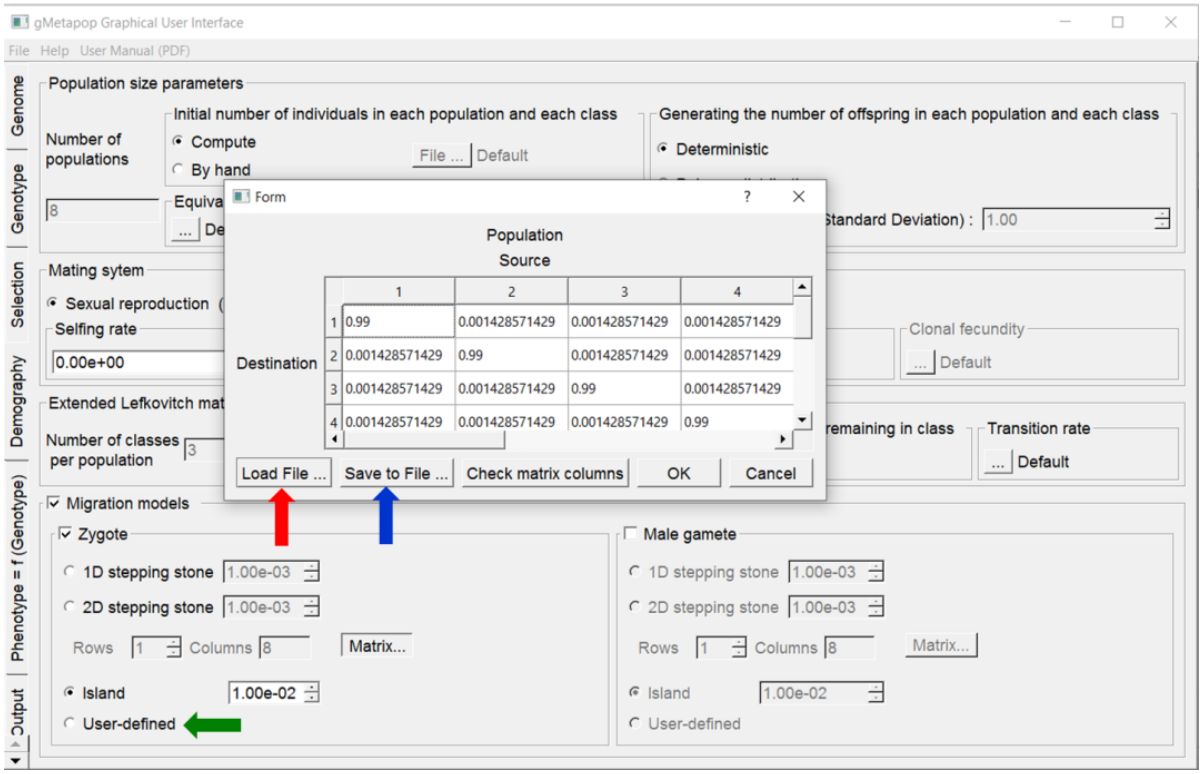
These values can also be changed manually in the GUI, or loaded as simple text files using the "**load file**" options throughout the GUI as shown in the **Figure 4.2** below. The format of these files is of three possible types:

A) They are either matrices including ";" separators (as in *.csv files) with no header, matching the [line x column] structure shown in the secondary windows that allow either to fill them up by hand, or to load their values from a file (red arrow in **Figure 4.2**). When filling those matrix values, several of them can also be saved for future use and reloading, using the **Save** option (blue arrow in **Figure 4.2**). This also allows checking the format of the simple files to load, if needed.

B) Or they include the lines matching exactly those needed for the future conf.txt. This concerns the "**load map**" option in the "**Nuclear loci genetic map"** window of the *Genome Tab***:** the genetic map file required here is a text file corresponding precisely to the lines extracted from the conf.txt file. These lines set the relative position of neutral and loci potentially under selection on the map, and their pairwise recombination rates. The easiest way to get the correct format is to ask for the genetic map file in a preliminary run, using the option "**Save map**" of the same window.

C) Or in the particular case of mutation rates (i.e. "**Mutation rate"** window, "**User-defined"** options in the *Genome Tab*), all values need to be in a text file in one line *with the ";" separator* (e.g. "1e-005; 1e-006; 1e-006; 0.0001" for 4 loci). See examples of such files online here. Note that in the conf.txt file, mutation rate values are organized according to the relative positions of the different classes of loci saved in the **genetic map**.

Figure 4.2: ***Demography Tab*** with an example showing values for the default island migration model, with a migration rate of 1% and 8 populations. These values can be further modified by hand, or saved to a text file (blue arrow), and/or reloaded with different values (red arrow) with the option "User-defined" (green arrow). The user can also load its own matrix with this option.



Examples for each type of file are provided in the different tests described in **Chapter 5** (see the links in **Table 4.2** below)

Table 4.2: List of possible Input files with examples.

Possible Input File	**Tab**	**Format type**	**Test examples with files described in Chapter 5** (and to download from the website at **5-Tutorials**)
Mutation rates	Genome	**C**	See examples at the website in the folder …**4-Format.Examples**
Genetic map	Genome	**B**	**5.5**, and examples as above
Recombination rates in Genetic Map	Genome	**B**	**5.5**
Allele frequency	Genotype	**see 4.1**	**5.1, 5.2, 5.5, 5.6, 5.9**
Genotypes	Genotype	**See 4.1**	**5.5, 5.9**
Genotype fitness values	Selection	**A**	**5.5, 5.6**
By hand initial distribution of individuals in classes	Demography	**A**	**5.10, 5.9**
Carrying capacities	Demography	**A**	**5.1, 5.5, 5.9**
Female potential fecundities	Demography	**A**	**5.9**
Equivalent class sizes	Demography	**A**	**5.9, 5.10**
Probability of remaining in class	Demography	**A**	**5.9, 5.10**

Transition rates across classes		**A**	**5.9**, 5.10
Zygote migration rates	Demography	**A**	**5.3, 5.4, 5.9**
Male gamete migration rates	Demography	**A**	**5.4, 5.9**

4.3 Files which are created in the working folder

4.3.1 When preparing a simulation, just before running it

4.3.1.1 Compulsory files needed to run a simulation

Conf.txt: the GUI (see **Chapter 3**) gives all the details on how the conf.txt has been created. For more information, see a few annotated conf.txt examples provided online.

NB.4.3: the last line of the conf.txt file refers to the path where the allele frequency file has been created by the GUI (in both Linux and Windows versions). **This last line is ignored** if frequency or genotype files are loaded as initial conditions, and are being copied in the working folder (see **3.3.2**, **3.8**, **3.8.7**, and **NB.3.8.7-d**).

Type.txt: text file needed to generate the computation of various statistics during simulations, see **1.2**, **1.4**, **2.4** and list of keywords for summary statistics requested in **Table 3.7.5**.

Command lines (*.bat under Windows or *.sh files under Linux), which are produced by clicking "**Run"** in the *Run Tab* of the GUI (see *3.8.7.1* and also *3.8.7.4* for starting simulations directly from command lines).

4.3.1.2 Additional files

A "QTL_initial_value.txt" file is created that includes the initial additive values for all potential alleles of the different QTL potentially under phenotypic selection. This file corresponds to the lines printed below "Additive allelic effect" the conf.txt file. Please note however that the term "effect" here is inherited from older codes, since the values printed in the conf.txt file are theoretical additive allelic values set up for one scenario and not the additive allelic effects *per se* which are recomputed at each time steps, as explained in part **3.6.6** above.

Other files are used for various unitary or functional tests linked to the reading and checking of information loaded into the *param* file, they are automatically created by the GUI while building the *param* file. They are located either in the working folder or in the "/temp" folder below. They are listed here but should not be needed by users:
allele_number.txt
conf_create.log
inherit.txt
mem.txt
mutation_rate.txt
recom.txt.
res1_simul.log

4.3.1.3 PRNG files

PRNG files are printed either at the start of a simulation or updated at the end of it. See the **3.10.1** part for a detailed description.

4.3.2 During or after launching a simulation

After launching a simulation from the GUI (see the various ways in doing so in **3.8.7**), two types of files are created and filled during the run:

- Output results files: see exhaustive list in **4.4** below

- Files produced by the **File/Default plot** or **File/Custom plot** options from the GUI menu (see **3.9**).

4.4 Description of Output result files and summary statistics

Output files include for example **resX_per_gen_Y.txt**, **resX_per_pop_Y.txt**, **resX_freq_Y.txt**, **resX.txt,** and more **(**see **Table 4.4.1** below**)**. "**X**" is the replicate number (see *3.8.7.1*), and "**Y"** is the index indicating the set of reproductive events and time steps at which summary statistics (see **Table 4.4.2**) are requested in the *Output Tab* in each "Output # **Y"** window.

Table 4.4.1: List of possible result files or files produced by the GUI, and list of softwares in which they can serve as input (column 3). Many examples of those result files are provided in the different tutorial folders at the gMetapop website, with the corresponding step-by-step tutorials in **Chapter 5** below.

File name	Description	Files can be used as input in
map text file (name chosen by the user)	File saved when requesting that the genetic map file be saved (click on **Save map** in *Genome Tab*, "**Nuclear loci genetic map**" window, see **3.2.3** and examples in the **5.5** tutorial)	**gMetapop_GUI**
Simple text files of parameter value matrices which can be saved from the GUI (not detailed here)	See examples in tutorials **5.3** and **5.7** of Chapter 5	**gMetapop_GUI**
resX_per_gen_Y.txt	Summary statistics requested for replicate X (= **repX**) and merged in one file across reproductive event(s) steps for each Output # (=**stepY**, see many example files in tutorials)	**R,** with default plot script **plot.gM.perGen.r**
resX_per_pop_Y.txt	Summary statistics requested per population for **repX** and **stepY**	
resX_freqY.txt	Allele frequencies per population for **repX** and **stepY**	**gMetapop_GUI**, and **R**, using the Custom plot option for example
resX_sample_Y.txt$^{\$}$	Sample files for **repX** and **stepY**	**R**$^{\$}$
resX_sample_Y_genepop_neutral_Z.txt$^{\$\$}$	Separate sample files produced for codominant neutral loci at each **repX** and **stepY, and at each reproductive event Z (=genZ)** in the format required for the program Genepop	The **Genepop** software
resX_sample_Y_genepop_sel_Z.txt$^{\$\$}$	Separate sample files produced for loci under selection at each **repX, stepY and genZ** in the format required for the program Genepop	The **Genepop** software
resX.txt*	File including demographic information at each replicate **repX**	
resX_ai_Y.txt	Estimated average excess effects, and additive allele effects in panmixia (ai) and in the general case (ai.gc) across QTL and at each **repX** and **stepY**	**R**
gVal.merge.pop.txt	Genotypic/Breeding values at each QTL printed after the last set of reproductive events (working file)	**R**
save_indX.txt**	Genotype file for all remaining individuals at the end of a simulation for **repX** (see **3.8.3**)	**gMetapop_GUI** and **R**
resX_simul.log	Information on the computation of some statistics	
Conf_create.log	Output from unitary tests while creating the con.txt file, not intended for the user unless bus are encountered	

Content of **Run log** window of the GUI	See **3.8.8** for a full description. Since this window content has been freezed to avoid bizarre mixed outputs while running simulations, extracts for bug reporting can be done with the Shell window below	
Content of Win64 OS/DOS Shell window***	Same as above and more details reporting the filling up param xml file; text extracts can be copied.	

$ The generic sample file can easily be imported into R. This file includes two more columns compared to a genotype file: an index column for sampled individuals, and individuals' phenotypic values. It can thus be easily edited to serve as an input genotype file.

$$ The Genepop files can be used as input files for the Genepop software (Rousset 2008). The Genepop format can also be imported into other programs (e.g. Arlequin, SpaGeDi, R).

* In contrast to all other results files, the resX.txt is a working file which can provide useful information about the details on the demographic cycles during simulations.

** The format of the complete file of genotypes at the end of a simulation is very close to that of sample files without the Generation column and without the column with phenotypic data.

*** Available in the Win64 OS GUI only, not visible in linux in current release.

NB.4.4: In scenarios with selection on phenotypes, individuals' phenotypic values can be obtained in sample files only. If the user needs to obtain phenotypic values across all individuals remaining at the end of a simulation, a "New output" can be requested for the last set of reproductive/demographic event(s), using the maximum possible number of individuals (i.e. Carrying capacity), even if the populations sizes have decreased during simulations.

Table 4.4.2 Definitions and/or references for summary statistics abbreviations as they are printed in result files, and in the legend of the default plot for some of them (see **Table 3.7.5** for the correspondence between this list, the checkbox options in the O*utput Tab,* and the different types of results files; see also part **3.7.6** for more information on particular statistics). Column 3 provides abbreviations appearing in the plot legend of the Default plot R, for both statistics and a summarized list of parameter values from the configuration file (conf.txt).

Summary statistics abbreviations in result files*	**Definition and/or Reference**	**Abbreviations used in the Default plots legend**
Genetic Diversity within populations		
Ho_lx	observed frequency of heterozygote individuals (observed heterozygosity) at locus x	NA
He_lx/_clx	diversity (expected heterozygosity) at nuclear locus x **lx** (Nei 1987), or cytoplasmic locus **clx**	NA
Fis_lx	deficit or excess in heterozygotes at locux x: 1-(Ho_lx/He_lx)	NA
Ho_cod1/2	mean observed heterozygosity Ho_lx across codominant 1 (cod1) or codominant 2 (cod2) groups of loci	NA
He_cod1/2	mean He_lx across cod1 or cod2 loci	NA
Ho_sel	mean Ho_lx across loci potentially under genotypic selection	NA
He_sel	mean He_lx acroos loci potentially under genotypic selection	NA
Ho_qtl	mean Ho_lx across qtl	NA
He_qtl	mean He_lx across qtl	NA
Ho	mean Ho_lx across all nuclear loci in each population	Ho$
He	mean He_lx across all nuclear loci in each population	He$
Fis	mean Fis_lx across all nuclear loci in each population	Fis$
Allele frequencies *printed in files* ***resX_freq_Y.txt***		
f.lxaz, f.clxaz	Allele frequency at allele "z" for nuclear locus lx or cytoplasmic locus clx	

Between or across populations *(statistics are only computed for more than one population)*		
Gst_lx/_clx	Gst at nuclear locus x or cytoplasmic locus x (Nei 1987 p.190 formula 8.27), Gst_lx/_clx =(Ht_lx/_clx-Hem_lx/_clx)/Ht_lx/_clx	Gst_l/Gst_cl
Gstcor_lx/Gstcor_clx	Gstcor at locus x (=F'_{ST} of Nei 1987 p.163 formula 7.37)), Gstcor =d*(Ht_lx-Hem_lx)/(d*Ht_lx-Hem_lx), same formula with clx for cytoplasmic loci	Gstcor_l
Hem_lx/_clx	mean across populations of He_lx/_clx at each locus	Hem_l
Fism_lx	mean across populations of Fis_lx at locus x	Fism_l
Ht_lx/_clx	expected diversity in the total population (using mean allele frequency) at locus x/clx (Nei 1987 formula 8.26 p.189)	Ht_l
Gstm	(1-Hem/Ht) with **Hem** = mean Hem_lx across nuclear loci and **Ht** = mean Ht_lx across all loci	Gstm
Gstcorm	Gstcorm =d*(Ht-Hemx)/(d*Ht-Hem), see also **tests 5.3**	Gstcorm
Gstcod1/2 (resp. Gstcorcod1/2)	Gstm for cod1 or cod2 loci (resp. Gstcorm for cod1 or cod2 loci)	Gstcod1/2 (resp. Gstcorcod1/2)
Gstsel (resp. Gstcorsel)	Gstm for loci potentially under genotypic selection (resp. Gstcorm for the same loci)	Gstsel (resp. Gstcorsel)
Gstqtl (resp. Gstcorqtl)	Gstm for loci potentially under phenotypic selection (resp. Gstcorm for the same loci)	Gstqtl (resp. Gstcorqtl)
Demography		
St	total simulated equivalent carrying capacity (K) across populations (e.g. space occupied by individuals) at one particular reproductive event	NA
NA	St at the last reproductive event (**re**) recorded in **res1_per_gen_1.txt**	St.**re**
NA	mean St across all recorded **re**	St.m
Sp	simulated equivalent K in each population	NA
Nt	Total number of individuals (across classes and populations)	NA
NA	Nt at the last recorded **re**	Nt.**re**
NA	mean Nt across all recorded **re**	Nt.m
Np	total number of individuals per population (across classes)	NA
Numbers of individuals in each class per population**	**printed in the resX.txt files (see "Population sizes after demography by class")**	NA
Nsext, Nsexp	number of new individuals produced from sexual reproduction, across classes per population ("t" in total population and "p" for each population)	NA
NA	mean Nsext across all recorded **re**	Nsext.m
Nclot, Nclop	number of new individuals from clonal reproduction ("t" across all populations, and "p" per population)	NA
NA	mean Nclot across all recorded **re**	Nclot.m
Nremt, Nremp	number of individuals from previous reproductive/demographic events remaining in their classes ("t" in total Nremt=Nt-Nsext-Nclot and "p" per population Nremp=Np-Nsexp-Nclop)	()
	mean Nremt across **re**	Nremt.m
Npass**	number of individuals changing class in each population at each set of reproductive/demographic events	NA
Nsurv**	matrix of the number of individuals which remained in their population by classes after one set of reproductive/demographic events, (see also Nremp)	NA

Nenf_sex_tot**	statistic that is the same than Nsext above	NA
Nenfv**	matrix of the number of new individuals issued from clonal reproduction in each population by class (see also Nclop)	NA
mean_fitness	mean fitness across individuals in each population (for loci potentially under phenotypic or genotypic selection)	NA
***Quantitative Trait Variability**# Refer to **Lynch and Walsh 1998 chapter 4** for all additive effects, variances and linked statistics (see also some details in **3.7.6.3**)*		
ai.qxaz, ai.gc.qxaz	additive effect assuming panmixia for **qtl "q" x** and **allele "a" z**, or in the general case (no assuming panmixia) in one population	NA
aei.qx.az	average excess for **qtl x** and **allele z** in one population	NA
VA.p.qtlx, VA.gc.qtlx	additive variance at each **qtl x** assuming panmixia, or not assuming panmixia (.gc)	NA
VA.p§, VA.gc§	additive variance in each population	NA
VA.p.tot, VA.gc.tot	additive variance in the total population (i.e. all individuals of the different populations considered together)	VA.p.tot, VA.gc.tot
h2.pan, h2.gc	VA.p/VP, VA.gc/VP Narrow-sense heritability	NA
NA	mean h2.pan/h2.gc across populations	mh2.p$, mh2.gc$
thetaW.p.g	Add formula	NA
thetaW.gc.g	Add formula	NA
thetaW.p.t	Add formula	NA
thetaW.gc.t	Add formula	NA
mthetaW.p.g/.gc.g	mean thetaW.p.g/.gc.g across populations	mthetaW.p.g/.gc.g$
mthetaW.p.t/.gc.t	mean thetaW.p.t/.gc.t across populations	mthetaW.p.t/.gc.t$
Qst.p, Qst.gc	*Add ref*	Qst.p, Qst.gc
Geno/Pheno additional information		
VG	total genotypic variance in each population (includescovariation components among pairs of loci)	
Vg_qtlx	genotypic variance – at each qtl x	NA
Vg	mean across loci of Vg_qtlx	NA
mVG	mean VG across population	mVG$
mVg	mean Vg across populations	mVg$
VP	phenotypic variance in each population	NA
mVP	mean phenotypic variance across populations	mVP$
Additional statistics plotted in the Default plot##		
NA	mean **Ho** value (see above) across populations	Hom$
NA	mean **He** value (see above) across populations	Hem$

NA	mean **Fis** values (see above) across populations	Fism[$]
NA	mean of Ho_cod1 values across populations	Hom_cod1
NA	mean of Ho_cod2 values across populations	Hom_cod2
NA	mean Ho across loci under genotypic selection and populations	Hom_sel
NA	mean Ho across QTL and populations	Hom_qtl
NA	mean He across cod1 loci and populations	Hem_cod1
NA	mean He across cod2 loci and populations	Hem_cod2
NA	mean He across sel loci (i.e. loci under genotypic selection) and populations	Hem_sel
NA	mean He across qtl and populations	Hem_qtl
NA	average of **mean_fitness** values (computed across individuals in each population)	mean fitness
NA	mean narrow-sense heritability across populations	mh2.p[$], mh2.gc[$]
Parameter values from the configuration file (conf.txt), which are plotted in the Default plot legend of a simulation scenario (the first column here refers to parts of the User Manuel with more details and definitions)		
3.4.3	Whether the scenario considers soft or hard selection	Soft/hard selection
	Number of populations	Npop
	Number of demographic classes	Ncla
3.5.3	demographic class number for clonal offspring	Clonal.offspring.cla
3.5.3	class transition rate range	Cla.transition
3.5.3	range of equivalent class size values[$$]	Cla.eq.size
3.5.4, *3.5.5.1*	range across theoretical or potential (initial) zygote migration rates from source to destination populations	t.zy.m
3.5.4, *3.5.5.2*	range across theoretical or potential (initial) male gamete migration rates from source to destination populations	t.mg.m
	number of cytoplasmic loci	Ncyto
3.2.1	range of mutation rates across cytoplasmic loci	cyMut
	maximum number of alleles for cytoplasmic loci	cyAl.nb
3.2.1	range of paternal inheritance values	cy.pat.inheritt
	number of codominant loci (cod1 + cod2)	Ncod
	number of loci potentially under selection	Nsel
	number of qtl	Nqtl
3.2.2	range of mutation rates across all nuclear loci, whatever their type (neutral or under selection)	nuMut
3.2.3	range of recombination rates across all nuclear loci, whatever their type (neutral or under selection)	Rec
3.3.1	maximum number of alleles for codominant loci and qtl	Al.nb cod/qtl
3.3.1	maximum number of alleles for codominant loci and loci potentially under genotypic selection	Al.nb cod/sel
3.5.1.2	distributions for generating offspring: det=deterministic, Poi=Poisson, or Norm=Normal	Offspring
3.5.2	realized selfing rate in panmixia	Equil S rate
3.4.2.1*, 3.4.4, *3.4.4.2	range of stabilizing selection intensity across populations	w2.span
3.4.2.1*, 3.4.4, *3.4.4.3	Variance of phenotypic optimum values across populations	Vzopt
3.4.2.2	range of genotype fitness values	fitness.val
3.5.1.1	range of carrying capacities across populations	K.span
3.5.1.1	mean of carrying capacities across populations	K.m
3.5.2, *3.5.5.6*	mean/variance across absolute female fecundity values	ffec.m/var
3.5.2, *3.5. 5.6*	range of absolute female fecundities	ffec.span
3.5.2, *3.5. 5.7*	clonal reproduction absolute fecundity	Clonal.fec

***** Result files are listed in **Table 3.7.5** that indicates which statistic is printed in each file, and the few statistics also computed for cytoplasmic loci for which the same definitions apply.

****** additional demographic statistic printed in **resX.txt**

for all quantitative trait variability statistics, the suffix "**.p**" correspond to estimates that assume panmixia, and "**.gc**" correspond to estimates with no panmixia hypothesis (See **Tests 7** in **Chapter 5** for example).

these statistics are not available in the res1_per_gen_1.txt output file, but they are computed averages from the **res1_per_pop_1.txt** result file when available, and included for plotting in the default plot. See the code of the **plot.gM.perGen.r** script for more details on those statistics if needed.

§ Additive variances in each population are not printed in the **res1_per_pop_1.txt** but can be deduced from the corresponding h2.pan/h2.gc and VP values.

$ In scenarios with only one population, the population statistics' abbreviations are used in the **plot.perGen.png/pdf** figure legends (they do not include the "m" prefix or suffix).

$$ For all parameter value ranges indicated in the Legend of the plot, a single value means that they are all the same.

5. CHAPTER 5 Evolutionary scenarios: testing gMetapop by comparing simulated results with theoretical or published results.

This chapter provides step-by-step tutorials (or tests) for using the GUI in different evolutionary scenarios. Please refer also to part **1.4** for a quick overview of gMetapop, and to the **Figure 1.4** for the main steps involved for running scenarios, either from the GUI or directly from the CORE.

In the tests below, the scenarios were chosen in order to:

1) Compare simulation results with classical population genetics theory or published results, with the aims of a) validating the way we modelled the main evolutionary processes in gMetapop_CORE during the development of the software, and b) extending or correcting the historical versions that we merged at the beginning of the project.
2) Explore various GUI options from their contrasted initial conditions, which helped debugging the large range of proposed options and their connections, and check the stability of particular components when developing new features.
3) Provide examples that illustrate many possibilities offered by gMetapop.

The tutorials explain how to re-create the GUI parameter files (*param* *.xml files) in many scenarios. These files generate all conditions for running simulations and getting output files from both the GUI and the CORE. **All files cited in the different tutorial steps are also available online from the gMetapop website …/5-tutorials**, if you want to examine them. Each online folder includes *.zip to download that include one or more *param* files, other files that need to be loaded into the GUI for re-creating those *param* files for each scenario, the working sub-folders with output files from simulation runs, plot examples and corresponding R scripts. The *param* files can then be loaded directly into the gMetapop GUI, and be re-used directly after updating the users' local paths (see **NB.3.0** and **NB.5.1-c**). Additionally, the parameter values in these files can be changed directly from the GUI for building new scenarios. A minimum set of files needed to easily recreate *param* files from all tutorials and corresponding scenarios can also be downloaded here.

As explained previously, one advantage of the GUI is to easily obtain, in preliminary runs, all template formats for the initial files needed for particular scenarios (e.g. initial allele frequency, genotype files, and corresponding genetic maps). These files can be first edited if needed, and then loaded into the GUI, thus defining initial conditions for simulations (see NB.5.1-a, and examples in 5.1.1, 5.2.1, *5.5.5.1*). Allele frequency and genotype files obtained from simulation outputs can also be re-used as new input files for other simulations, or can be loaded into R for further analyses.

NB.5-a: For users who did not read **Chapters 3** and **4**, please examine **Figure 3**, which provides an overview of possible life-cycle events and corresponding evolutionary processes as they are modelled in gMetapop.

Simulation runtime: In addition to the initial settings of each simulation scenario (numbers of populations, classes, individuals, loci, alleles…), the main factors that can increase simulation runtimes are:

- life cycle complexity (overlapping generations and all associated features),
- number of summary statistics requested in the *Output Tab,*
- using selection on phenotypes,

- requesting many quantitative genetics statistics, in particular additive variances and associated statistics in the general case (i.e. no panmixia assumption, see **Tests 7** and **Tests 10**).
- and number of replicates (from multiple conf.txt or replicated runs in the *Run Tab*).

In practice, we first suggest running simulations for one replicate, a few tens or hundreds of reproductive events, and a few summary statistics, in order to roughly estimate runtimes and verify all output contents before launching more time-consuming simulations. For each test and summary statistics requests described below, we provide indicative runtime estimates on particular OS.

NB.5-b: Asking for n replicates means that the CORE program will run replicates one after another, as can be seen in the **run_gMetapopCore.bat** (**or *.sh** under linux) created when clicking on the "Run" button from the ***Run Tab***. The release 1.0 of gMetapop is not built specifically to run in a cluster environment. However, In the case of complex simulation scenarios and expected long runtimes on a classical OS, it's up to the user to split a set of replicates in different groups which can be sent to different nodes in a cluster, since it can greatly decrease runtime. In order to do that, please make sure that you start for each group of replicates with different initial seeds for all **PRNGs**. We aim at improving this issue in future releases of the program.

All tutorials below have been prepared so they can be read and reproduced as independently as possible. However, **we recommend starting with the Tests 1 tutorial** below, since more details and basic principles are provided there about file management**.** Morever, most notes (i.e. **NB…**) can be read separately from the main text. In all tests below, we use default parameter values unless described otherwise. The numbers of the tutorials below refer to the folders and files provided online, which can be imported into the GUI whatever the OS (Win64, Win32 and various linux OS).

5.1 Tests 1: DRIFT

5.1.1 DRIFT on nuclear loci

Aim: Plot the average time of persistence of neutral alleles in an ideal diploid population of size N for a biallelic locus, as a function of the initial minimum allele frequency (see Fig. 3.8 p. 113, and equation 3.10 in Hartl and Clark, 2007).

Building the parameter xml files in the GUI (see Figure 1.4 for an overview):

In a Windows OS, launch gMetapop_GUI_Win64/32.exe. In a Linux OS, please see the corresponding instructions in part **2.1.**

Choose **File/New Interface parameter file** in the GUI menu, then in the "*Initial settings"* window, choose 40 populations, non-overlapping generations, and no selection.

Genome Tab: in the "**Nuclear loci / Codominant 1**" window, change to "**50**" loci, and put "**0**" in "**Mutation rate"/"Mean**" sub-window for a simulation with no mutation. In the "**Nuclear loci genetic map**" window, keep the default option "**Independent loci**".

Genotype Tab: In the "**Nuclear loci / Codominant 1**" window, put "**2**" and "**2**" for maximum and initial number of alleles per locus respectively.

NB.5.1-a: As summarized in part **2.5**, **it is possible** in most cases **to automatically create all the formats required** for allele frequency or genotype files that need to be loaded in the GUI **for particular scenarios**. This is because the required formats for input files are the same than the formats obtained in output files of the same type and characteristics (see their list in **table 4.4.1**), which can be produced by the GUI in the following way: **First**, the correct formats are obtained in **preliminary runs,** using different options across *Genome, Genotype*, *Output Tab* or *Run Tab* (see **Chapter 3** if needed for more details on those Tabs), depending on the type of file to export. **Second**, values within those preliminary files can be chosen so that they are as close as possible to those required in the files to load. In many cases however, the actual values in these preliminary files need to

be edited further (see the examples provided throughout this chapter). This can either be done with your favorite text editor for specifically changing values at particular loci, alleles, populations, individuals (e.g. Notepad++ at https://notepad-plus-plus.org/downloads/), making sure that your editing work does not change the format, or with small (R, …etc) scripts for either filling up the values in these files (as in the test below). Please remember that **default values are always proposed** to easily simulate scenarios that would be similar or simpler than the particular scenarios of interest.

In this test, performing a preliminary run from the GUI allows obtaining the correct format for the initial allele frequency file that is needed to run the simulation (see *5.1.1.1* below, or go directly to ***5.1.1.2* Complete run configuration** using the files to load, which are provided online).

*5.1.1.1 Preliminary run configuration (saved in **test1.1-prelim.xml**)*

We illustrate here two different ways for producing allele frequency files with the correct formats and headers (i.e. first lines with statistic names).Then initial allele frequency values within this file need to be modified since they vary from 0.01 to 0.975, with an increment of 0.025 (see **NB.5.1-a** above, and more details below).

a) Either load the **test1.1-prelim.xml** file and go directly to the *Run Tab* part: this should give an error message that is expected: "**The working directory isn't correct**" since you are loading a file that was saved from another computer and folder organization. Therefore, you only need to update the working directory name (see **NB.5.1-c** below or the Appendix online where most error messages can be searched for and further explained).
b) Or re-create the configuration file as explained below:

In the *Genotype Tab*, choose "**Create"** in the "**Allele frequencies**" windows, and choose "**All loci fixed …"** in the "**Codominant 1**" window.

Selection Tab: keep default values (no selection case).

Demography Tab: all default values for this run.

Output Tab: choose 0 "**Total number of reproductive events**", and then "**reproductive event 0 to 0 every 1**" in the "**Output # 1"** window. Either unclick any default summary statistics (or ask for the one that need to be checked initially in particular scenarios). Click for "**Allele frequencies**".

Run Tab: Define or choose the working folder. Here you can (re)save the preliminary *param* xml file.

Then click on both "**Create"** in the "**Type of result file**" window and in the "**Prepare Configuration Files**" window with the "**Simple**" option (default). Ignore messages if you do not wish the save the xml file, or if you have saved it already (since messages are issued if the xml file has not been saved in the last 10 seconds), and ignore the warning message related the "0" mutation rate.

Then in the "**Run gMetapop**" window, click on "**Run**".

This is instantaneous since only generation 0 has been requested in the *Output Tab*. In the working folder, you can then examine two different allele frequency files that have been created, and which have the correct formats required to perform a complete simulation if needed:

a) A file named **allele_frequency.txt** that is always created at the start of each run, which includes the frequencies used for producing the genotypes at generation 0, according to the different options of the *Genotype Tab*.
b) A file named **res1_Freq1.txt.**

Both files are identical in this example.

NB.5.1-b: In general, please note that the **allele_frequency.txt** file that is systematically created at the start of a simulation is set according to the initial theoretical conditions chosen in the "**Initial conditions**" window, and in the *Genome* and *Genotype Tabs*, with a generation column starting at "0". Also, if you click on "**allele frequencies**" in the *Output Tab*, the program re-computes frequencies at generation 0 (of one simulation run or replicate) and they are printed in the **resX_FreqY.txt** output file (see **Table 4.4.1**). This is done at generation 0 by using the genotypes of the individuals that were previously created by random draw using the allele frequencies at the start of simulations, so depending on the initial population sizes chosen in the *Demography Tab* (which can be empty or have a much smaller size than the ones at equilibrium), those allele frequencies can differ slightly from the initial values. The **resX_FreqY.txt** file also stores allele frequencies for each generation or reproductive event at which they have been requested in the *Output Tab*.

*5.1.1.2 Complete run configuration (saved in **test1.1-complete.xml**)*

a) Either load **test1.1-complete.xml**, check values in the different Tabs, and go directly to the *Run Tab.* Here again, a GUI error message "**The working directory isn't correct**" appears, which means that the working directory needs to be updated (see **NB.5.1-c** below), as well as the path for the allele frequency file.

In order to do that, go into the *Genotype Tab*, choose "**Load frequencies**" and click on "**load file**", then choose the file **test1.1.loadfreq.40pop.50loci.2al.csv** which is provided at the root of the **5.1.1** folder that can be downloaded online). A *.txt extension is also possible for this frequency file to load. This file was created separately in a spreadsheet or with a small script as explained in **NB.5.1-a**, with ";" as separator, see also **4.1**), using 50 independent biallelic loci with varying frequencies across populations. The correct format structure for this file can be obtained in a preliminary run, as explained in *5.1.1.1* above

NB.5.1-c: In general, when reloading the *param* *.xml files that are provided for each test, most parameter values are already included, even if they were loaded from text files (i.e. you do not need to load these files again). However for files such as allele and genotype frequencies, or genetic maps, only their their paths are stored in the *param* files, so these need to be updated to match those of the user. To do this, you simply **need to reload** the frequency, genotype or map files into the GUI, as this will then document users' own folder organization. The files to reload are provided with each test/tutorial at the gMetapop website, just above working simulation folders. Finally, **do not forget to save your updated *param* file** ("**File/Save Interface parameter file"**), before running a simulation or before closing the application.

b) Or continue to re-create the configuration file, after having done the first steps described before part 5.1.1.1 for the *Genome* and *Genotype Tabs*:

Then still in the *Genotype Tab*, choose "**Load frequencies**" and click on "**load file**", then choose the file **test1.1.loadfreq.40pop.50loci.2al.csv** as in a) just above**.** The whole path for the file location appears in the window (see **3.3.2**).

NB.5.1-d: When the conf.txt is created (in the *Run Tab*, see below), the program performs most of the time a simple check of the headers of the loaded files. Therefore, if there are any inconsistencies between the loaded files and the GUI concerning the number of populations, the number of loci or the maximum allele number, error messages are generated. For example, make sure that the number of alleles chosen in the "**Nuclear loci / Codominant 1**" window is consistent with the loaded frequency file. More information can be found in the **Appendix** online on possible error messages.

NB.5.1-e: Please note here the column "Generation" with a "0" value in the initial files to load. This is the general case, and by convention the first generation number in simulations. However if the user loads a file with a generation value different from "0", this value will be ignored and the generation counter reset at "0".

In the *Selection Tab*, choose no selection.

In the *Demography Tab,* for the Carrying Capacity (K): click on "**Equivalent carrying capacity** …" **default** values to 100 in each population to mimic ideal populations with an effective population size (Ne) of around 100 in a deterministic model. Either load **test1.1.K100.40pop.csv** (faster to produce beforehand given the number of populations), or

the user can change by hand 1000 to 100 in each population K value window (convenient for a small number of populations).

Default values are kept for male and female fecundities.

No migration, each population is an isolated population evolving neutrally.

Phenotype = f(Genotype) Tab: this Tab is not activated when there is no selection.

Output Tab: Put "**1500**" for the "**Total number of reproductive events**" (i.e. number of generations here), which is enough *a priori* to vizualise the test results for a carrying capacity ***K*** (i.e. *Ne* here) of 100, then put values **from "0" to "1500" every "4" "Reproductive events**" in the **Output #1** window, and click on "**Allele frequencies**" only (i.e. unclick other default summary statistics), alternatively click on all statistics available.

NB.5.1.f: "4" here is an example for the generation (or reproductive event) step at which requested output statistics are printed, but you can choose any number from 1 to the largest number of generations. Using "4" instead of "1" allows the simulations to go faster with a limited impact on the precision of the final plot, since computing and printing summary statistics constitute important parts of the runtime in general.

*Run Tab: "***Select/Create***"* the working folder (which is either already available in your folder structure, or which can be directly created here using your graphical OS environment). The working folder is the folder indicating the location of all files that are produced by the GUI during the construction of the *param* xml file, and by the CORE when running simulations.

NB.5.1.g : As mentioned before, if the *param* xml file that is being used is the provided file, names of working folder path and other paths for files to load need to match the users' own folder organization. Otherwise, a message should warn that the working folder is not correct (see **NB.5.1.c**).

NB.5.1.h: **As a general rule**, avoid space characters in folder names. Even if they are not a problem in Windows OS, they might be in a Linux OS if files are copied from one system to another.

Save the *param* file: **File/Save Interface parameter file** in the GUI menu.

Then click on "**Create"** in the "**Type of results file**" window. This creates a type.txt (default name) file in the working folder. Alternatively, click "**Select"** and choose files from other folders, as long as they have the correct type.txt-like format (for example here in the same working folder you can choose the file **test1.1.type-all-stats.txt** that is located in the folder above the working folder). You can examine the difference between **type.txt** and **test1.1.type-all-stats.txt** where more statistics are requested.

Then click on "**Create**" in the "**Prepare Configuration Files**" window and choose "**Simple**" (default) for a simple run (in contrast to replicates with different configuration files when choosing "**Multiple**"). Here, a warning will tell you that the "**test1.1.loadfreq.40pop.50loci.2al.csv**" file already exist in the working folder if you choose to use the provided folder online with all its files, or if you relaunch a simulation twice. Both answers are possible here.

NB.5.1-i: Asking for multiple configuration files means that they will include more differences among replicates for starting conditions (i.e. since different random draws are used for particular options in the different Tabs), compared to simply replicate simulations with the same initial configuration files (see "**Run gMetapop**" window): after choosing **"Multiple**", choose also the number of conf.txt to be produced per run, and then click on the "**Prepare run and create conf.txt files"** option. See also **3.9.1** for performing plots from the GUI with "**Multiple"** conf.txt.

Finally, in the "**Run gMetapop"** windows, ask for **1** replicate (default) since the 50 loci here serve as replicates. Alternatively, additional replicates per configuration file can be chosen. Then click on "**Run**" and you can see a "dot" counter in the log window for every 10 reproductive events.

Runtime of ~1 min on **Win64 OS** (i.e. here and in following tests with a Intel Core i7-10850H CPU @ 2.70 GHz processor, and 32 GB Ram)

NB.5.1-j: in another example of the same simulation with K~1000 and 15000 generations, with all statistics being asked, the *runtime* was ~1h30 (Win64 OS).

NB.5.1-k: At this stage, once runs are finished, it is possible to close the application, and reopen it later in order to use R plot functions to get an overview of the results.

Post-treatment

First have a look at the default plot with all statistics requested (see *Output* and *Run Tabs* above to select this option): click on "**Default plot**" to check that the overall plot across generation is consistent with the drift expectations. See the example in the file **plot.perGen-all-statistics.pdf**.

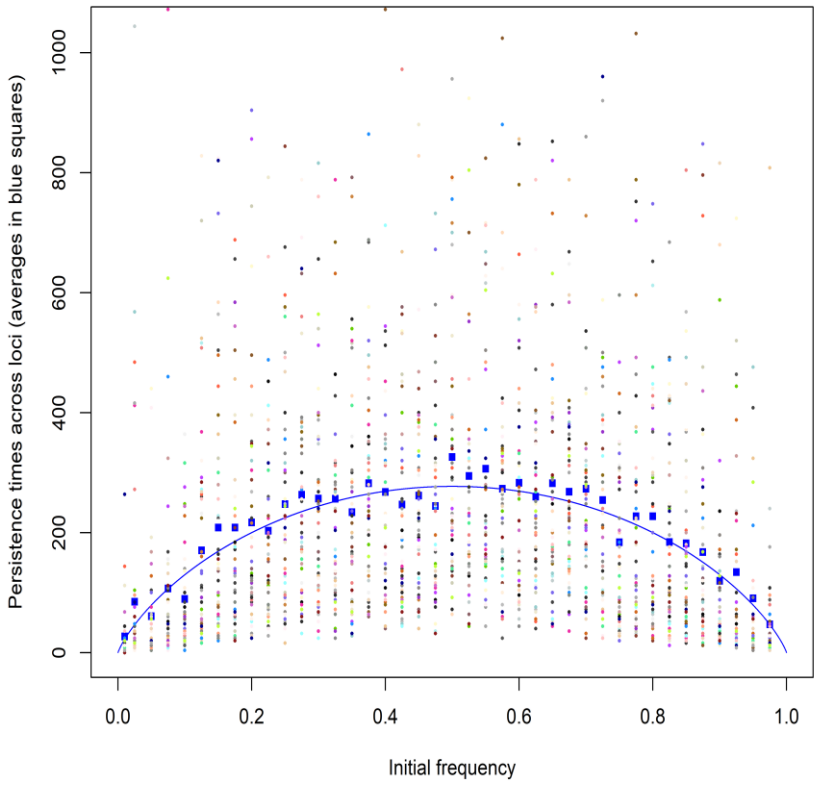
NB.5.1-l: Both "**Default plot**" and "**Custom plot**" options will be active after having performed the "**Choose R path**" once successfully in the **File** menu (see also **3.9**).

Second, use the R script provided **test1.1.custom.plot.persistence.times.r** with the **File/Custom plot** option in the GUI menu in the **run-complete** folder for example (so copy both the R script and popfinit2.txt in the working folder). The R script file, and also the file **popfinit2.txt** need to be copied into the working folder beforehand.

Then click **File**/**Custom plot** in the GUI menu, then type (or copy & paste) the following command in the window:

test1.1.custom.plot.persistence.times.R res1_freq_1.txt test1.1.custom.50loc

Examine the plot **test1.1.custom.50loc.png** (below), which should appear onscreen, see also **3.9.2** for more details on Custom plots.



This script requires the R package plyr to be installed, it imports the final allele frequency file **res1_freq_1.txt**, computes loci persistence times and averages across loci, which follows the theoretical expected curve (see the figure in file **test1.1.custom.50loc.png**). The results illustrates the stochasticity of the genetic drift process across loci, especially in this scenario with small effective population sizes.

5.1.2 DRIFT on loci with maternal (or paternal) inheritance

Aim: similar to **5.1.1** but for haploid cytoplasmic loci, K=100 in the example provided.

Here since haploid loci are completely linked by construction, the way to mimick 50 statistically independent loci is to simulate 50 replicates that represent independent simulations of a single haploid locus. Each population has a different initial frequency as in **5.1.1** (see *Genotype Tab* below).

Inheritance of cytoplasmic loci is modelled as being either maternal, paternal or mixed. Since hermaphrodite diploid individuals are simulated in gMetapop, genetic drift is expected to be twice as strong overall using these haploid loci compared to nuclear loci.

Building the parameter xml file: (saved in **test1.2.maternal.1cyto.50rep.xml**)

Initial settings: 40 populations, non-overlapping generations, no selection.

Genome Tab: only 1 cytoplasmic locus in each population, choose maternal inheritance (default) so unclick also "**Nuclear loci**", no mutation (or check that it is unclicked if you reload the saved xml file).

Genotype Tab: "2" for maximum and initial number of alleles, then load frequencies, with files **test1.2.loadfreq.40pop.1cl.2al.txt** (see **NB.5.1-a** above for getting the correct allele frequency file format and header from a preliminary run if needed, and use the same range of allele frequencies as in **test 5.1.1**).

Selection Tab: no selection

Demography Tab: Carrying Capacity (here ~Ne) is 100 thus load (**test1.2.K100.40pop.csv**), "**Deterministic"** draw of offsprings (default), default values for male and female fecundities, and no migration to mimick 40 separate populations.

Output Tab: 1000 reproductive events (generations here), every second generation, ask for allele frequencies in the **Output # 1** window that are needed for the custom plot (with any additional statistics if you want ot vizualise the default plot).

Run Tab:

Click on "**Select/Create**" to define the working folder (named "**run-maternal"** in the **test 5.1.2** *.zip file to download **online**.

Change the default "**Number of replicates per configuration file**" value of "**1**" to "**50**".

Save the *param* file (**test1.2.maternal.1cyto.50rep.xml**), and create both the type.txt and conf.txt files.

NB.5.1-k: Warning – as a general rule, **do not use the name res1_freq_1.txt for an initial allele frequency file** as it is reserved for output frequency files. This would yield an error message in the case of more than one replicate since the file of the same name would be filled with more generations, and thus become incorrect as an initial file for the following replicate. Just give a different name to the initial file.

Runtime: ~2 minutes.

Post-treatment

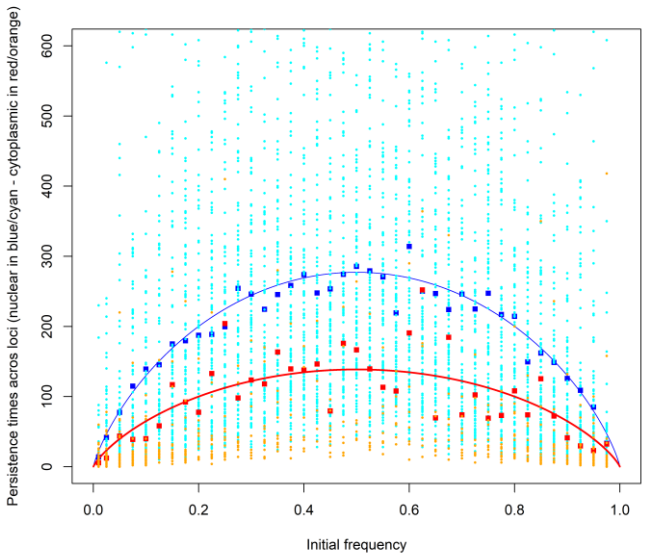
In order to compute average persistence times and compare them with theoretical values, we use the R script **test1.2.custom.plot.persistence.times.50rep.r** in a **File/Custom plot** option from the GUI menu. The script needs to be copied into the working folder **run-maternal** (with the file popfinit2.txt needed in the script; this can be done independently to running simulations), and use a similar command line than in the post-treament part of *5.1.1.2* above

test1.2.custom.plot.persistence.times.50rep.R res1_freq_1.txt test1.2.maternal

This script imports the 50 allele frequency files **resX_freq_1.txt (X** being the number of replicates, but only the first one is needed as the output file argument), merges them, computes all loci persistence times and their mean values across loci for each initial frequency. Then, it plots them along with the theoretical curve (see the plot in the file **test1.2.maternal.png** using the command line example above, or enter any other plot file name in the custom plot command line, see **3.9.2**). As in *5.1.1.2*, the stochasticity of the genetic sampling process at each generation is illustrated across the 50 independent replicates of a haploid locus. In the working folder, see also the default **plot.perGen.pdf** that shows the different summary statistics requested in the *Output Tab* for the first replicate (i.e. for one locus).

Related examples provided:

A combination of both nuclear and cytoplasmic loci with paternal inheritance: configurations (saved in **test1.2.paternal.100cyto.100nuc.xml**), the corresponding allele frequency file to load being **test1.2.loadfreq.100cyto.100nuc.txt** (illustrating the correct format for both types of loci here, using a preliminary run as in ***5.1.1.1***, see also **NB.5.1-a**, and see table **4.4.2** for an exhaustive list of statistics abbreviations). All output files are in the **run-paternal.100cyto.100nuc** working folder from the *.zip file(s) to download online. Here, a comparison of persistence times for both haploid and diploid loci shows the higher strength of drift for cytoplasmic loci on average due to a lower effective population size (see the default plot.perGen.pdf in this particular working folder, and the figure **test1.2.paternal-100cyto.100nuc.1rep.600.png** (below):



This plot is produced by the script **test1.2.custom.plot.persistence.times.100cyto.100nuc.1rep-600.R** used with the

"**Custom plot**" option). This plot also shows the increased stochasticity for haploid loci compared to nuclear ones. This is due here to the simulation of only 1 replicate of those 100 paternally inherited cytoplasmic loci which are linked and not independent by construction, despite their different initial frequencies. In contrast, the 100 nuclear loci serve as independent replicates so the means of persistence times across loci for each initial frequency is closer to the theoretical curve.

A scenario with 50 independent cytoplasmic loci (simulated as 1 locus with 50 replicates as before,) with a mixed "**User-defined**" inheritance of 0.4 (i.e. of paternal inheritance, saved in **test1.2-mixed.xml** - please check that nuclear loci is not clicked if you reload this file). Here a file with a mutation rate of 0 for one locus (**5.1.2-1cyto-mut-at-0.txt)** is loaded with the user-defined option in the mutation rate window for the sake of the example (for multiple loci, **";"** is needed between mutation rate values, see **C) in 4.2**). The corresponding allele frequency file to load is the same file as for the example with maternal inheritance above (i.e. **test1.2.loadfreq.40pop.1cl.2al.txt**). All output files are in the **run-mixed** folder. The **test1.2.custom.plot.persistence.times.50rep-600.r** script can be used for visualizing persistence times (setting Y axis maximum value at 600 generations for a better vizualisation). The plots **test1.2.mixed-600.png** and **test1.2.maternal-600.png** (obtained by re-using the custom plot option with the same **test1.2…-600.r script** as above, copying it in the **run-maternal** folder) in their respective run-maternal/mixed folders are very similar as expected when considering distances of persistence time averages to the theoretical curves.

5.2 Tests 2: Mutation and Dirichlet

5.2.1 Mutation

Aims:

1) Test the modelling of the mutation process.

NB.5.2-a: Historically, no stochasticity was associated to the total number of alleles obtained from mutation events, which was based on the initial mutation rate provided and the total number of individuals and loci. In the current gMetapop_CORE release, this number is the parameter value of a Poisson distribution).

2) In order to do that, we follow the evolution of allele frequencies in two populations that are initially fixed for two alternative alleles, and mimic **reversible mutations** among those two alleles (in gMetapop, this can be modeled if the reversible mutation rate is the same as the actual mutation rate, which is the case for biallelic loci) – see Hartl and Clarck 2007, p. 157, for an example in which forward and reversible mutation rates have different values.

Parameter files are provided for 2 different scenarios and corresponding working folder and files.

Scenario 1: mutation rate =0.00001, Ne=100000

Scenario 2: mutation rate=0.00005, Ne=20000

Initial settings: 2 populations, non-overlapping generations, no selection.

For both scenarios, a preliminary run is needed to get a template of allele frequency file with the correct format. This file can be further edited and loaded for the complete run.

*5.2.1.1 Preliminary run (saved in **test2.1-prelim.xml**, see online)*

Genome Tab: 100 independent "Codominant 1" loci, default values otherwise.

Genotype Tab: In the **"Number of alleles/Nuclear loci/Codominant** 1" window, put 2 for the maximum number of alleles per locus. In the "**Allele Frequencies**" windows, choose "**Create**" and **"All loci fixed …"** in the **"…/Codominant 1**" window.

Selection and Demography Tabs: keep default values (no selection case).

Output Tab: choose "0" "**Total number of reproductive events"** and "**From reproductive event # 0"** to **"0"** every **"1**"… in the "**Output # 1**" window. Click on "**Allele Frequencies**" only by unclicking any default summary statistics being asked. Alternatively, ask for the ones that need to be checked initially.

Run Tab: define the working folder (see the **run-prelim** folder in the *.zip provided online for this test), save the *param* file, then click on "**Create"** in the "**Type of result file**" widow, and "**Create**" in the "**Prepare Configuration Files**" window with "**Simple**" (default) run, then in the "**Run…"** windows, ask for 1 replicate (default) and click "**Run**".

Here as in *5.1.1.1*, two allele frequency files are printed in the working folder: the initial Allele_frequency.txt and the **res1_freq_1.txt** that was requested in the *Output Tab* (see also **NB.5.1-b**). Please note that here due to a large number of initial individuals simulated (K=1000 as default value), both files are identical (but see **5.1.1** for a different case). The res1_freq_1.txt file can be edited to start with alternative alleles fixed in each population (here very minor editing is required despite the large number of loci, thus it can be done by hand, see **NB.5.1-a** above) and is renamed **test2.1.loadfreq.2pop.100loc.2al.txt,** to be used in the complete runs below.

*5.2.1.2 Complete runs (available in **test2.1-complete-sce1/sce2.xml** files)*

Genome Tab: 100 loci, mutation rates of 0.00001 (scenario 1) or 0.00005 (scenario 2), so in case of biallelic loci, the probability to mutate into the other allele is the same, the number of alleles being indicated in the *Genotype Tab*.

Genotype Tab: same values than in the preliminary run for allele numbers. In the "**Allele Frequencies**" windows, thus choose "**Load frequencies**" and select file **test2.1.loadfreq.2pop.100loc.2al.txt**.

Selection Tab: no selection

Demography Tab: change **"Equivalent carrying capacity** …" default values to 100000 in each population (scenario 1) or 20000 in each population (scenario 2), and no migration.

Output Tab: ask for a "**Total number of reproductive events**" of 50000 (i.e. generations here), and in the **Output # 1** window, for **"0"** to **"50000"** every **"10" reproductive event(s)**, ask for summary statistics **GstCor** per locus, **Mean GstCor across loci,…** and for **Allele Frequencies**.

Run Tab: define the working folder, clicking on *"***Select/Create***"* (see the **run-complete-sce1/sce2** folders in the *.zip provided online), save the *param* file, and create both the type and conf.txt files. As above, only 1 replicate is sufficient (default) since the 100 loci can serve as replicates in these scenarios.

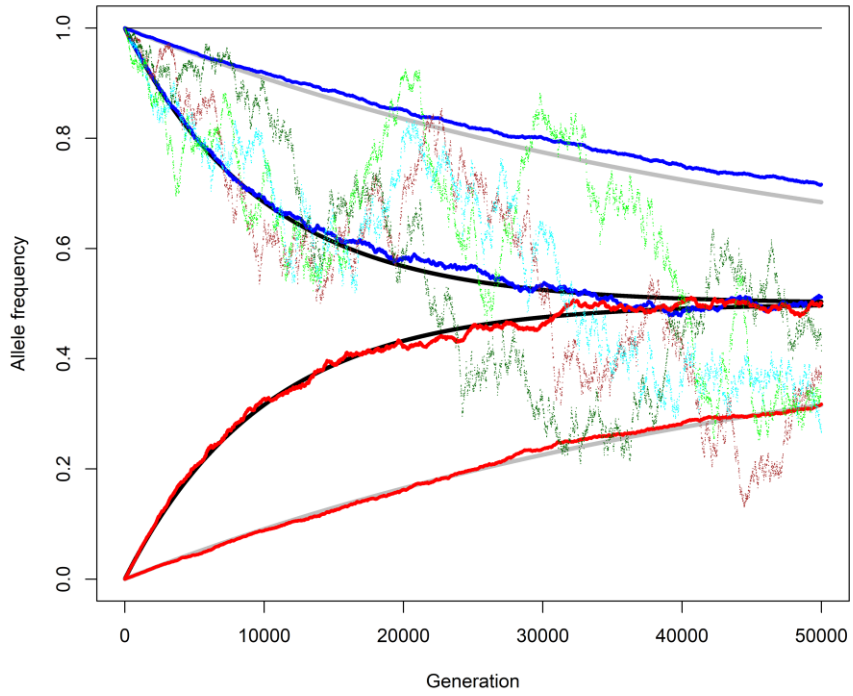
Runtime: ~15 hours for 100000 individuals, 100 loci, and 50000 generations on a Win64 OS for scenario 1; ~3h for 20000 individuals, 100 loci, the same parameter values and 50000 generations for scenario 2; in another example ~3h30 for 1000 individuals, 100 loci, 15000 generations, and same other values with all statistics requested.

Post-treatment:

Click **File**/**Custom plot** in the GUI menu, then type (or copy & paste) the following command in the window:

test2.1.custom.plot.R res1_freq_1.txt test2.1

This script reads the res1_freq_1.txt output files for both scenarios, and plot together both theoretical cases (i.e. the expected theoretical curve for allele frequency defined by the following recurrence formula: p(t) =p(t-1)*(1-m)+(1-p(t-1))*n for n=m (mutation rates), the equilibrium frequency in both populations is n/(n+m)= ½, see more in the R script comments, one path needs to be adapted by the user in this example since output files from both working folder are used), and gives the plot **test2.1.png** (below), which should appear onscreen:



Theoretical curves (black and grey lines) can be compared to the means across the 100 simulated loci in both scenarios (red and blue lines), and the trajectories of a few loci for scenario 2 with an initial frequency of 1 illustrate again the stochasticity of neutral evolution.

See also the default plots in both "**run-complete**…" folders.

Conclusion:

The Poisson distribution allowed producing the level of stochasticity that we need for modeling mutations (see Wakeley 2008, for detailed explanations about the use of Poisson processes).

5.2.2 Dirichlet

Aim: Compare the simulated expected diversity values *He* to the theoretical ones across a range of mutation rates and *Ne* values with frequencies drawn from Poisson-Dirichlet distributions, using the formula:

He = 4Neμ/(1+4Neμ*k/(k-1)), where k is the number of alleles (based on equation 3.70 p. 110 in Ewens 2004), and **μ** the mutation rate.

Building param files *(saved in* ***test2.2.sce1/sce2.xml*** *provided in the tutorial folder online):*

A range of neutral scenarios are simulated across different mutation rates and *Ne* parameter values for the Dirichlet distribution. Therefore, these scenarios are each associated with

different expected *He* means. We used 27 scenarios with 3 different values of *Ne* (10000, 100000, and 1000000) and 9 mutation rates ranging from 0.01 to 10^{-09}. Two examples among those scenarios are described here:

Scenario 1: mutation rate =10^{-4}, Dirichlet *Ne* of 10^{5}.

Scenario 2: mutation rate=10^{-8}, Dirichlet *Ne* of 10^{7}.

Initial settings: 1 population, non-overlapping generations, no selection.

Genome Tab: 1000 (or 9999) neutral independent loci and mutation rate of 10^{-4} (or 10^{-8}) for scenario 1 (or 2),

Genotype Tab: For scenarios 1 and 2, the expected *He* values are respectively 0.9660892 and 0.2848921 if we choose loci with 100 potential alleles and the *Ne* values as above. In order to compare these values to those obtained from simulations with a sufficient level of precision (i.e. below 1%), the issue was to compute the average across a sufficiently large number of possible variant alleles (across all loci) at generation 0. This explains why we used 1000 and 9999 loci for scenarios 1 and 2 respectively.

NB.5.2-b: Please note that we used 9999 loci here for scenario 2, as this is the current limit for the number of possible loci of one type which can be typed in the GUI *Genome Tab.*

Demography Tab: keep all default values for male and female fecundities, and all other parameters in other Tabs, they are not relevant here.

Output Tab: set to "**0**" for the **Total number of reproductive events** (since the counter starts at "0") and "**From reproductive event # 0"** to **"0"** every **"1**"… in the "**Output # 1**" window. Ask for "**Mean statistics across loc (Fis, Ho, He)**" for the Genetic diversity statistics, and unclick all other statistics. The "**Statistics per locus…**" are not needed here except if you want to examine their range.

Run Tab: create the working folders (see **run-sce1/2** folders), save the *param* file, create the type.txt and conf.txt, and click **Run** (which is instantaneous since we need the starting conditions only).

Post-treatment

Across the 27 scenarios, we verified that in the simulations, the mean heterozygosity values across loci (which are printed in one line in the res1_per_pop_1.txt output files) were all within 1% of the theoretical values (by computing He.theory – mean(He.simulated))/ He.theory), as long as the initial expected number of variant alleles was of at least a few thousands (i.e. more loci than for scenario 2 were simulated if needed). This is the case for both examples provided: in the **res1_per_pop_1.txt** files for scenarios 1 and 2, you can observe that the values 0.965686 and 0.283393 for He (averages of 1000 and 9999 loci respectively) are very close to the theoretical values above.

5.3 Tests 3: Drift/migration/mutation equilibrium in a structured population and genetic differentiation estimates.

Aim:
To compare, in a range of different scenarios with increasing migration rates, various estimates of Sewall Wright's F_{ST} statistics:

a) The equilibrium value of genetic differentiation in an island model of population structure (Wright 1951):

F_{ST}=1/(1+4*Nm*) if m is small (named **Th1** in the rest of the text and in the R script for post-treatment described below see **NB.5.3-a)**, *m* being the migration rate and *N* the effective population size.

b) Slatkin and Aoki (1983) theoretical approximation as a function of *Nm* and *d* (number of demes) at equilibrium between mutation and migration, expressing F_{ST} statistics as the ratio of average coalescence times of different pairs of genes in subdivided populations (see also Slatkin 1991, Takahata 1983):

$F_{ST}=1/(1+4Nm*d^2(d-1)^2)$ (**Th2**).

c) Another possible approximation that might be better suited when migration and mutation rates (*μ)* are small and of the same order (Takahata 1983, Takahata and Nei 1983):

$F_{ST}=1/(1+4Nm\mu*d/(d-1)+4Nm*d^2(d-1)^2)$ (**Th3)**.

With:
d) The two computed estimates provided by gMetapop:

G_{ST} as a function of the mean *Hem* and *Ht* (Nei 1973), and **G_{ST}cor** (corrected for the number of demes and named *Fst'* in Nei 1987), *Hem* here being the mean expected diversity across populations and *Ht* the total expected diversity (see also **Table 4.4.2**):

G_{ST}=(*Ht-Hem*)/*Ht*

G_{ST}cor=(*Ht-He*m)*(d/(d-1)/[*Hem*+ (*Ht-Hem*)*(d/(d-1))]=d*(*Ht-Hem*)/(d**Ht-Hem*)

And:
e) The classically used Weir and Cockerham's F_{ST} (**FstWC**) estimate, as computed with the *fastDivPart()* function of the diveRsity R package (for R versions 3.5.0 and above, see **3.9**) using samples of simulated data.

Building param files *(saved in* ***test3-sce1/2/3/4.xml,*** *see the online folder)*

In order to perform this, we simulated a range of scenarios with *Nm* values ranging from 0 to 5, for either 2 or 50 demes (for a total number of 42 scenarios, see **NB.5.3-a**). Examples of *param* files are described below for 4 constrasted scenarios among the 42 (with all files and folders needed to reproduce the tutorial provided online):

Scenarios 1 and 2 with *d*=50 and two different overall migration rates:
m=0.001 and $m=5.10^{-4}$ respectively in an island model of migration, and

Scenarios 3 and 4 with *d*=2, associated with the same two different migration rates.

Initial Settings: two or 50 populations (demes) depending on the scenario, non-overlapping generations, no selection.

Genome Tab: 100 independent **Codominant 1** loci, mutation rate of 10^{-6} (default).

Genotype Tab: 256 maximum and initial allele numbers, Dirichlet distribution with Ne of 10^{+6}.

Selection Tab: No selection.

Demography Tab: ***K*** default values (corresponding to a Ne of 1000 in these conditions), all default values for mating system and potential fecundity parameters.

In the "**Migration models**" window, choose "**Island**" in the "**Migration models/Zygote**" window with *Ne*m*=1 (m=0.001, scenarios 1 and 3) and *Ne*m*=0.5 ($m=5.10^{-4}$, for scenarios 2 and 4), no male gamete migration. You can visualize the matrix of zygote migration rates by clicking on "**Matrix**" in the **Zygote** window This opens a new window from which the matrix can be saved as a text file with the "**Save to File…"** option (for example in the

test3.sce1.mig.rates.txt file, which can be modified and reloaded using the "**Load File…"** option, if needed).

Output Tab:

Choose 10000 "**Total number of reproductive events**" (= generations here since there are no overlapping generations), which is likely sufficient to reach equilibrium conditions for all tested scenarios.

In the "**Output #1**" window, ask for "**From reproductive event" "0" to** event **"9900" every "50" "reproductive event(s)"** to get an overview of the evolution of diversity and differentiation with default plots across generations in each scenario. Therefore, ask for all "**Genetic diversity"** statistics except "**Allele frequencies"**.

Then in the **"Output #2"** (illustrating here the flexibility of output options at different time steps across one simulation), ask for **"From reproductive event" "9901" to "10000" every "1" "reproductive event(s)"** to later compute a mean across the last 100 generations (*a priori* close to the expected equilibrium). Ask for "**Extracting sample…"** of 50 individuals and click for the "**Genepop** [software] **format"** as well (only one example Genepop file was left in the online *.zip for **scenario 1**), and also **Gst** and **Gstcor** statistics here.

Run Tab: for each scenario, "**Select/Create**" the corresponding working folder (**run-sce1/…sce2/…sce3/…sce4**), and click on both "**Create"** in the "**Type of result file**" window and in the "**Prepare Configuration Files**" window with the "**Simple**" option (default). Ask for "**1**" replicate in the "**Run gMetapop**" window in each scenario, the 100 loci and the last 100 generations serving as replicates here, and click "**Run"**. See all the result files in the four different working folders **run-sce1/sce2/sce3/sce4**.

Runtime: For the 21 scenarios and 50 demes (resp. 2 demes), in total, it took ~30 min (resp. ~1 min) per scenario on a Win64 OS (if all statistics are being asked from the start at every 50 generations, except allele frequencies, Gst and mean Gst, with also Genepop formatted sample files being requested for the last 100 generations). On a Linux64 OS **(e.g.** Ubuntu 18.04.3 LTS wih a less efficient processor Intel(R) Xeon(R) CPU E5-2643 0 @ 3.30GHz, hereafter **Ubuntu64**), the whole range of 42 scenarios took ~24h to run in batch.

NB.5.3-a: In order to perform the 21 scenarios in batch for a particular *d*, we used the "**Multiple"** conf.txt option (either for the "2d" case or the "50d" case, choose "**Multiple"** in the "**Prepare Configuration Files"** window of the *Run Tab,* using "**21**" in the "**Number of configuration files per run**" window of the *Output Tab*, and "**create**" the type.txt that will be common to all scenarios, then "**Prepare run and create conf.txt files**". Notice all the **../simuX** subfolder, each corresponding to the same conf.txt at this stage). The zygote migration rate values in each conf.txt file are next automatically changed with the **test3_batch.run.gM_Stat.comparison.R** script provided (see part # **A)**, which can be launched from the working folder of the multiple conf.txt option, either from an R editor or in command line; all steps being described in the script for producing the different initial conf.txt before running simulations. Once # **A** has been done, check the new zygote migration rate values in the conf.txt from some of the 21 scenarios. All simulations can then be launched in command line mode, either under windows using the *.bat file, or under linux with the *.sh file (see *3.8.7.4*), or from the R editor (see part # **B** of the script), or in this case directly from the GUI. Each scenario or groups of scenarios could also be launched independently on different nodes of a cluster environment, provided that the initial random seeds are different.

See all available files in the subfolder "**.../Ex-multiple-conf-2d-NB5.3-a"** of **tests 5.3** provided online. Result files from the full simulation are only provided in **../5.3_50d/simu1** and **../5.3_50d/simu4** (corresponding to 2 of the 21 migration scenarios) due to space constraints, but all simulations can simply be redone by loading **multiple-conf-2d.xml**. For the 50 demes case, all steps from the start can be reproduced in the subfolder "**5.3_50d**" (empty online, except for the final result files and plot), by loading the **multiple-conf-50d.xml** param file, launching the simulations, and reproducing the R script. Both final results files res2d/50d.txt are available in the main folder to reproduce the plots (**#D** step in the script).

Post-treatment:

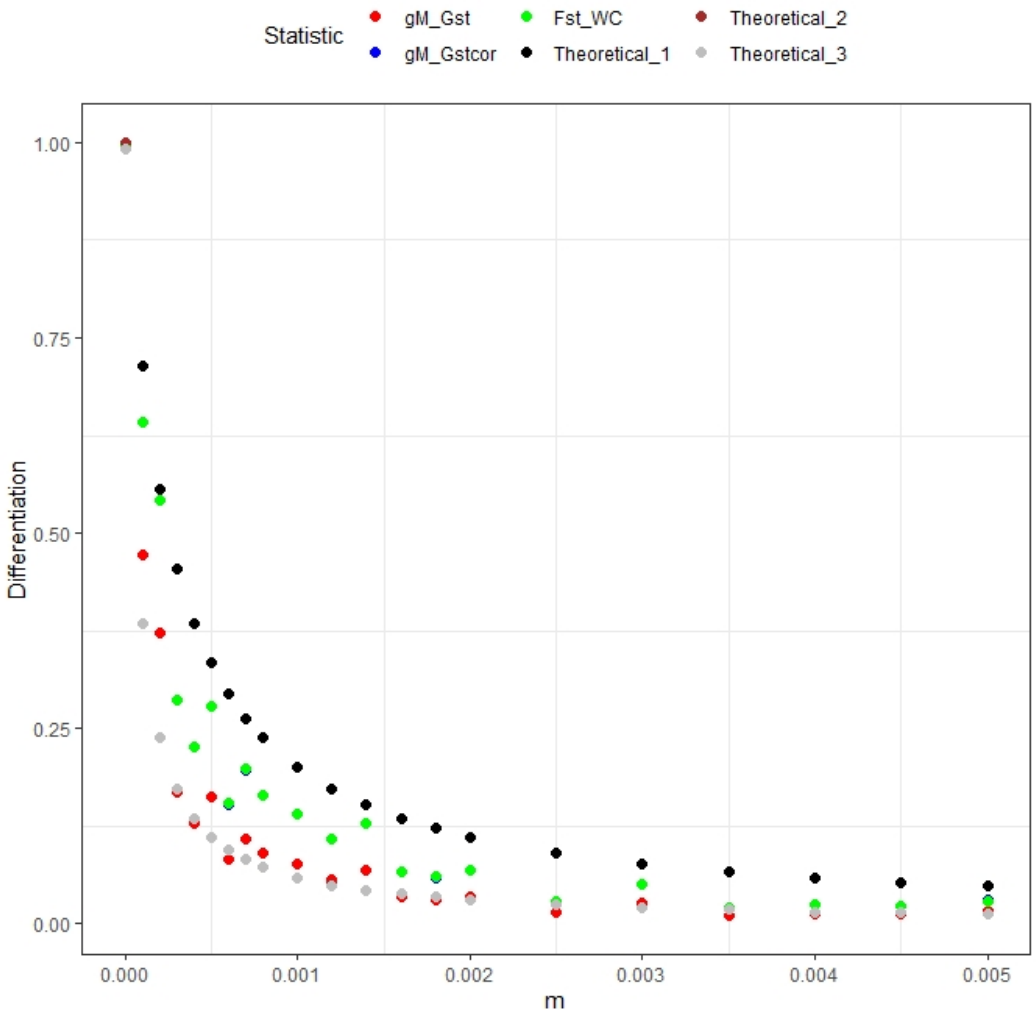
First, you can compare the default plots across the four example scenarios, and notice how equilibrium levels of differentiation vary with different migration rates. In order to make the plots, click on "**Default plot**" and examine the **plot.perGen.jpg** (or *.pdf) files in each working folder (see **NB.5.1-l** to activate the plot options in the GUI).

Additional analyses are described in the script **test3_batch.run.gM_Stat.comparison.R** (in parts # **C** and # **D**, see **NB.5.3-a** above) to compare results of the simulations across the 21 scenarios:

- Computing **Gst** and **Gstcor** means across the 100 loci, the total number of simulated individuals (i.e. ~1000), and for the 100 last generations,
- Computing theoretical estimates across scenarios with different migration rates,
- Using sample files for computing **FstWC** estimates,
- Computing correlations among all estimates of F_{ST} for both 2 and 50 demes.

The plots and correlations among all estimates below illustrate the full range of 21 scenarios, **Th2** and **Th3** can't be distinguished in both 2 and 50 demes cases, due to a mutation rate that is two orders of magnitude smaller than m, except for m=0).

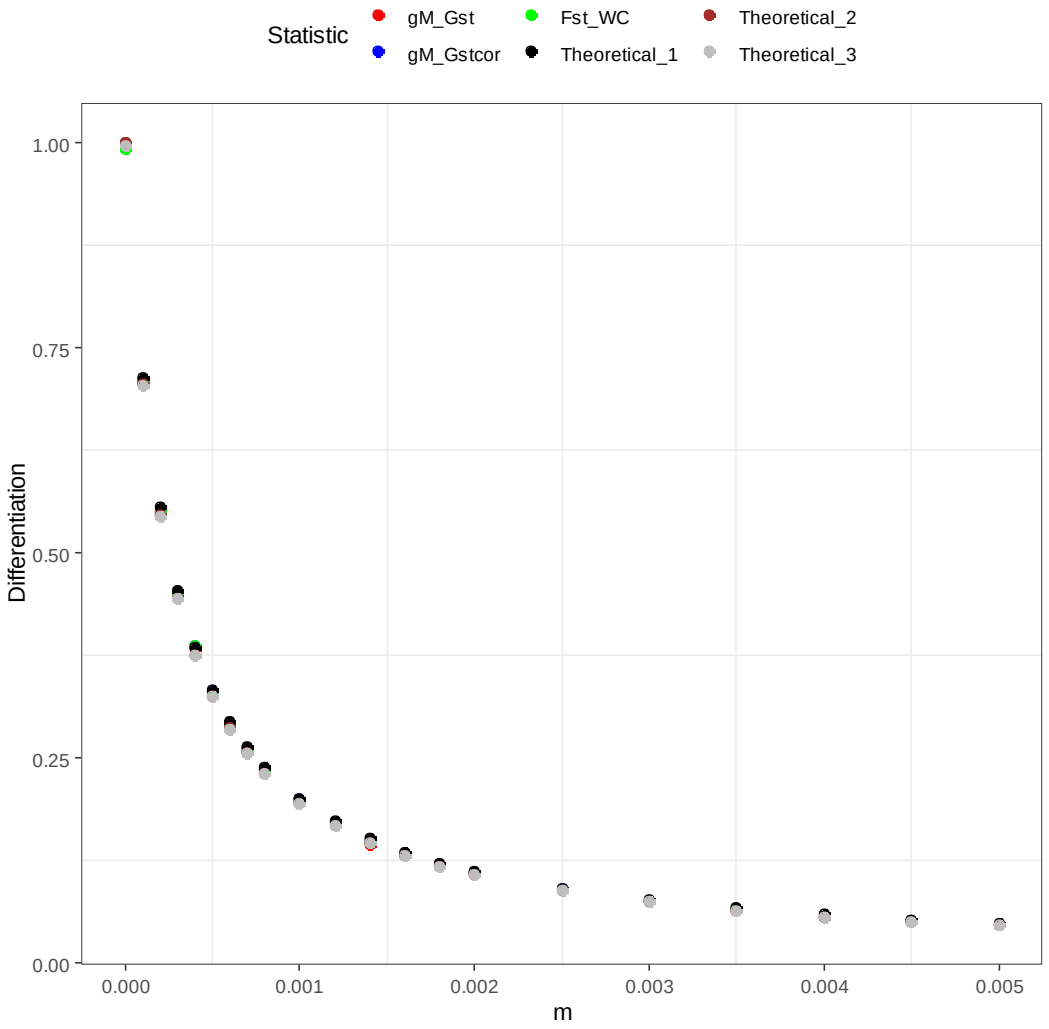
For 2 demes: plot **test3_2demes.jpg**



	Gst.2d.mean	Gstcor.2d.mean	Fst_WC	Th1	Th2	Th3
Gst.2d.mean	1	9.84E-15	9.98E-15	8.54E-11	7.90E-17	6.53E-17
Gstcor.2d.mean	0.9798	1	9.37E-52	1.50E-15	1.34E-10	1.20E-10
Fst_WC	**0.9797**	**1**	1	1.39E-15	1.33E-10	1.19E-10
Th1	0.9467	0.9834	0.9836	1	4.84E-09	4.40E-09
Th2	0.9879	0.9441	0.9441	0.9175	1	5.76E-52
Th3	0.9881	0.9447	0.9447	0.9183	1.0000	1

Pearson correlation coefficients are below the diagonal and associated test *P*-values above the diagonal.

For 50 demes: the plot **test3_50demes.jpg**



	Gst.2d.mean	Gstcor.2d.mean	Fst_WC	Th1	Th2	Th3
Gst.2d.mean	1	8.18E-44	6.77E-44	1.51E-42	4.60E-39	7.41E-40
Gstcor.2d.mean	1	1	4.32E-70	6.31E-43	9.75E-36	2.44E-36
Fst_WC	1	1.00E+00	1	7.50E-43	9.48E-36	2.38E-36
Th1	1	1.00E+00	1.00E+00	1	5.15E-38	6.82E-39
Th2	0.9999	1.00E+00	1.00E+00	1.00E+00	1	6.24E-57
Th3	1	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1

Conclusions:

a) Compared to theoretical expected values, all computed differentiation estimates from simulated data (**Gst, Gstcor** and **FstWC**) perform well, and their correlations are all very close to **1** when the number of demes is 50 (as also seen in the **test3_50demes.jpg** plot).

   This is consistent with the original Weir and Cockerham (1984) study. For a smaller number of demes (2 here) with the same conditions of computation, **Gst** correlates better with **Th2** and **Th3,** which both account for and thus limit bias due to a small *d* number, as illustrated in the plot **test3_2demes.jpg** shown previously.

NB.5.3-b: WARNING: please note that more scenarios could be explored for comparing genetic differentiation statistics, and the above conclusions might be different with different initial conditions: for example if mutation and migration rates were of the same order of magnitude, or if using much smaller population sizes and thus larger genetic drift, or if using smaller sample sizes that would affect statistics harbouring large sampling variances.

b) Also, using the scenarios and conditions here, **Gstcorm** has a correlation of ~1 with **FstWC**. This has also been verified in very different cases (for example with much lower ***K*** or Ne values) and is consistent with Nei's (1987) demonstration that *Fst'* (i.e. the gMetapop **GstCo**r) and **FstWC** are very close estimates.

5.4 Tests 4: 1D and 2D Stepping-Stone migration models

Aim: test the predictions of the one-dimensional and two-dimensional Stepping Stone migration models for estimating dispersal variances from the relationships between physical distances and F_{ST} (Kimura and Weiss 1964, Weiss and Kimura 1965, Rousset 1997, and see *Post-treatment* below).

Building param files *(saved in the* ***test4-SS….xml*** *files descrived below)*

Three scenarios:

- One-dimensional stepping stone with 10 populations (**1D-10pop**): **test4-SS-1dir-10pop.xml**.
- One-dimensional stepping stone with 100 populations (**1D-100pop): test4-SS-1dir-100pop.xml.**
- Two-dimensional stepping stone with 10x10 populations (**2D-100pop): test4-SS-2dir-10x10pop.xml**).

If you want to redo the *param* files, follow the description below:

*Initial settings***:** 10 or 100 populations, non-overlapping generations, no selection.

Genome Tab: 10 independent loci, default values for mutation rates.

Genotype Tab: 100 alleles (default values for maximum and initial numbers). All populations start with equal frequencies.

Selection Tab: no selection.

Demography Tab: Each population has a size *N (or K)*=1000 individuals (default values). Migration rates are always set at 0.025 (("**Migration models/Zygote/1D (or 2D) stepping stone**", no **Male gamete** migration). Use 10 rows and 10 columns for the 2D Stepping Stone. Default values are used for fecundity rates parameters.

Output Tab: Populations are simulated for 10,000 generations. We are only interested in this last generation, as we want to observe equilibrium patterns, but to illustrate the use of one of the R scripts for the **File/Custom plot** option below, we ask in the "**Output #1**" window for "From reproductive event" **"0"** to **"10000"** every **"5000" "**reproductive events" (for the **1D–10pop**. scenario) Click on "**Extracting sample files…"** for 50 individuals per population for this last generation (sample files will be provided for generations 0, 5000 and 10000, as requested in the corresponding type.txt file), then click for "**Genepop format"** (needed for the post-treatment below).

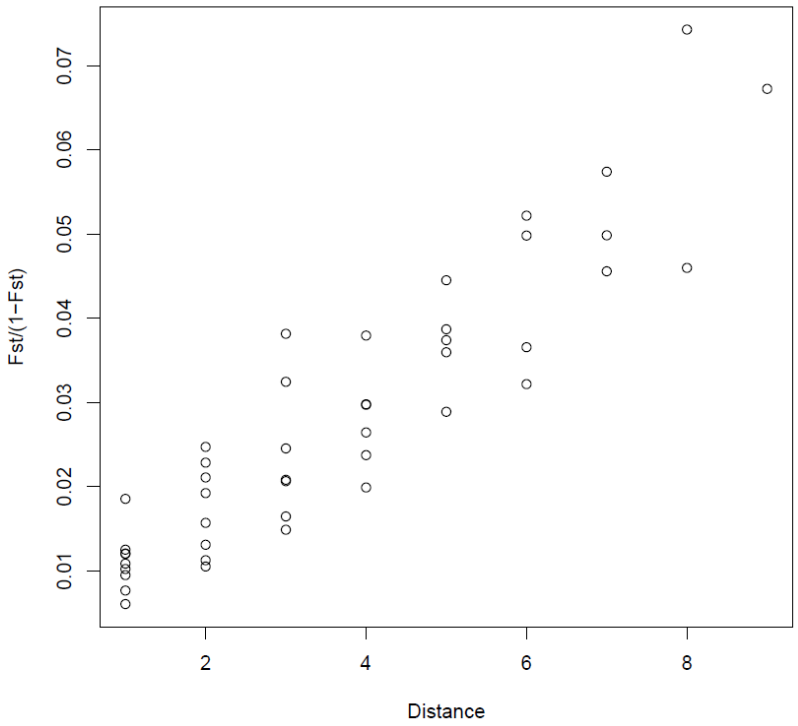
Run Tab: type **100 replicates** in each scenario for testing the predictions (see the working folder **test4-1D-10pop** with all replicates, and 1 replicate is provided online for the other 2 scenarios, they can be redone easily).

Runtime: scenario **1D–10pop**: less than a min per replicate on a Win64 OS so ~ 1h15 for 100 replicates; scenario **1D–100pop**: ~7 min per replicate on a Win64 OS; scenario **2D–100pop**: ~15 min / replicate on a Win64 OS.

Post-treatment:

Examples of R scripts illustrate how to plot results for one or more replicates in the different scenarios. First, from the **1D–10pop** working folder, use the "**File/Custom plot**" option in the GUI menu (see **3.9.2** for more information on Custom plots). In the pop-up window, type the following command (copy and paste the **bold text**):

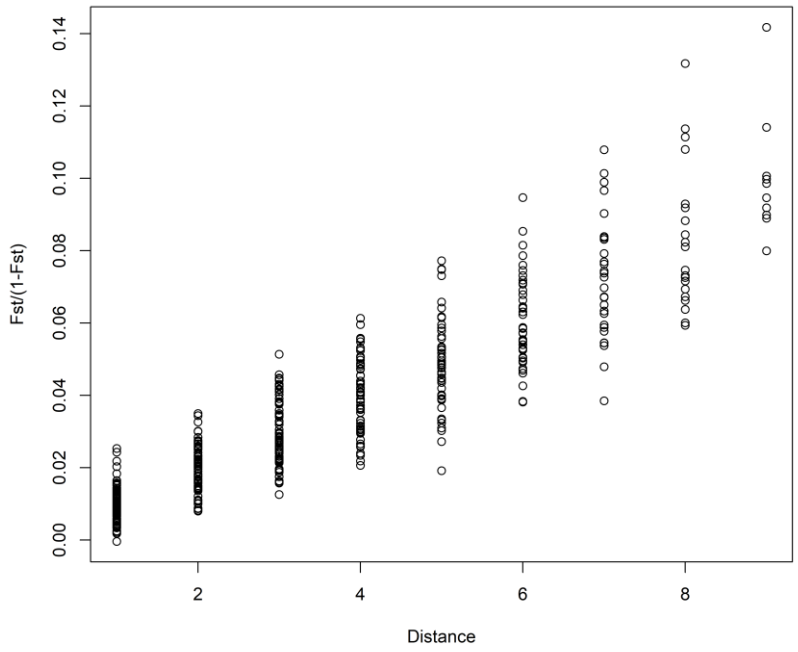
test4_1D.custom.plot.1rep.only.2Arg.r (*R script, needs to be copied in the working folder beforehand, and more information is provided in the script comments*) **res1_sample_1_genepop_neutral_10000.txt** *(i.e. compulsory first argument = character string refering either an output file from gMetapop or one created by the R script, located in the working folder*: *here for this script, it is a sample file for replicate 10000 in the Genepop format)* **custom10000** *(i.e. compulsory second argument = always the name of *.png plot opened via the GUI, file extension not needed)*. You obtain the following plot:



In order to illustrate the flexibility of the "**File/Custom plot**" option, here is another script with more arguments, which allows processing more than one replicate (**test4_1D.custom.plot.Xrep.4Arg.r**). From the same **1D–10pop** working folder, a similar command line than above can be used with 2 additional arguments (copy and paste the **bold text** below in the pop-up window):

test4_1D.custom.plot.Xrep.4Arg.R output.txt custom10rep 10 *[optional 3rd argument, here the number of replicates processed]* **5000** *[optional 4th argument here, the number of reproductive events at which sample files are requested]*

Thus in this example, sample files from the first 10 replicates of generations 0, 5000 and 10000 for the 1D-10pop case are being used (see them in folder **.../run-1D-10pop**). This gives the following plot:



The script **test4_2D.custom.plot.Xrep.4Arg.r** is also provided for visualizing results of the **2D-100pop** scenario (more information on how to use it in the script comments).

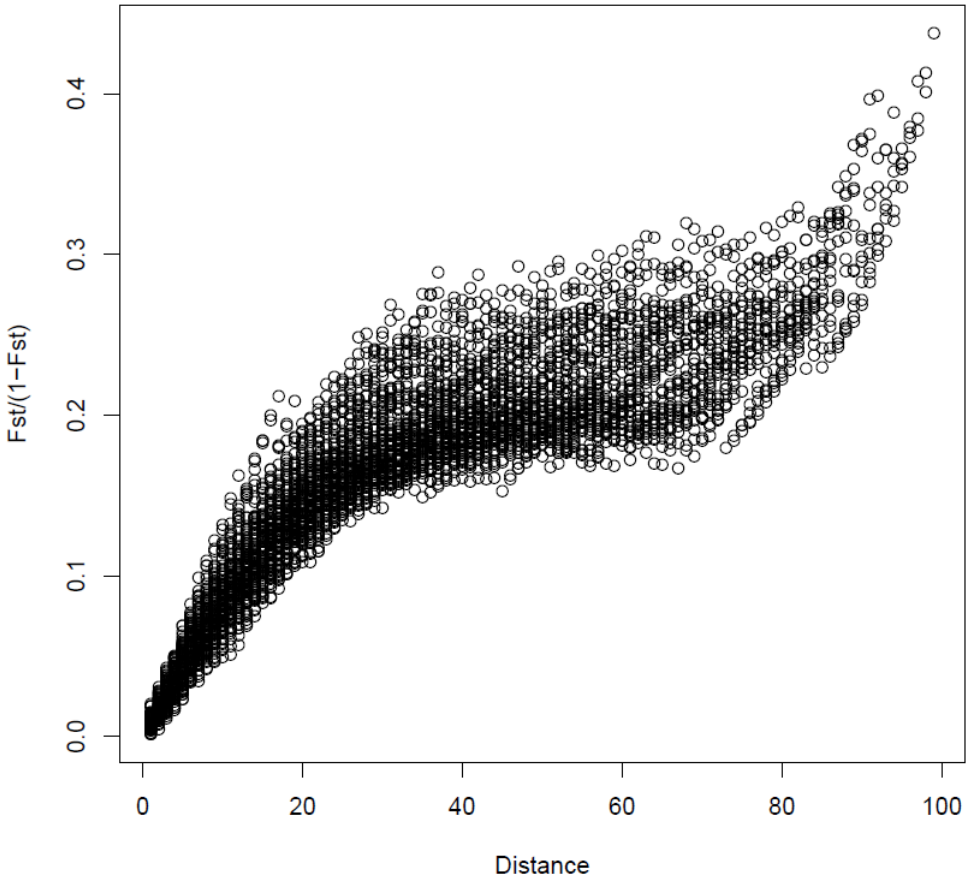
R scripts above load the genotype data sample files (which are in the Genepop format and have been printed across generations), and compute Weir & Cockerham (1984) pairwise *Fst* values among populations with the package diverRsity (for a R version >= 3.5.0). Then, following the method of Rousset (1997), the Fst/(1-Fst) is regressed against the distance between pairs of populations (assumed to be 1 between adjacent populations) for the **1D-10pop** scenario, or against the logarithm of their distance for the **2D-100pop** scenario. The slope (*b*) of this regression is computed, and the axial variance of dispersal (σ^2) is also obtained as $N\sigma^2 = 1/(4b)$ for the 1D-10pop case ($N\sigma^2 = 1/(4\pi b)$ for the 2D-100pop case). In the 1D case, it is predicted theoretically that $\sigma^2 = m$, so we should directly estimate *m* when we estimate σ^2. In the 2D case, it is expected that $\sigma^2 = m/\pi$ (Austerlitz, unpublished).

Averaging over 100 replicates for each case, this leads to the following results:

Model	estimated	Predicted
1D stepping stone 10 populations	0.0277	0.025
1D stepping stone 100 populations	0.0946	0.025

2D stepping stone 10x10 populations.	0.00756	0.00796

The figures **test4-[1D-10pop/1D-100pop/2D-100pop]-Xrep.pdf** show results for **X** replicates in each scenario. The discrepancy between the estimated and predicted values for the **1D-100pop** scenario comes from the loss of linearity for long distances. This illustrates that the regression leads to a good prediction only over shorter distances (see the plot in test4-1D-100pop-1rep.pdf below):



5.5 Tests 5: strong selection on genotypes

Aims : Examine the evolution of advantageous alleles in additive and dominant models (**5.5.1** below), or in the recessive model (**5.5.2**), given that the alleles are initially rare. Compare results to the theoretical and deterministic models described in Hartl and Clark 2007, see their Figure 5.3 p. 213).

5.5.1 Additive and dominant models (scenarios **1** and **2)**

Building param files:

In this example, we need a **preliminary run** for creating the correct formats for the genetic map, allele frequency or genotype files (see *3.2.3.2*, **3.3.2** and **NB.5.1-c**). These files will be loaded at the start of the full simulation. The required scenarios can be simulated separately in one (or more) populations, and for one (or more) loci, both serving as replicates for the same evolutionary trajectories. Alternatively, both scenarios can be simulated in a single run

in isolated populations, using the same configuration file, which is what we chose here. This can be done in different manners:

a) Either choose different sets of independent loci under selection with different selection coefficients depending on each scenario, and use replicated populations. Thus, in each population, both selection models are simulated together at different loci across genomes with similar genetic maps,
b) Or choose two different selection models in different populations, some loci under selection in some populations thus behaving as neutral loci in others. We chose this latter configuration to illustrate the flexibility of the GUI in setting up configuration files: here two sets of 10 loci are considered as potentially under selection, one set being submitted to one selection scenario while the other one remains neutral, and the other set being submitted to the other type of selection scenario in the other populations. An additional set of 10 "**Codominant 1"** loci serves as the same neutral reference in both scenarios. We use, therefore, 30 loci in total.

NB.5.5-a: In theory, a) and b) are slightly different since the selection strength at one group of loci in a) can affect the overall diversity and population effective sizes, and thus the selection efficiency across all loci. However, for large population sizes and independent sets of loci, the assumption that b) can be approximated by a) is realistic and could permit simpler configuration files, given that all loci would be defined with similar numbers of alleles.

Five populations are simulated per scenario, in order to increase the number of replicates (see the preliminary run below and/or when all files to load have been created).

*5.5.1.1 Preliminary runs (saved in **test5.1-prelim-geno.xml**)*

You can either go directly to the *5.5.1.2* **Complete run(s)** below, using the **test5.1-prelim-geno.xml** that is provided online, or follow the steps for creating the file:

Initial Settings: 10 populations, non-overlapping generations, selection on genotypes.

Genome Tab: 30 loci, 20 loci under selection with the default mutation values (10^{-5}), and 10 neutral loci (**Codominant 1** window) with default mutation value (10^{-6}). Please note here that we allow mutation at loci under selection in order to counterbalance the loss of rare alleles by drift when starting from very low initial frequencies (see **option B** below in the initial genotype file). This is not the case in the theoretical model.

NB.5.5-b: You can first "**Select/Create"** the working folder in the *Run Tab*, which is then memorized into the GUI, for automatically pointing to this folder or the folder above when saving or loading any type of files.

Click on the "**Save map"** option in order to name the file defining the relative positions of the different types of loci, i.e. the **genetic map** file, and save it in any folder (preferably the working folder or the folder just above, which can be created beforehand or within the GUI in the *Run Tab*). Here we named this map file **test5.1-map.txt**, available in the folder online.

NB.5.5-c: As long as the format of the map file is correct, and its consistency with the corresponding genotype files is conserved, one can edit it and change values of recombination rates between loci for particular scenarios.

Genotype Tab: keep default values (**"2"** and **"2"**) for maximum and initial allele numbers' windows for the "**Loci under selection**" and type "**10**" and "**10**" for the common set of neutral "**Codominant 1**" "Nuclear loci" in the same windows.

NB.5.5-d: For the sake of the example, we use different allele numbers for loci potentially under selection and for "**Codominant 1**" neutral loci, since it will be easier to distinguish both types of loci in genetic map and genotype files, if needed. This also means that the 10 loci under selection in one scenario, but behaving as neutral in the alternative one (i.e. their genotypes harbouring the same fitness), are also biallelic in this example, and will start from different initial conditions (see below), compared to the 10 "**Codominant 1**" loci across both scenarios, which are set up with 10 alleles.

Use "**Create"** in the **"Allele frequencies"** window**,** and for the loci under selection, choose "**All loci fixed …**" (by default, allele "1" is the one that is fixed). This impacts the **Initial number of alleles per locus**… which goes back the value of "1". For "**Codominant 1"** loci, choose the option "**All loci with allele in equal frequencies**".

Selection Tab: keep default values of **"1"** for this preliminary run for the "**Matrices of fitness values**…", i.e. there is no fitness differences among all genotypes.

Demography Tab: use "**compute**" in the "**Initial number of individuals…"** window and edit ***K*** to values with 10+04 in the "**Equivalent carrying capacity** …" window (or load **test5.1.K10000.10pop.csv)**, in order to limit genetic drift effects. No migration.

Output Tab: choose 0 "**Total number of reproductive events**", and then "From reproductive event" **"0"** to **"0"** every **"1**"… in the "**Output # 1"** window. Unclick any default summary statistics, and ask for "**Allele frequencies"**, and any other statistics that would need to be checked initially.

Run Tab: Define the working folder (**…/5.5.1/run-prelim)** here**,** if it has not been created or chosen previously). In the "**Genotypes**" window, click on "**Final genotype filename**" and provide a name suffix for the file (**test5_1_loadgeno** here, or keep default and rename it afterwards). Save the *param* xml file (**test5.1-prelim-geno.xml** here). Then click on both "**Create"** in the "**Type of results file**" window and in the "**Prepare Configuration Files**" window with "**Simple**" (default), before launching the "**Run**".

As a result of this preliminary run, two output files have been created in the working folder: one allele frequency file (**res1_freq_1.txt),** and one genotype file (**test5_1_loadgeno1.txt**) with the correct formats for being used as initial conditions for complete runs:

a) For the complete run to start with initial allele frequencies (**option A**), use **res1_freq_1.txt.** We choose here to edit this file by setting the initial frequency of rare advantageous alleles (i.e. allele "2" in loci potentially under selection) to 1%, in order to reduce the stochasticity *a priori* in the first selection steps. Simply open the file in a text editor, and change all "**1.000e+00"** values into **0.99** and all "**0.000e+00**" values into **0.01 (**see also **NB.5.1-a** for more on editing input files). Please check that "0.01" values are for allele "2" and that you replaced 200 values each time. We renamed the file **test5.1.loadfreq.10pop.p0.01.txt.** You can check both files in the **…/run-prelim** folder.

b) For the complete run to start with an initial genotype file (**option B**), **test5_1_loadgeno1.txt** can be used as it is**.** Rare alleles are the ones under positive selection (see the loading of fitness values below), and the genotype file content depends on the initial "**Create**" conditions for allele frequencies that were chosen before in the *Genotype Tab.* You will either observe that the rare allele is absent (if the option "**All loci fixed for one allele**" was chosen with a mutation rate at zero chosen in the *Genome Tab*), or that very few copies of the rare allele are present (depending on the mutation rate chosen, default being 10^{-5}). This file can be easily edited by adding more heterozygotes for example, compared to the few that might already be present.

NB.5.5-e: Due to one of the density dependent regulation added to the model, the initial number of individuals will likely be smaller than the theoretical carrying capacity K for large population sizes (see **NB.3.5-a**, and the genotype file **test5_1_loadgeno1.txt** created at generation 0 where exact numbers are printed in line 4). Either these numbers suits the user, or they can be adjusted manually by increasing the K values, and redoing the preliminary run until exact numbers are obtained in all populations, or these exact initial numbers can be chosen for files with correct formats using scripts if needed. For this test, the initial numbers of individuals in **test5_1_loadgeno1.txt** are large enough and appropriate.

When the files described above have been produced, **complete runs** can be performed as follows. Please note that if the files described are recreated from scratch, they will likely

correspond to a different genetic map, thus the corresponding frequency or genotype files need to be used also.

5.5.1.2 Complete runs

Here, you have three possible ways to launch the simulations.

1) Either load one of the full *param* files which are available online:

test5.1-complete-loadfreq.xml for **option A**, or

test5.1-complete-loadgeno.xml for **option B.**

then the parameter values can be checked in the different GUI Tabs. The paths to the genetic map file (**test5.1-map.txt),** and either the initial frequency file (**test5.1.loadfreq.10pop.p0.01.txt, option A**) or the initial genotype file (**test5_1_loadgeno1.txt, option B**) also need to be modified by reloading these files (provided online) in the *Genome* and *Genotype Tabs* (in order to fit the user's path organization). Then you can go directly to the *Run Tab* and launch the simulation.

2) Or follow the instructions below to reproduce similar *.xml files, since the genetic map produced in the preliminary run will likely be different as it involves a random process for assigning loci positions in the GUI. For reproducing the exact same **test5.1-complete…xml** *param* files provided as examples, the user simply needs to also load the corresponding genetic map file (**test5.1-map.txt)**, which is provided online.

In the GUI menu, use the **File/Open Interface parameter file** option in the GUI File menu and load the **test5.1-prelim-geno.xml** file in order to further change it for the full run,

3) Or use the **File/New Interface parameter file** option, and select the same initial settings than in the preliminary run (10 populations, and selection on genotypes).

The third choice is recommended here as it allows using the **recurrence rule** for filling up automatically fitness values for the different scenarios (see **3.4.3**).

Genome Tab: 20 loci under selection and 10 Codominant 1 neutral loci, use default mutation values. Then load the genetic map **test5.1.map.txt** (i.e. which was previously saved in the preliminary run) since there are more than one type of locus that is simulated (see **options A** and **B** below). Use the "**Load map**" option in the "**Nuclear loci genetic map**" window, and click on "**Load file…"**

Genotype Tab: choose 2 and 2 in the "**Loci under selection**" window, and 10 and 10 in the "**Codominant 1**" Nuclear Loci windows.

For **option A**: choose the "**Load frequencies**" option, click on "**load file"** and select the file previously created in the preliminary run (i.e. **test5.1.loadfreq.10pop.p0.01.txt**)**.**

For **option B:** Similarly than for option A, the genetic map file needs to have been loaded first in the *Genome Tab*. In the **"Allele frequencies**" window, use "**Load genotypes**" and choose the file generated in the preliminary run above (i.e. **test5_1_loadgeno1.txt).**

NB.5.5-f: Here the genetic map that was loaded in the *Genome Tab* defines which loci are under selection, and initial genotypes or allele frequency files at corresponding loci need to be consistent with the map in terms of number of alleles for example (which can vary across different types of loci**). This will be the case if all initial file formats are obtained in the same preliminary run**, as is shown in this tutorial.

Selection Tab :

Use the default "**soft"** selection (population size independent of mean fitness, see **3.4.4**) where selection is performed on fecundity.

For a fitness difference of 10% between 11 and 22 genotypes in scenarios 1 and 2, where allele "2" is the favourable and rarer allele, fill the "**Genotype fitness values"** as follows**:**

a) Click on "**Matrix(ces) of fitness values for user editing"** where all default values are "1" (no selection case).

b) Then, either modify these values by hand (*i.e. following option b)* under the part *Building param files* of **5.5.1:** *change fitness values of genotypes 1-1 to 0.9, that of genotypes 1-2 to 0.95 for the first 5 populations (additive model), then click "OK" for the next 10 loci, then do the same for populations 6 to 10 using fitness values for the dominant model – only genotypes 1-1 have a value of 0.9, all other genotypes in populations 1 to 5 and other genotypes have a value of "1"- click OK for loci 11 to 20*), or load a file with genotype fitness values (recommended) for one locus across a set of populations (described in c) below). Once values are modified, the rest of the loci are automatically filled with the same values by default (**recurrence rule**).

c) For the additive scenario for example, load **test5.1-Wval-additive-10pop.txt**, click "**OK**" for the next 10 loci that have been filled by the **recurrence rule**,

d) then at locus 11 for the dominant scenario, load **test5.1-Wval-dominant-10pop.txt** for scenario 2, and the rest of the loci are being filled with the same values by the same rule, no need to check them here (but you can if you want to by clicking "OK" which gives the following locus matrix), then close the window with the **X** at the top right of the window.

NB.5.5-g: Alternatively, for cases with a much larger numbers of loci under selection, the lines of the fitness values in the final configuration file (conf.txt) could simply be replaced by a small R script. See for example **Tests 5.3** for an R script modifying migration rate values in conf.txt files.

You can also load the **test5.1-complete-loadgeno/loadfreq.xml** files provided and check the fitness values, if this part seems a bit complex the first time.

NB.5.5-h: Please note that the numbering of loci under selection, when assigning fitness values in the *Selection Tab,* corresponds to the order in which they appear in the genetic map (randomly drawn by default, see the conf.txt). This can be useful for extracting loci submitted to different types of selection (see the "**Custom plot"** R script provided below in *Post-treatment*). If loci under selection need to be located in particular positions compared to other loci, this can be controlled beforehand by loading a user-defined genetic map, and then corresponding user-defined genotype or allele frequency files produced in a preliminary run.

Demography Tab:

Change K to 10+04 in the "**Equivalent carrying capacity** …" window (or load **test5.1.K10000.10pop.csv)**. Default values are kept for all other parameters.

NB.5.5-i: Here you can observe that in the case of a loaded genotype file, it also controls the "**Initial number of individuals…"** in the corresponding window of the *Demography Tab*, which is thus not available anymore in the GUI.

NB.5.5-j: The carrying capacity values (*K*) chosen in the GUI can differ from the initial number of individuals (*Indnb*) present in the file that is loaded (option "**Load genotypes**" in the *Genotype Tab)*. Either the regulation will operate after the first generation if *K* is lower than *Indnb*, or the population will grow if it is higher, since the file loaded has the priority to set the initial conditions. If a population is absent in the loaded file, its *indnb* will be set to 0.

Output Tab: Ask for "**1000**" reproductive events in total, with an event step of "**5**". Ask for genetic diversity statistics ("**GstCor per locus**", see **Tests 5.3** for more details on differenciation statistics; "**Mean GstCor across loci…"; "Fit, Ht, Fism, Hem per locus"; "Allele Frequencies",** not to forget since the frequency output file will be used for the "**Custom plot**" below**),** and for all "**Demography**" statistics.

Run Tab:

Define the working folders **(…/5.5.1/run-complete-loadfreq** for **option A**, and **(…/5.5.1/run-complete-loadgeno** for **option B**). For **option B** only, the name of the loaded genotype file

should be seen in the **"Initial Genotype filename"** window with “NOT CHECKED”. Then format checking is performed when creating the conf.txt file.

Save the *param* file.

Click on “**Create”** in the “**Type of results file**” window, then on “**Create**” in the “**Prepare Configuration Files**” window with “**Simple**” (default), and look for “CHECK OK” for the initial genotype file above for **option B**, since a basic check of its format is performed when creating the conf.txt file. See more information in the Appendix online about error messages which can be issued depending on format problems.

Number of replicates : here one replicate can be used with the 10 loci per type of selection in 5 populations serving as replicates, since they have the same fitness values. Alternatively, 10 replicates per configuration file can be used instead of 10 loci under selection for each scenario, but the R script provided for visualising the data with a custom plot would have to be modified.

Launch the simulation by clicking on “**Run”**.

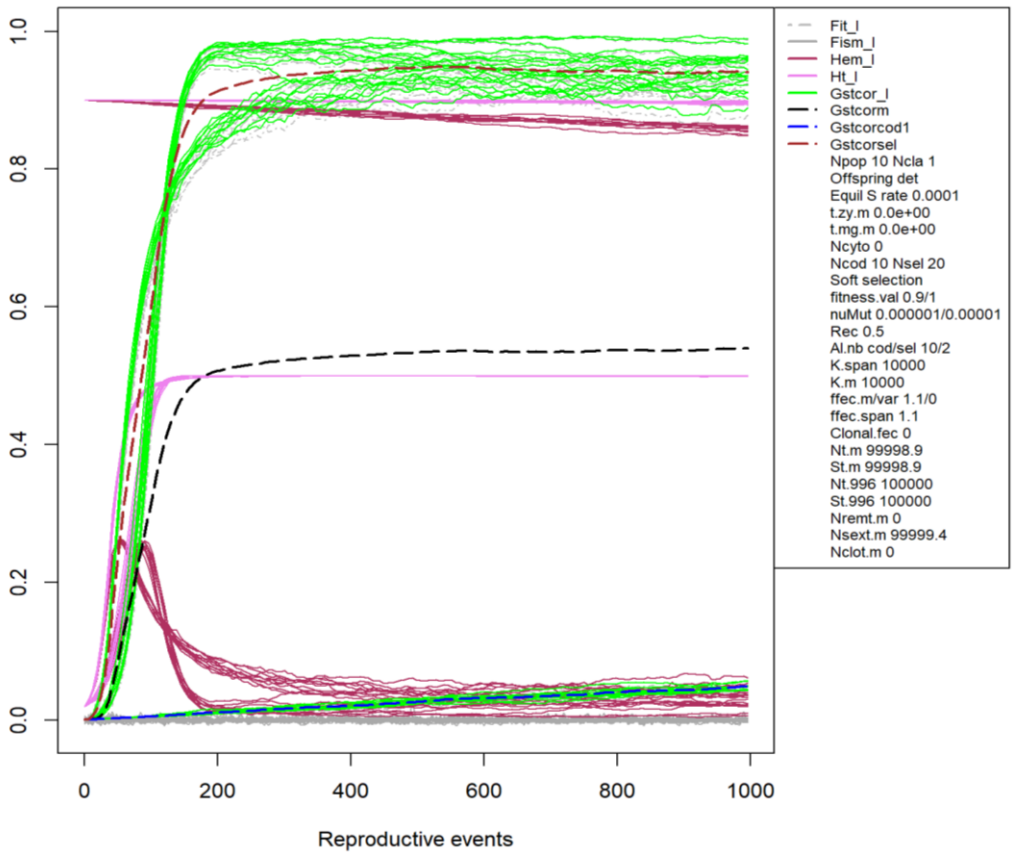
Runtimes:

for the complete run with both scenarios together: ~5 min on a Win64 OS for 1000 generations, 10+4 individuals per population in 10 populations (so 100000 individuals in total), 30 loci in total, with 10 alleles at 10 neutral loci (see folders **.../5.5.1/run-complete-etc…**).

Post-treatment (see also **NB.5.1-k**)*:*

Use **File/Default plot** in the GUI menu and see the default plots **plot.perGen.png** for both options, which are rather complex since both scenarios are seen together, but differentiation statistics for selected loci are consistent with the increase of selected allele frequencies, until their quasi fixation in some scenarios.

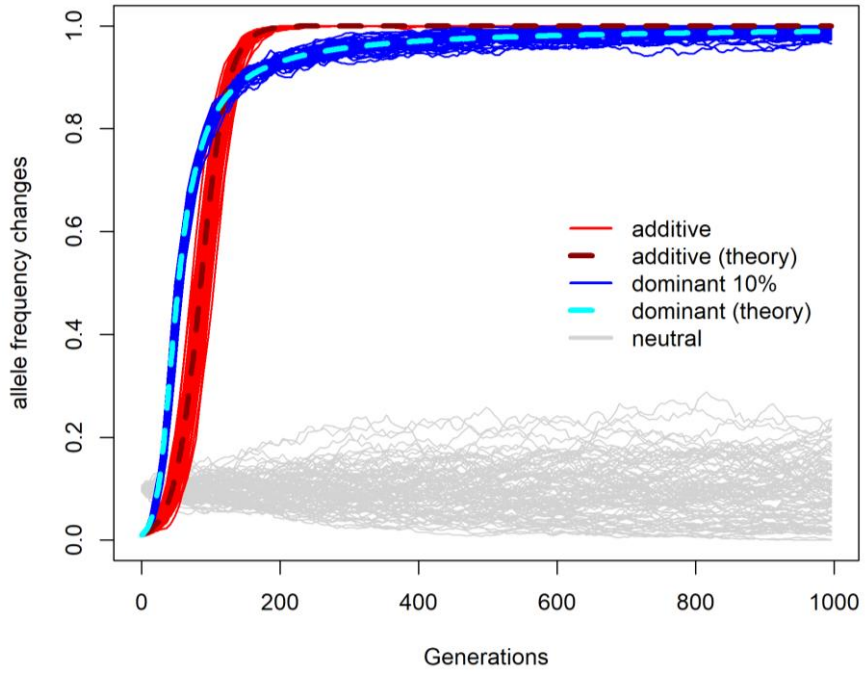
For **option A**, see in the **run-complete-loadfreq** folder for example, reproduced below:



Then for a better vizualisation of allele evolutionary trajectories across generations, at all loci under selection across both scenarios, and also at neutral loci, you need first to copy the R script **test5.1.plotFreq.r** in the working folder**,** then click on "**File/Custom plot**" from the GUI menu, and in the **Command line** window, type (or copy and paste) what is in bold below:

test5.1.plotFreq.r res1_freq_1.txt test5.1.p0.01 0.01 [*number of reproductive events*]

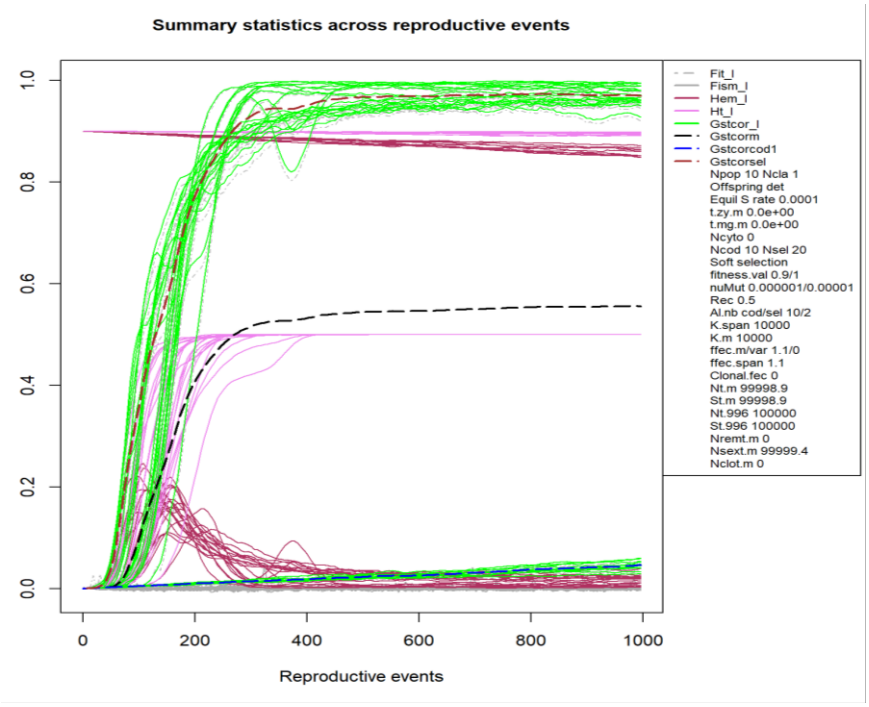
"0.01" is the initial frequency that was used in the loaded allele frequency file for **option A**, and more information is provided at the start of the script file. This gives:



Compare allele frequencies for neutral loci and loci under selection. Here, simulations are consistent with theoretical predictions, the different curves across replicates being very close to corresponding analytical models.

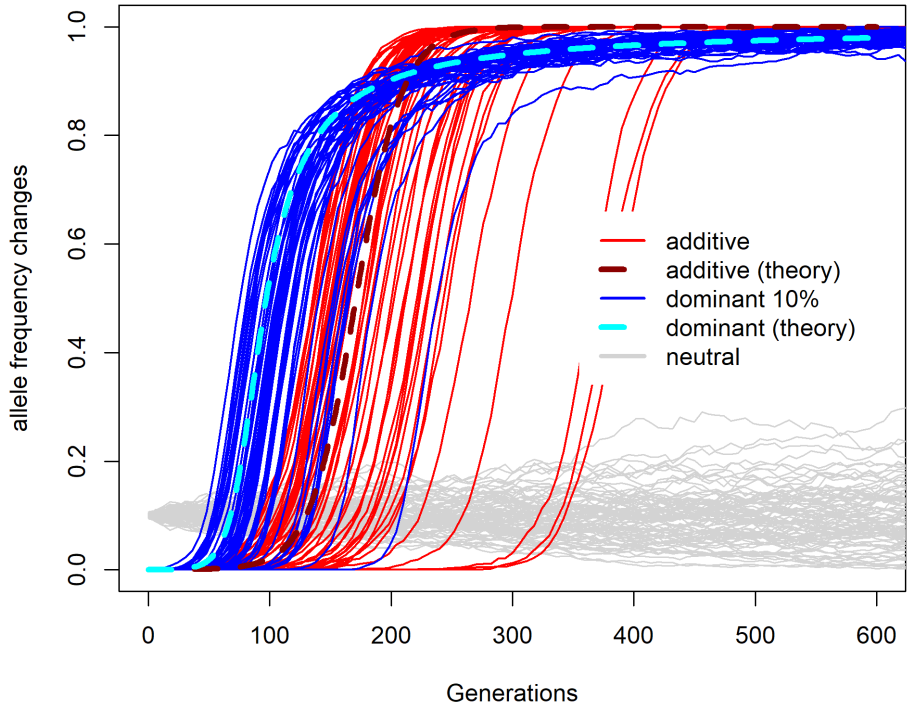
Adding a smaller reproductive event number such as “600” to the command line above allows zooming at the start of the simulations until that particular reproductive event. Look at the **test5.1.p0.01.600.png** plot obtained in this case (in folder **../5.5.1/run-complete-loadfreq).**

When looking at simulation results for **option B** (in the **.../5.5.1/run-complete-loadgeno** folder), the initial genotype file included only one or a few copies of the alleles under selection. Many trajectories are thus still close to theoretical curves but with more stochasticity. This can be seen with the default plot below:



Or using the “**Custom plot”** option with a frequency plot (drawn with a theoretical initial frequency of approximately p0=0.0001, zooming on the first 600 generations) and the command line:

test5.1.plotFreq.r res1_freq_1.txt test5.1.p0.0001.600 0.0001 600



In the plot **test5.1.p0.0001.600.png** above, in contrast to **option A**, alleles under positive selection can be lost more easily by drift in the first generations, but with a non-zero mutation rate, favourable alleles recurrently occur in the populations. Therefore observed frequency of selected alleles increase enough to be able to reach fixation, despite lagging behind predicted curves for both scenarios.

5.5.2 Recessive model

For the recessive model, the **test5.2-recessive-p0.01-run1.xml** file is provided with all the other files cited in the *.zip to download online**.** To recreate this file, follow the steps below:

Initial Settings: 5 populations, non-overlapping generations, selection on genotypes.

Genome Tab: 10 loci under selection (with no mutation here to be consistent with the theoretical model), 10 independent **Codominant 1** neutral loci.

For filling up the following Tabs, use the same choices as in the previous scenarios for preliminary runs, saving a genetic map from the *Genome Tab* (**test5.2-map.txt**), and asking for “0” generation and allele frequencies in the *Output Tab* (*param* file saved in **test5.2-prelim.xml**) Then edit the **res1_freq_1.txt** output file to start with the rare allele “2” for loci under selection at an initial frequency of 0.01 (following instructions for **option A** above in *5.5.1.1*). This file is renamed **test5.2-loadfreq-5pop.p0.01-run1.txt**.

Then all the steps are the same than for the complete run above for **option A**, using the four following files to load into the GUI:

test5.2-map.txt to load in the *Genome Tab,*

test5.2-loadfreq-5pop.p0.01-run1.txt (that corresponds to the map above) to load in the *Genotype Tab*,

test5.2-Wval-recessive-5pop.txt from the *Selection Tab,*

test5.2.K10000.5pop.csv in the *Demography Tab* (or the K values can be filled by hand)

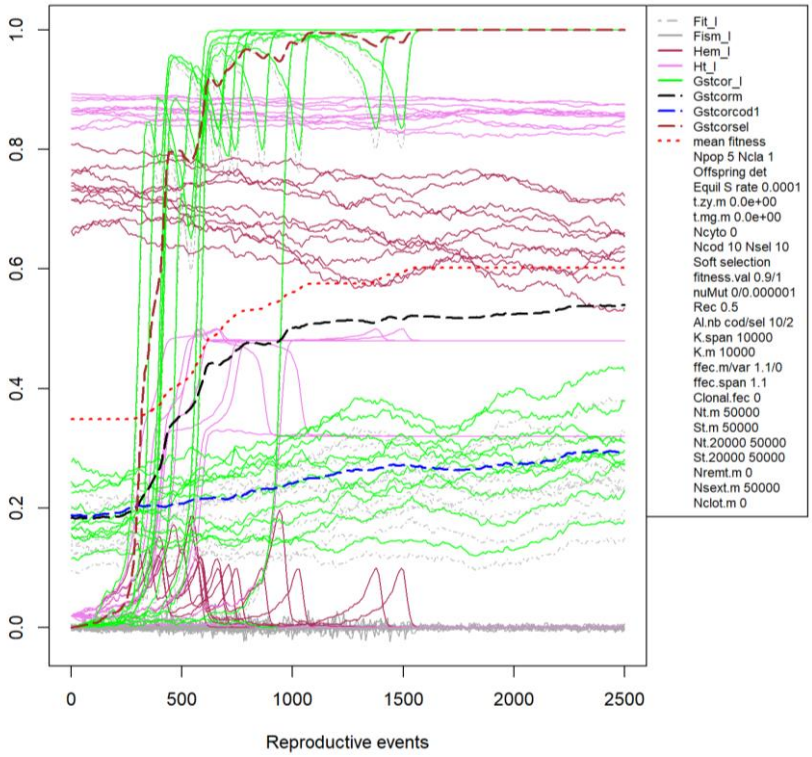
In the *Output Tab*, ask for "**20000**" reproductive events in total, with an event step of "**10**". Ask for genetic diversity statistics ("**GstCor per locus**", "**Mean GstCor across loci…"; "Fit, Ht, Fism, Hem per locus"; "Allele Frequencies"),** for all "**Demography**" statistics, and for **"Mean fitness across individuals…**". In the *Run Tab,* define the working folder **(…/5.5.2/run1-recessive**), save the *param* file, click on "**Create"** in both "**Type of result file**" and "**Prepare Configuration Files**" windows, and click "**Run"**.

Post-treatment

Once the run is completed, results can first be visualized with the "**Default plot**" in the working folder: in **plot.perGen.png,** loci under selection show a much more rapid increase in differentiation, among isolated populations, than neutral loci. The first 2500 generations can also be examined using the **plot.gM.perGen.zoom.R** script available from the exe folder (see more details on arguments in the script comments). Copy this script in the working folder and use the **File/Custom plot** option with the following command line:

plot.gM.perGen.zoom.R res1_per_gen_1.txt plot.gMperGen2500 1 2500

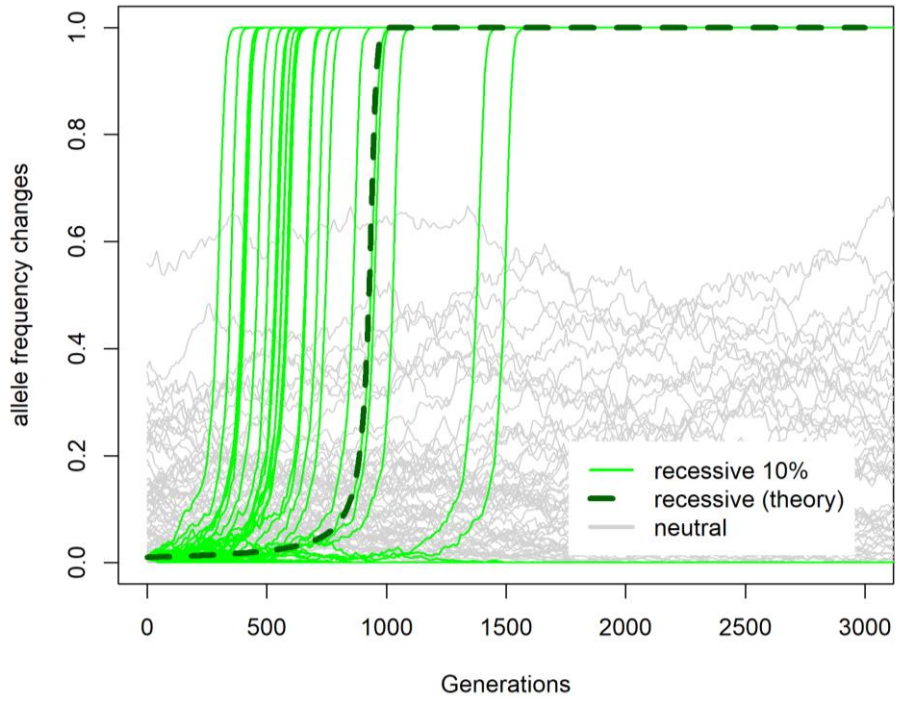
which gives the following plot:



As in the previous models described in **5.5.1** above, using the script **test5.2.plotFreq.R** allows looking directly at the evolution of allele frequencies. In the pop-up window from the **File/Custom plot** option, type the following command line

test5.2.plotFreq.r res1_freq_1.txt test5.2.p0.01.3000-nomut-run1 0.01 3000

which gives, in the working folder **.../5.5.2/run1-recessive**, the figure **test5.2.p0.01.3000-nomut-run1.png** below:



It can be observed that many simulated trajectories for selected alleles are close to the theoretical curve (dotted darkgreen line), but drift also affects the timing of increase in frequency strongly in some replicates (from 5 times less to higher than 28 times more than predicted in few other replicates), with some of the favourable recessive alleles being lost in some populations after a few hundreds generations.

Redoing another run of the same scenario with no mutation yields similar results (see in **test5.2-p0.01.3000-nomut-run2.png).**

Additional simulations for this recessive scenario with a lower initial frequency of 0.0005 for the selected allele, a larger population size ($K=10^5$) further illustrates the effect of drift on alleles under selection. In such a case, most alleles under selection are lost across the 50 replicates (i.e. 10 loci by 5 populations), and after 20000 generations, only 3 trajectories indicate that the favorable allele has been selected, while the inflexion point of the theoretical curve in a deterministic model is at around 18000 generations in this case (see the default plot **test5.2.plot.perGen.p0.0005-K10+05.png**, and the custom one **test5.2-p0.0005-K10+05….png**).

5.6 Tests 6: Weak selection on genotypes

Aim : Compare the theoretical fixation probability of a weakly selected allele for a range of N_e x s values (as in Fig. 5.14 p. 250, and equations 5.23 to 5.25 in Hartl and Clark (2007), Kimura 1957), with simulated results. In N_e x s, N_e is the effective population size and s the selection coefficient. So we simulate here 13 scenarios in 13 isolated populations for different conditions of selection, which vary from being weakly deleterious to weakly advantageous at one locus. For each scenario, 50 replicates of 1000 independent loci are simulated and the numbers of loci at which the mutant alleles have reached fixation are recorded. These are compared to the theoretical probabilities of ultimate fixation given by equation **5.23:**

$$u(p)=(1-4e^{-4N_e sp}) / (1-4e^{-4N_e s}),$$

N being the effective population size. The initial allele frequency of the advantageous allele is set at ***p*** = 0.03.

Building param files

Initial settings: 13 populations, selection on genotypes, non-overlapping generations. These are simulated in isolation to combine 13 different cases in one run, but each could be simulated separately.

5.6.1 Preliminary run (*saved in* ***test6.prelim.xml***):

As in **test 5.5.1**, the starting *ad hoc* conditions can be loaded either with an allele frequency file or with a genotype file. Since only one type of locus under selection is required here, the simplest way is to use an initial allele frequency file including 1000 loci where mutant alleles under selection are at frequency 0.03 (in equation 5.23 above), in order to avoid too much initial stochasticity.

NB.5.6-a: But see *5.5.1.1* in Test 5 for an example where a genotype file is loaded for starting conditions.

In order to get the correct format for this frequency file, do the following:

Genome Tab: 1000 locus under selection, 0 neutral codominant loci (so unclick the default 10 Codominant loci if creating a new *param* file).

Genotype Tab: default values for maximum and initial allele numbers in the "**Loci under selection**". Use "**Create"** in the **"Allele frequencies"** window**,** and "**All loci fixed …**"

Selection Tab: keep default values (no fitness value differences among genotypes, soft selection).

Demography Tab: use "**compute**" in the "**Initial number of individuals…"** then click on the "**Equivalent carrying capacity** …" and load the **test6.K100.13pop.csv** including 100 individuals in each population.

Output Tab: choose 0 "**Total number of reproductive events**", and "…**reproductive event #" "0" to "0" every" "1**" …in **Output # 1**. Unclick any default summary statistics and click **"Allele Frequencies".**

Run Tab: Define the working folder (**.../5.6/run-prelim**). Save the *param* xml file. Then click on both "**Create"** buttons in the "**Type of result file**" window and in the "**Prepare Configuration Files**" window with "**Simple**" (default), before launching the "**Run**". Then edit the file obtained (**res1_freq_1.txt)** by changing values of 1.000 into 0.97 and 0.00 into 0.03 (and rename it **test6.loadfreq.13pop.1000loc.p0.03.txt**), and use it in the complete run below.

5.6.2 Complete run (*saved in* ***test6.complete.xml***)

Genome and Genotype Tabs: as in the preliminary run, except that mutation rates must be set to zero.

In the **"Allele frequencies**" window: use "**Load frequencies**" and choose the file generated in the preliminary run above (**test6.loadfreq.13pop.1000loc.p0.03.txt**).

NB.5.6-b: Reminder: do not forget to rename the local paths in the *param* file provided **test6.complete.xml** if you want to use it, easily done by reloading the corresponding files (see **NB.5.1-c** and **NB.5.1-e** for more information).

Selection Tab : Use the default "**soft"** selection. Click on **Matrix(ces) of fitness values for user editing,** and load the file **test6-Wval-K100.txt** that contains the fitness values across genotypes for 13 populations in which the weak selection coefficients range from -4 to 2 (with a step of 0.5) if expressed as $4N_e s$, the range of coefficients ***s*** being set here for N_e=100.

The **recurrence rule** here insures that the same fitness values are being used for all 1000 loci. This can be checked by clicking OK for several following loci (or looking at the conf.txt once it has been created).

Demography Tab: as in **5.6.1** above.

Output Tab: ask for "**Allele Frequencies"** only, for 2000 reproductive events, with an event step of 2000 (more than what is needed for fixation in most cases).

Run Tab: Number of replicates : 50 (thus added to 1000 loci, this yields 50000 replicates in total for the same evolutionary conditions).

Runtime of ~ 2 min per replicate so less than 2 hours in total for 50 replicates on a Win64 OS.

Post-treatment

The script **test6.custom.r** can be launched directly from the GUI: copy the script into the working folder, then use option **File/Custom plot**, and type:

test6-custom.r res1_freq_1.txt test6-1000loc 50,

with the arguments being explained in the script comments. Then examine the plot **test6-1000loc.png,** which shows both the theoretical curve for equation **5.23** (in black and for a ***p*** of 0.03), and the percentage of loci (among 1000) having reached fixation for the mutant allele across the 50 replicates: black squares are the average across loci and replicates, and round dot size indicates an approximative weight for each probability value, both showing a good adequation with the theoretical curve. The red curve is for a newly arisen mutation (equation 5.24 in Hartl and Clarck 2007), which would have needed to many replicates to be observed with simulations easily. See also the default plot **plot.perGen.png** in the working folder, obtained by asking for more statistics in the *Output Tab***,** and the plots **test6-100loc.pdf** and **plot.perGen-100loc.pdf** for plots of previous runs with 100 loci and 10000 generations.

5.7 Tests 7: Selection on Phenotypes: comparison of trait variance estimates

Aims :

1) Compare theoretical estimates of phenotypic differentiation (and genetic differentiation at QTL) in a fully neutral model (no selection on the phenotypic trait)
2) Compare the evolutionary dynamics of scenarios where 1) panmixia can be assumed or is not a strong hypothesis, or 2) panmixia is not likely (such as with very strong selection or complex mating systems).
3) Compare diversity and differentiation statistics in those different scenarios.
4) Compare genetic variance estimates (VG: total genetic variance across genotypes, VA.p: additive genetic variance assuming panmixia, VA.gc: additive genetic variance with no particular assumption on mating, and associated heritability estimates (see **Table 4.4.2**).

The following scenarios are explored

Scenario 1: Neutral, so realized panmixia, one population or 2 isolated populations (initialized as a scenario with possible selection on phenotypes, but with negligible strength of selection values within populations). The prediction here is that

Scenario 2: medium strength selection within populations, strong selection among populations, no gene flow

Scenario 3: same as scenario 2 but with some gene flow among populations.

Scenario 4: with overlapping generations

5.8 Tests 8: Complex demography. Combination of clonal and mixed mating systems.

Aim: Test various features of the mating system module in the *Demography Tab,* in order to simulate a complex mating system combining selfing and clonal reproduction. This test was designed:

1. to validate the clonal reproduction rate ***c*** *a posteriori*, based on demographic data from simulation outputs and initial clonal and female fecundity values.
2. to compare the simulated results and F_{IS} computed statistics to the analytical formula of Balloux *et al*. (2003):

F_{IS} = (Nn(1-***c***)***s***-1)/(Nn(1-***c***)(2-***s***)+1), N being the population size, n the number of populations, ***c*** the clonal reproduction rate (defined as the clonal fecundity divided by the total fecundity (=female fecundity + clonal fecundity), and ***s*** the selfing rate. As explained before (see *3.5.5.7*), parents of clonal offsprings originate from the same population in gMetapop, unlike in simulations and theory developed in Balloux *et al.* (2003), so we expected to observe differences *a priori*. Clonal offspring are also identical to their parents (no mutation process during clonal reproduction).

Parameter files: based on Balloux et al. (2003), scenarios are explored for two constrasted clonal reproduction rates ***c*** of 0 or 0.95, combined with either a carrying capacity ***K*** value of 100 with a migration rate of 0.1, or a ***K*** value of 1000 and a migration rate of 0.01 (see details below). Thus 4 scenarios are illustrated altogether in 4 corresponding working folders (*param* files saved as **test8.K100.m0.1.c0.xml, test8.K100.m0.1.c0.95.xml, test8.K1000.m0.01.c0.xml, and test8.K1000.m0.01.c0.95.xml**.

Initial settings: 50 populations, non-overlapping generations, no selection.

Genome Tab: 20 independent loci, mutation rate 10^{-5}*.*

Genotype Tab: 99 alleles (maximum and initial numbers) per locus, we use an initial situation with maximum diversity (so click on option **"All loci with alleles in equal frequency"**).

Demography Tab: 1000 (default) or 100 individuals per population (load **test8.K100.50pop.txt**, K being the equilibrium *Ne* here). In the "**Mating system**" window, choose "**Sexual and clonal reproduction"**. The "**Selfing rate**" is 0 (default in the GUI, but this means that the actual realized selfing rate is 1/population size, thus either 0.01 or 0.001 as in Balloux *et al.* 2003, see **NB.5.8-a** below), and clonal reproduction rate values ***c*** are either 0 or 0.95.

NB.5.8-a: Please recall that the selfing rate in gMetapop corresponds to selfing events additional to those occurring in panmixia. Thus the actual selfing rate that needs to be chosen in the **GUI** is deduced from the total selfing rate needed in simulations less the one naturally occurring in panmixia by default for a given population size. In the case of scenarios 1 and 2, s=0.01, which is exactly 1/(population size) so no additional selfing events are needed during the simulations.

Defining clonal and female fecundities: "Clonal fecundity" values (which are different to clonal reproduction rates) are input values in gMetapop. Therefore, by assuming here a fixed total fecundity of 1.1 in each population, and based on the definition of ***c*** above, the clonal fecundity is either set to **0** in 2 among the 4 scenarios explored with a (default) female fecundity is value of **1.1** across populations (resp. **1.045** and **0.055** in the other 2 scenarios, loading **test8.FemFec.0.055.50pop.txt** for female fecundity values).

In the "**Migration models**" window, "**Zygote**" migration is set at a rate of 0.1 (for K=100, or 0.01 for K=1000) in an "**Island**" model. Male fecundities are default values.

NB.5.8-b: For the full range of scenarios needed to compare simulations to results in Balloux et al. (2003), clonal fecundity values were derived similarly from clonal reproduction rates varying between 0 and 1 (0, 0.8, 0.9, 0.95, 0.99, 0.999, 1), and selfing rate with three values: 0, 0.49 and 0.79 (to obtain realized selfing rates of 0.01, 0.5 and 0.8 in the simulations with 100 individuals per populations , so 19 simulated scenarios in total (see post-treatment below).

Output Tab: Ask for 10000 reproductive events (generations here) in total to reach equilibrium, and "**From reproductive event" "0" to "10000" every "10" "reproductive events"** in the "**Output #1**" window. Only values in the last generation are used for comparing with the theory and previous simulations, but more data help understanding the dynamics until an equilibrium is reached.

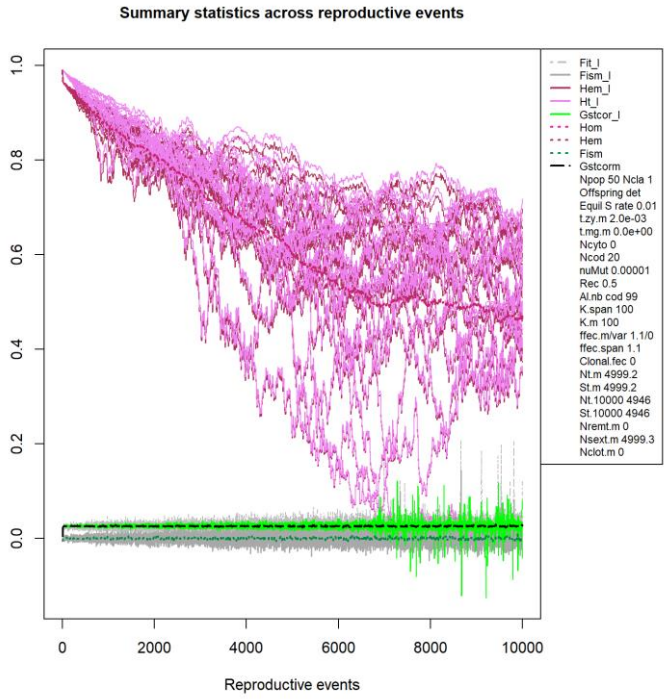
Ask for genetic diversity **Statistics per locus (Fis, Ho, He**), and **GstCor per locus**, and also the "**Mean statistics across loci (Fis, Ho, He)**" and "**Mean GstCor across loci…"**, and for all "**Demography**" output summary statistics.

Run Tab: Create the 4 working folders corresponding to the 4 explored scenarios (**.../5.8/run.K100.m0.1.c0**, **.../5.8/run.K100.m0.1.c0.95**, **.../5.8/run.K1000.m0.01.c0**, **.../5.8/run.K1000.m0.1.c0.95,** save each corresponding *param* file and ask for 1 replicate before launching the runs (20 replicates were used for each scenario and included in the plots mentionned below).

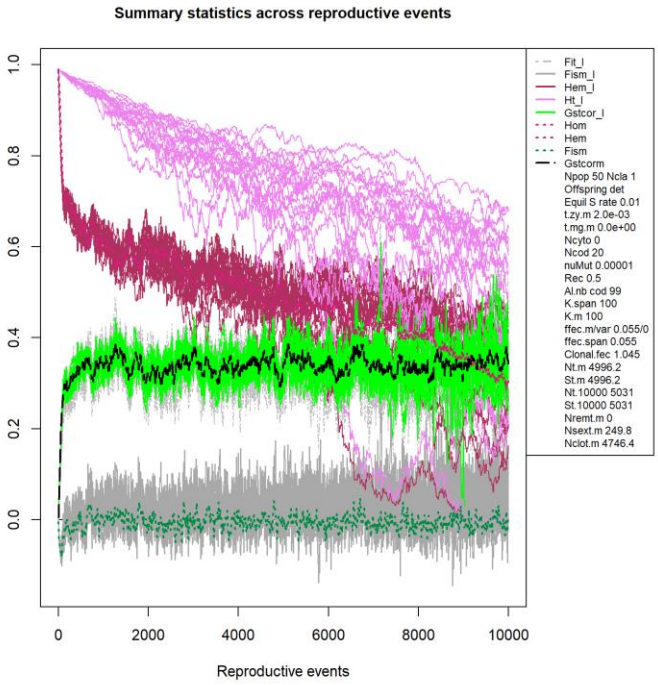
Runtime: less than a min for K=100 and ~3 min for K=1000 on OS Win64.

Post-treatment

a) For all scenarios, produce and examine the default plots **plot.perGen.png** (**File/Default plot** option in the GUI menu) in each subfolder of the **5.8.zip** that can be downloaded from the gMetapop website. Each plot correspond to simulation runs of one replicate after asking for 1 to 10000 generations, and summary statistics every 10 generations (different to the example above). Below for the scenario with ***c***=0 and and ***K***=100:



To compare with the scenario with ***c***=0.95 and ***K***=100:



b) We verified that the clonality rates estimated *a posteriori* from the simulated data using c= Nclot (= Nb of offspring from clonal reproduction)/(Nclot+ Nsext =Nb of offspring from sexual reproduction) were very close to expected values.

c) We computed both F_{ST} and F_{IS} mean values across loci for the 4 scenarios above and all other scenarios used in Balloux *et al*. 2003, across 20 replicates for each scenario. Both mean statistics and standard deviations were plotted as a function of initial ***c*** values and for 3 different values of selfing rates, comparing them to the theoretical curves derived in Balloux *et al*. 2003 (Figures **Fis=f(clonal-rate)….pdf** in the test8 folder).

The comparison between simulated scenarios reveals that mean F_{IS} values are close to theoretical values (or within their confidence intervals) except for the ***c*** value of 0.999, where clonality is the main way of producing offsprings. In this case, we indeed expected the strongest differences with the Balloux *et al*. theoretical case, especially for high migration rate (m=0.1), since the parents of clonal offspring cannot originate from migrants from other populations. With populations evolving in quasi-isolation and high clonal fecundity, large deficits of heterozygotes are expected (which is observed with F_{IS} values being negative and lower than in Balloux scenarios with a higher migration rate, see Figures above). Looking at F_{ST} values (highly correlated to GstCor statistics, see **Tests 3**), the increased isolation created by higher clonal rates leads to higher differentiation among populations than in Fig. 3 of Balloux *et al* (2003), with the greatest impact for ***c*** values being 0.99 and 0.999 (Fst > 0.75, see **Fst=f(clonal-rate)….pdf** Figures in the **…/5.8/** folder).

The fact that parents of clonal offspring are not migrants is a realistic biological assumption for natural populations of many plant species. Successful clonal individuals however will mate with others in subsequent generations and their lineage might migrate via offspring from sexual reproduction.

5.9 Tests 9: Complex demography with overlapping generations. Comparison with published results.

Aims: Revisit several simulations from Machon *et al*. 2003 that model a rare species with overlapping generations, user-defined demographic parameters, and very low number of individuals at the start of simulations. This allows testing the limits of the software in a demographic model with extreme parameter values. In addition, this test allowed validating code development when extending the class structure to be variable across populations, for age or development stages.

5.9.1 Preliminary runs *(saved in the param files provided online **test9-prelim-freq.xml** for **a)**, and **test9-prelim-geno.xml** for **b)** below)*

Here we illustrate alternative ways available in the GUI, for creating initial files and conditions to start simulations.

Building param files:

Initial settings: 10 populations, 30 classes, overlapping generations, neutral model, no selection.

Genome and Genotype Tabs: 1 biallelic locus, no mutation (such assumption is possible since the timescale of the simulations is very short) so put "**0**" in the **Mean** mutation window of the *Genome Tab*, and **"2"** and **"1"** in the "**Maximum…"** and "**Initial Number of alleles per locus"** in the "**Codominant 1"** window of the *Genotype Tab,* the **"1"** being automatic if you ask that all loci be fixed for one allele.

In the scenarios of Machon *et al*. (2003), the initial demographic conditions can easily be entered directly with the "**By hand"** option of the *Demography Tab* (see below). However the same conditions can also be created with:

a) either an initial allele frequency file indicating that the single individual in population 3 and the single one in population 7 are homozygous for alternative alleles (see the *Demography Tab* **3.5.1** options for more details). In this case, in the *Genotype Tab*, choose "**All loci fixed for one allele**",

b) or an initial file with individuals' genotypes (more on this below, in the *Run Tab*).

You can either load t*he param *.xml file* provided online, or re-create them by folloing the steps below:

Selection Tab: no selection

Demography Tab: two individuals initially in class 10, located in populations 3 and 7 respectively, each having a different homozygote genotype: this is done with the option "**by hand**" in the "**Initial number of individuals…"** window, adding "**1"** to **pop3** and **pop7**, each in **Class 10**. In this case, do not forget to unclick the first column for all populations, which indicates that populations will not be submitted to the search for an initial demographic equilibrium for the given ***K*** values. Alternatively here, you can also load a file previously created with the initial number of individuals in the different classes: click "**by hand**" then **"load file":** for example **test9-initial-number-indiv.csv.**

Default values for all other parameters.

NB.5.9-a: You also need to click on migration models and zygote, which have no impact here, to allow the conf.txt to be created in the *Run Tab* below, otherwise you get a generic ERRor preventing a simulation to start with isolated empty populations.

Output Tab: choose 0 "**Total number of demographic events**", and "**From demographic event #**" "**0"** to **"0"** every **"1**"… in the "**Output # 1"** window. Ask for "**Allele frequencies**" in case a), and no statistics are needed in case b).

Run Tab: define the working folder.

In case **a),** click on "**Select/Create**" for the "**run-prelim-freq"** working folder, and save the *param* file with all options above (**test9-prelim-freq.xml**), create both the type.txt and conf.txt files (click on "**Ignore**" when you get the warning "**Only one initial allele at codominant 1 loci and mutation rate is 0, ignore?"**)**,** and click **Run**.

In case **b),** define the "**run-prelim-geno"** working folder, and click on "**Final genotype filename**" in the "**Genotypes**" window, then save the *param* file (see **test9-prelim-geno.xml)**, create both the type and conf.txt files, ignore the **Warning** messages, and click **Run**.

When the simulations are finished (should be instantaneous here, unless genotype or allele frequency files are very large):

In case **a),** check the **res1_freq_1.txt** and edit it so that the second individual is fixed for the alternative allele, rename the file **test9_initial_freq.csv** (or .txt, see those provided) in the working folder. This file has the correct format for the complete run example below.

In case **b),** check the **save_ind1.txt**, which has been renamed **test9_initial_geno.txt** and has been edited to include two different homozygote individuals, and thus has the correct format for the genotype file to load into a complete run (see also the file **test9_initial_geno_format_notes**.txt explaining the file structure in this particular example).

5.9.2 Complete runs *(all param files cited below are provided online and can be loaded into the GUI directly to examine Tab values:* ***test9.pol1.det/.poi.xml, test9.pol3.det/.poi.xml,*** *and* ***test9.pol1.det.loadgeno.xml****)*

If you load the provided *.xml, do not forget to update paths of working folders and initial files to load into the GUI (see **NB.5.1-c** for more details).

At this stage, either keep the current *param* file, or "**File/Close…"** this file, and start a new one with "**File/New…"** for complete runs, re-using the same *Initial settings* and *Genome Tab* than for the above **5.9.1 Preliminary runs**.

Genotype Tab: in the "**Allele frequencies"** window,

a) either choose "**Load frequencies**" and "**load file**" using the **test9-initial-freq.txt** which has been previously created. Do not forget to update

NB.5.9-b: Please note here that although the initial number of alleles ("1") is correct, the only constraint is on the maximum number of alleles that will be checked when creating the conf.txt file, while loading the allele frequency file. Consistency will also be checked in the loaded frequency file with the Initial number of individuals.

b) or choose "**Load genotypes**" and load an initial genotype file (e.g. **test9-initial-geno.txt**).

NB.5.9-c: If the genotype file has been previously loaded in the *Genotype Tab* (see above) then the **"Initial number of individuals…"** window is not accessible.

Demography Tab: click on the "**Equivalent carrying capacity** …" to load the file including the equilibrium carrying capacities **test9.K100.10pop.csv** (K=100 individuals in each population).

In the "**Generating the number of offspring in each population and each class**" window, either choose the **"Deterministic"** option for offspring generation (corresponding to the **run-**

pol1-det working folder, defined in the *Run Tab* below), or a "**Poisson distribution" (**corresponding to the **run-pol1-poi** working folder**)**.

In the "**Mating system**" window, choose "**Sexual reproduction (only)",** and load the class Fecundity matrices for males and females: **test9-fem+male-fec.csv** in the "**… /** female and male fecundity windows (in this test, load twice the same file for each type of fecundity).

In the "**Extended Lefkovitch Matrix – Class parameters**" window, load **test9-equivalent_classSize.csv** in the "**Equivalent size**" window: (accounts for space occupied by juvenile individuals versus adult ones, here juveniles are considered equivalent to adults in terms of their class size properties, but see **Tests 10** below for different scenarios). Then load the values for the probabilities of remaining from one class to another in the **test9-remain**.csv file: here all values are "**0**" since no individual remain in the same class from one reproductive event to another. Load also the transition rates: **test9-transition**.csv: here all individuals change class at each reproductive event (in absence of density-dependence).

In the "**Migration models**" window, load the "**User-defined**" migration rate files:
one for the seed migration rates,
and one for male gamete migration rates.

In the "**Migration models/Zygote**" window, load and keep the **test9-seed.csv** after clicking on **"Matrix**", and after selecting the **"User-defined"** option for all scenarios. Five different male gamete migration rates are compared in Machon *et al.* 2003. Here, we illustrate two of these scenarios corresponding to different male migration rates, available either in **test9-pol1.csv** (within-population pollen rate=0, extreme case where no selfing is authorized, **pol1** scenario), or in **test9-pol3.csv** corresponding to higher rates of within-population pollen flow (**pol3** scenario). These are loaded in the "../**Male gamete**" window. Please note that since only two individuals are present at the start of simulations, the "**pol3**" scenario allows them the possibility of selfing. Click on "**Check matrix columns**" for a sum of "**1"**.

For each scenario, we also illustrate two different ways of generating offspring: either in a deterministic way with very little stochasticity or using a Poisson distribution, as described previously. In total, you can find five *param* ***.xml** files in the tutorial online: **test9.pol1.det/.poi.xml, test9.pol3.det/.poi.xml,** and the same **pol1** scenario but using the loading of initial conditions with genotypes (case **b**), saved in **test9.pol1.det.loadgeno.xml**.

Output Tab:

Ask for a total of 50 demographic events (correspond to iterations in both demographic and reproductive processes in the case of overlapping generations), with a number of classes used for computing summary statistics of 30 (all of them), with a "**Minimum population size**" of 1 (otherwise summary statistics won't be computed until the populations reach at least 10 individuals – default value-, impacting default plots). Put values "From… **"0"** to **"50"** every **"1"** "reproductive events" in the "**Output #1**" window. Ask for all genetic diversity statistics and all demography statistics.

Run Tab: "**Create/Select**" the working folders: **run-pol1/pol3-det/poi** for the case **a)** of loading frequency files, and **run-pol1-det-loadgeno** for case **b)**, and save the corresponding *.xml files. Then for both cases:

a) Click on "**Create"** in the "**Type of result file**" window, then "**Create**" in the "**Prepare Configuration Files**" window with "**Simple**" (default). Ignore any warning.
b) Perform the same clicks as above, the name of the genotype file that has been loaded should be seen in the "**Genotypes**" window, and its format is checked when creating the conf.txt file, so look for "**CHECK OK**" in the same window.

Ask for 10 replicates and click on "**Run**" in the "**Run gMetapop"** window (performed for **pol1** only in examples provided).

Post-treatment:

The figures obtained when using the **File/Defaul plot** option of the GUI menu can be compared across different scenario replicates (see in sub-folder "**all-plots…"** for a few replicates obtained for one particular scenario)**.** In order to use the default plot for a different replicates, use the "**File/Custom plot**" option of the Menu, and use the default **plot.gM.perGen.r** script in the command line window (see **3.9.2**) with any of the replicated res**2/3…/10**_per_gen_1.txt result files.

We can see that the evolution of the statistic F_{it} across generations is very similar to that in the corresponding scenarios in Figure 7 of Machon *et al*. (2003).This is despite additional regulation implemented in this program, which does not impact the results here (see *3.5.5.5*). *Runtime* is very rapid (~a few seconds) for each scenario, due to the small number of generations and individuals, and a neutral model with one locus, despite the large number of classes. You can also examine the evolution of demography in the **resX.tx**t files across scenarios.

5.10 Tests 10: Complex demography: comparison of neutral and phenotypic selection scenarios with different life-cycles

5.10.1 Different initial demographic conditions

A brief description of configurations are provided here for the 5 scenarios corresponding to the 5 general cases of initial demographic conditions described in *3.5.1.1* (see also **Figure 3.5.1.1** and **Figure 3**). The *param* xml files can be obtained from the website and imported into the GUI to examine parameter values:

Case 1 in **test10.1-case1.xml:** 4 populations, 5 classes, 50 neutral loci, 5000 reproductive events in total with a t (correspond to iterations in reproductive processes in the case of overlapping generations), with a number of classes used for computing summary statistics of 30 (all of them), with a "**Minimum population size**" of 1 (otherwise summary statistics won't be computed and thus plotted with the default plot until the populations reach at least 10 individuals). Put values "From… **"0"** to **"50"** every **"1"** "reproductive events" in the "**Output #1**" window. Ask for all genetic diversity statistics and all demography statistics.

Case 2 in test10.1-case2.xml:

Case 3 in test10.1-case3.xml:

Case 4 in test10.1-case4.xml:

Case 5 in test10.1-case5.xml:

5.10.2 Overlapping *versus* non-overlapping generations and variances in fecundities.

*Aims***:**

1) Compare the dynamics of genetic diversity and differenciation in overlapping *versus* non-overlapping generations, with or without selection on phenotypes.
2) Compare the dynamics of genetic diversity and population size in scenarios with contrasted ranges of fecundities across size classes;

Based on various theoretical studies dealing with overlapping generations, which are generally fairly complex given the wide range of possible models with an age class structure demography (e.g. Orive et al .revoir aussi ref in Waples et al 2014, Hellner), we can make several qualitative predictions.

For 1) a better maintenance of diversity in scenarios with overlapping generations whether the heterogeneity in fecundity is large or not (e.g. comparisons between scenarios 1 and 3, and between scenario 4 and 6)

For 2) a drop of *Ne* and and thus of genetic diversity when a higher heterogeneity in fecundities across age classes is simulated (e.g. comparison of scenarios 1 and 2 below, (see also **Tests 9**)

Scenario 1: neutral, 50 loci, overlapping generations, 4 isolated populations, 5 age/size classes, default homogenous increase of female and male fecundities across classes (= reference neutral scenario with overlapping generations).

Scenarios 2: same than scenario 1 with heterogenous female and male, and a Poisson distribution.

Scenario 3: same than scenario 1 with non-overlapping generations (so only 1 class per population)

Scenario 4: same than scenario 1 with medium strength selection on phenotypes within populations and a strong selection between populations, 50 QTL (= reference scenario with phenotypic selection, overlapping generations) , 50 codominant 1 neutral loci for comparison.

Scenarios 5: same than scenario 4 with heterogenous female and male fecundities

Scenario 6: same than scenario 4 non-overlapping generations

Comparisons: scenario 1 with 2 and 3, and scenario 4 with 5 and 6

5.10.3 Very large variance when generating offspring numbers

6. REFERENCES CITED

Abramowitz M, Stegun IA (1964) Handbook of mathematical functions with formulas, graphs, and mathematical tables U.S. Govt. Print. Off., Washington.

Albert AYK, *et al*. (2007) The genetics of adaptive shape shift in stickleback: Pleiotropy and effect size. *Evolution* **62**, 76–85

Andrello M, Manel S (2015). MetaPopGen: An R package to simulate population genetics in large size metapopulations. *Molecular Ecology Resources 15 : 1153-1162.*

Austerlitz F, Garnier-Géré PH (2003) Modelling the impact of colonisation on genetic diversity and differentiation of forest trees: interaction of life-cycle, pollen flow and seed long-distance dispersal. *Heredity* 90: 282–290.

Austerlitz F, Mariette S, Machon N, Gouyon P-H, Godelle B (2000) Effects of colonization processes on genetic diversity: differences between annual plants and tree species. *Genetics* 154: 1309-1321.

Balloux F, Lehmann L, de Meeûs T (2003) The population genetics of clonal and partially clonal diploids. *Genetics* 164: 1635-1644.

Barton NH, Keightley PD (2002) Understanding quantitative genetic variation. Nature Review Genetics 3: 11–21.

Briggs WH, Goldman IL (2006) Genetic variation and selection response in model breeding populations of Brassica rapa following a diversity bottleneck. *Genetics* 172: 457-465.

Caswell (2001) Matrix Population Models: Construction, Analysis, and Interpretation. Sinauer Associates.

Charlesworth B, Charlesworth D (2010) Elements of Evolutionary Genetics. Roberts and Company publishers, Greenwood Village, Colorado, US, 734 pp.

Christiansen FB (1975) Hard and soft selection in a subdivided population. The American Naturalist 109:11-16.

Crow JF, Kimura M (1970) An Introduction to Population Genetics Theory. Harper, Row editors, pp. 591. New-York, USA.

Day, C., Landguth, E. L., Bearlin, A., Holden, Z. A., & Whiteley, A. R. (2018). Using simulation modeling to inform management of invasive species: A case study of eastern brook trout suppression and eradication. *Biological Conservation*, *221*, 10–22. https ://doi.org/10.1016/j.biocon.2018.01.017

Ewens WJ 2004 Mathematical Population Genetics. I - Theoretical Introduction. Second Edition. Springer Science, New York.

Feng, C., Yi, H., Yang, L., Kang, M. 2020. The genetic basis of hybrid male sterility in sympatric Primulina species. BMC Evolutionary Biology, 20 (1),49–61.https://doi.org/10.1186/s12862-020-01617-4

Haller BC (2016) Eidos: A Simple Scripting Language. Available from: http://benhaller.com/slim/Eidos_Manual.pdf.

Fisher RA (1918) The correlation between relatives on the supposition of Mendelian inheritance. *Transactions of the Royal societyof Edinburgh.* 52: 399-433.

Haller BC, Messer PW (2019) SLiM 3: forward genetic simulations beyond the Wright–Fisher model. Molecular biology and evolution 36 (3), 632-637.

Hartl DL, Clark AG (2007) Principles of population genetics. 4th edition. Sinauer Associates. Sunderland, MA.

Johnson T, Barton N (2005) Theoretical models of selection and mutation on quantitative traits. *Philos Trans R Soc Lond B* Biol Sci 360: 1411–1425.

Kimura M (1957) Some problems of stochastic processes in genetics. *The Annals of Mathematical Statistics* 8: 882-901.

Kimura M, Weiss GH (1964) The stepping stone model of population structure and the decrease of genetic correlation with distance. *Genetics* 49: 561-576.

Kingman JFC (1978) A simple model for the balance between selection and mutation. *Journal of Applied Probability* 15: 1-12.

Kingsolver JG, Hoekstra HE, Hoekstra JM, Berrigan D, Vignieri SN, Hill CE, Hoang A, Gibert P, Beerli P (2001) The strength of phenotypic selection in natural populations. *American Naturalist* 157: 245–261.

Lande R (1975) The maintenance of genetic variability by mutation in a polygenic character with linked loci. *Genetical Research* 26: 221–235.

Lande R, Arnold SJ (1983) The measurement of selection on correlated characters. *Evolution* 37: 1210–1226.

Landguth EL, Cushman SA. (2010). CDPOP: A spatially-explicit cost distance population genetics program. *Molecular Ecology Resources*, 10: 156-161.

Landguth EL, Holden ZA, Mahalovich MF, Cushman SA (2017). Using landscape genetics simulations for planting blister rust resistance whitebark pine in the US Northern Rocky Mountains. *Frontiers in Genetics*, 8, 9.

Landguth E, Forester BR, Eckert A, Shirk AJ, Menon M, Whipple A, Day C, Cuchman SA (2019) Modelling multilocus selection in an individual-based, spatially-explicit landscape framework. *Molecular Ecology Resources* https://doi.org/10.1111/1755-0998.13121

L'Ecuyer P (1988) Efficient and portable combined random number generators. *Communication of the ACM* 31: 742-774.

L'Ecuyer P (2017) History of uniform random number generation. In *Proceedings of the 2017 Winter Simulation Conference*. Chan WKV, D'Ambrogio A, Zacharewicz G, Mustafee N, Wainer G, Page E (eds).
Le Corre V, Kremer A (2003) Comparative dynamics of genetic variability of an adaptive trait and its underlying genes in a subdivided population. *Genetics* 164: 1205-1219.
Le Corre V, Machon N, Petit RJ, Kremer A. (1997) Colonization with long-distance seed dispersal and genetic structure of maternally inherited genes in forest trees: a simulation study. *Genetical Research* 69, 117-125.
Lefkovitch, L.P. (1965) The study of population growth in organisms grouped by stages. *Biometrics* 21: 1-18.
Lowe WH, Kovach RP, Allendorf FW (2017) Population genetics and demography unite ecology and evolution. *Trends in Ecology & Evolution* 32: 141–152.
Lynch M, Walsh B (1998) Genetic analysis of quantitative traits. Sinauer Associates, Oxford University Press.
Machon N, Bardin P, Mazer SJ, Moret J, Godelle B, Austerlitz F (2003). Relationship between genetic structure and seed and pollen dispersal in the endangered orchid *Spiranthes spiralis*. *New Phytologist* 157: 677-687.
Matsumoto M, Nishimura T (1998) Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Transactions on Modeling and Computer Simulation* 8: 3-30.
Messer PW. (2013) SLiM: simulating evolution with selection and linkage. *Genetics* 194(4):1037–1039.
Neuenschwander S, Michaud F, Goudet J. (2018). QuantiNemo 2: A Swiss knife to simulate complex demographic and genetic scenarios, forward and backward in time. *Bioinformatics* 35: 886-888.
Nei M (1973) Analysis of gene diversity in subdivided populations. *Proceedings of the National Academy of Sciences of the USA* 70: 3321-3323.
Nei M (1987) Molecular Evolutionary Genetics. New York: Columbia University Press.
Raspail F, Austerlitz F, Mariette S, … Garnier-Géré P (2022) in preparation
Reznick D (2016) Hard and soft selection revisited: how evolution by natural selection works in the real world. *Journal of Heredity*. 107: 3–14
Roff DA (1997) Evolutionary Quantitative Genetics. Chapman & Hall, New-York, 493pp.
Rousset F (1997) Genetic differentiation and estimation of gene flow from F-statistics under isolation by distance. *Genetics* 145: 1219-1228.
Rousset F (2008) genepop'007: a complete re-implementation of the genepop software for Windows and Linux. *Molecular ecology resources* 8, 103–106.
Slatkin M (1991) Inbreeding coefficients and coalescence times. *Genetical Research* 58: 167-175.
Spitze K (1993) Population structure in Daphnia obtusa: Quantitative genetic and allozymic variation. *Genetics* 135: 367-374.
Takahata N (1983) Gene identity and genetic differentiation of populations in the finite island model. *Genetics* 104: 497-512.
Takahata N, Nei M (1984) F_{ST} and G_{ST} statistics in the finite island model. *Genetics* 107: 501-504.
Turelli M (1984) Heritable genetic variation via mutation selection balance: Lurch's zeta meets the abdominal bristle. *Theoretical Population Biology* 25: 138–193.
Wakeley J (2008) *Coalescent Theory: An Introduction.* Roberts & Company Publishers, Greenwood Village, Colorado, chapter 2.
Wallace B (1968) Polymorphism, population size, and genetic load. pp. 87-108 in R.C. Lewontin (ed). Population biology and evolution. Syracuse University Press, Syracuse, New York, USA.
Wallace B (1975). Hard and soft selection revisited. *Evolution* 29: 465-473.
Walsh B (2007) Evolutionary quantitative genetics, pp 533-586. In "Handbook of Statistical Genetics, third Edition", Balding DJ, Bishop M, Cannings C (eds), John Wiley & Sons, Chichester,UK.
Weir BS, Cockerham CC (1984) Estimating F-statistics for the analysis of population structure. *Evolution* 38: 1358-1370.
Walsh and Lynch 2018 book
Weiss GH, Kimura M (1965) A mathematical analysis of the stepping stone model of genetic correlation. *Journal of Applied Probability* 2: 129–149.
Whitlock MC (1999) Neutral additive genetic variance in a metapopulation. *Genetical Research* 74: 215–221. Maybe not, relire
Whitlock MC, Phillips PC, Moore FB-G, Tonsor SJ (1995) Multiple fitness peaks and epistasis. Annual Review of Ecology and Systematics 26: 601-629.
Wright S (1951) The genetical structure of populations. *Annals of Eugenics* 15: 323-354.
Wright S (1969) Evolution and the genetics of populations. Vol 2: The theory of gene frequencies. The University of Chicago press, Chicago.
Zeng ZB, Cockerham CC (1993) Mutation models and quantitative genetic variation. *Genetics* 133: 729-736.