q u a n t i N E M O

release 1.0.3

User Manual

September 5, 2009

authors
Samuel Neuenschwander
samuel.neuenschwander@unil.ch

Jérôme Goudet
jerome.goudet@unil.ch

Frédéric Hospital
frederic.hospital@jouy.inra.fr

Frédéric Guillaume
guillaum@zoology.ubc.ca

website
http://www.unil.ch/popgen/softwares/quantinemo

Contents

1 Introduction **1**

1.1 Scope . . . 1
1.2 Availability . . . 1
1.3 Technical . . . 1
1.4 License . . . 2
1.5 Acknowledgments . . . 2
1.6 Main features . . . 2
1.7 Input and output . . . 4

2 Using quantiNEMO **5**

2.1 Installation . . . 5
2.2 Launching quantiNEMO . . . 5
2.3 Input: the settings file . . . 6
2.3.1 Default value . . . 6
2.3.2 Comments . . . 6
2.3.3 Line end . . . 7
2.3.4 Macros . . . 7
2.3.5 Parameter types . . . 8
2.3.5.1 Integer . . . 8
2.3.5.2 Decimal . . . 8
2.3.5.3 String . . . 8
2.3.5.4 Matrix . . . 9
2.3.6 Temporal parameters . . . 10
2.3.7 External files . . . 11

2.4 Output files . . . 12
2.4.1 Types . . . 12
2.4.2 Naming convention . . . 12
2.5 Minimal settings file . . . 13
2.6 Simulation example . . . 14
2.7 Batch mode . . . 17
2.7.1 Multiple settings files . . . 17
2.7.2 Sequential parameters . . . 18

3 Life Cycle 20
3.1 Breeding . . . 21
3.2 Statistics . . . 25
3.3 Output . . . 27
3.4 Aging . . . 27
3.5 Regulation offspring . . . 28
3.6 Dispersal . . . 28
3.6.1 Density dependent dispersal rate . . . 31
3.7 Regulation adults . . . 33
3.8 Extinction . . . 33

4 Simulation 34

5 Metapopulation 38
5.1 Population sizes . . . 38
5.2 Selection pressure . . . 40
5.2.1 Stabilizing selection . . . 41
5.2.2 Directional selection . . . 42
5.2.3 Multiple traits with varying types of selection . . . 44
5.3 Summary statistics . . . 46

6 Quantitative traits 49
6.1 Architecture . . . 50
6.1.1 Allelic effects . . . 50

6.1.2 Dominance effects . . . 52
6.1.3 Epistatic effects . . . 54
6.1.4 Environment . . . 56
6.2 Mutation . . . 59
6.3 Multiple traits . . . 62
6.4 Selection models . . . 63
6.5 Genetic map . . . 64
6.6 Output . . . 65
6.6.1 Genotype . . . 66
6.6.2 Genotypic value . . . 68
6.6.3 Phenotypic value . . . 69
6.6.4 Architecture . . . 71
6.7 Summary statistics . . . 72

7 Neutral markers **75**
7.1 Architecture . . . 75
7.2 Mutation . . . 77
7.3 Multiple traits . . . 80
7.4 Genetic map . . . 81
7.5 Genotype output . . . 82
7.6 Summary statistics . . . 83

Bibliography **86**

Index **88**

Chapter 1

Introduction

1.1 Scope

quantiNEMO is an individual-based, genetically explicit stochastic simulation program. It was developed to investigate the effects of selection, mutation, recombination, and drift on quantitative traits with varying architectures in structured populations connected by migration and located in a heterogeneous habitat. quantiNEMO is highly flexible at various levels: population, selection, trait(s) architecture, genetic map for QTL and/or markers, environment, demography, mating system, etc.

1.2 Availability

The website http://www.unil.ch/popgen/softwares/quantinemo includes executables for Windows, Linux and Mac, the source code, a detailed user's manual, and also a syntax highlighting definition to edit the settings file with the shareware TextPad (http://www.textpad.com). All downloads are freely available under the terms of the GNU General Public License.

1.3 Technical

quantiNEMO is a console program, and is coded in standard C++ using an object oriented approach. This allows compiling quantiNEMO on any computer platform which supports standard C++ compilation. There is no limit on the number of populations, individuals, genes, etc that quantiNEMO can handle, apart from the available hardware capacities (CPU and memory). quantiNEMO was optimized for high computation efficiency in particular for large simulations on clusters. quantiNEMO

is built on the evolutionary and population genetics programming framework NEMO (Guillaume and Rougemont, 2006), with well developed demographic models. The demographic models of NEMO were kept, although several of these functionalities were re-coded, respectively adapted to the new functionalities of quantiNEMO.

1.4 License

1.5 Acknowledgments

We are grateful to Yves Rousselle, Patrick Meirmans, Christine Grossen, Claire Mouton, Olivier Blaser, Ilkka Kronholm, YiJian Huang, and Patrick Flight. These persons helped us to improve quantiNEMO by reporting bugs and/or great discussions.

1.6 Main features

quantiNEMO consists in several simulation components which may be easily extended. The simulation components with their corresponding parameters are described in more detail in the rest of this manual.

Quantitative traits
quantiNEMO allows the simulation of one to multiple quantitative traits each having its own specifications. Each trait is defined by one to many loci each with up to 256 alleles. The allelic effects at each locus can be drawn from a normal distribution or can be set explicitly. Mutations are implemented with several models. The trait determinism can be purely additive, or include dominance and/or epistatic interactions among loci. Environmental effects can also be set in different ways.

Neutral markers
quantiNEMO also allows the simulation of neutral markers, such as microsatellites

or SNPs with different mutation models (K Allele, Stepwise). Different types of neutral markers and/or quantitative trait loci can be combined within the same simulation.

Genetic map
quantiNEMO has an underlying genetic map, which may consist of several chromosomes. This allows an explicit positioning of all types of loci on the map (quantitative trait loci (QTL) and neutral markers).

Metapopulation
quantiNEMO allows simulating realistic population dynamics. Population sizes may vary in space and in time. The user can choose between several preset migration models (island, 1-D stepping-stone, 2-D stepping-stone), or specify the full migration matrix. The migration pattern can change over time, allowing to investigate scenarios of population fragmentation.

Lifecycle
Each individual undergoes a single life cycle (non-overlapping generations). The lifecycle is fixed (in contrast to NEMO (Guillaume and Rougemont, 2006)) and starts with breeding and reproduction. Several mating systems are available: random mating or selfing for hermaphrodites; promiscuity, monogamy or polygyny for doecious (gonochoric) species. Selection acts on the reproductive fitness of individuals. After reproduction, juveniles may disperse to other populations, and then population size is possibly regulated. Environmental stochasticity can also be introduced, where populations may go extinct due to an external factor, independent of population size or genetic constitution of the population.

Selection
Several modes of selection are available. In stabilizing selection modes, a specific optimum and selection intensity may be defined for each population and quantitative trait (Bürger, 2000). In directional selection modes, the strength and direction of selection may vary for each trait and population. Furthermore the selective pressures can change over time. Last, selection can be soft or hard (Wallace, 1968).

Initial settings
quantiNEMO is highly flexible in the setting of the initial simulation conditions. The initial population sizes may be set for each population and sex (parameters patch_ini_size, patch_ini_size_fem, or patch_ini_size_mal). Furthermore, the initial allele frequencies for each population may be either maximal polymorph or monomorph (parameters quanti_ini_allele_model and ntrl_ini_allele_model), or may be set explicitly for each population, locus, and allele (parameters quanti_allelic_file and ntrl_allelic_file).

1.7 Input and output

Input
quantiNEMO is launched using a settings file. The settings file is a text file with flexible and easy to understand structure. The information is specified in a parameter-argument scheme, where the order of the parameters does not matter. The file can be edited with any text editor, and comments may be added for better readability. We provide also a syntax-highlighting definition for better readability.

Output
quantiNEMO provides summary statistics for the different simulation components, including genetic variance estimates, quantitative trait analysis (e.g. Q_{ST}), and F-statistics for all types of loci (neutral and QTL). Most of the summary statistics are available for juveniles and adults. The summary statistics can be computed for any generation during the simulation. quantiNEMO can also produce files with the raw genetic data. The genotypes at all loci can be dumped to file in the FSTAT format (Goudet, 1995). Phenotypes, as well as the additive, dominance, and epistatic effect values can be written to a file and then analyzed with any population or quantitative genetic software, e.g. to get patterns of differentiation, study linkage disequilibrium, or scan for QTLs. We are currently developing an R package to carry out the most common analyses from a simulation.

Chapter 2

Using quantiNEMO

This chapter explains how to use quantiNEMO starting from the basics.

2.1 Installation

Executables of quantiNEMO for several operating systems can be downloaded from the web site http://www.unil.ch/popgen/softwares/quantinemo. The executables are stand-alones, meaning that quantiNEMO does not require an installation. After downloading the compressed file with the executable corresponding to your operating system simply extract it to a folder of your choice. The compressed file includes the executable for your operating system, the user manual, and also an example settings file.

2.2 Launching quantiNEMO

A settings file allows to define any simulation. The format of this settings file is described in section 2.3. There are several possibilities to pass the settings file to the executable:

standard. A simulation is launched by double-clicking on the executable or by writing the name of the executable (e.g. `quantinemo.exe` for Windows) to a console window. quantiNEMO will prompt you for the name of the settings file.

default file If the settings file has the default name `quantinemo.ini` and the settings file resides in the folder of the executable quantiNEMO will automatically use this settings file to perform the simulation when quantiNEMO is launched.

parameter. The settings file name may be passed as a parameter to the executable when quantiNEMO is launched. This can be done using a console window:

```
> quantinemo.exe settings.ini
```

Depending on the operating system a '`./`' in front of the executable is sometimes needed to specify that the executable is located in the current directory (Linux):

```
> ./quantinemo settings.ini
```

2.3 Input: the settings file

This section describes the format of the settings file. The settings file is a text file with in general one parameter per line in a key-value scheme. For example

```
patch_capacity 1000
```

sets the parameter patch_capacity to the value of 1000. The order of appearance of the parameters In the settings file does not matter. However, a particular parameter should appear only once in the settings file. If a parameter appears several times in the file only the last instance is considered.

2.3.1 Default value

Most of the parameters have default values. The default value of a parameter is taken into account when either the parameter is not listed in the settings file or its argument is missing. The default values are listed behind the parameter name in this manual and are specified by (default:). The default values are the most common setting of the parameter. the default values allow to keep the settings file short and clear.

2.3.2 Comments

quantiNEMO allows to add comments in the settings file (and all other input files). There are two different types of comments:

Single line comments. A simple hash character '#' defines the start of a single line comment. The hash character and the remaining text of the line are ignored by quantiNEMO.

Block comments. Block comments may start and end at any place in the file. This allows to comment out multiple lines at once or only a part of a line. A block comment starts with the characters '#/' and ends with the characters '/#'. The starting and ending characters and the text between them are ignored.

2.3.3 Line end

In general a parameter (the key and its argument) has to be written on a single line (except for matrices and temporal parameters). However, using a backslash '\' it is possible to bypass the end of a line and to write an argument on several lines. Note, that after the backslash any text on the line is removed.

2.3.4 Macros

quantiNEMO has two input macros. The macros help to keep the input file clear and small. In principle the macros just write out the sequence or the repetition, before the input file is read by quantiNEMO.

Sequence
With the command *seq(from, to, steps)* it is possible to specify a sequence of data points. *seq* works similar to *seq* in the statistical package R. The macro needs three arguments separated by a comma: the first one is the first data point of the sequence, the second argument is the last data point of the sequence and the third argument specifies the number of data points including the edges. Example:

```
patch_capacity {seq(100, 1000, 10)}
patch_capacity {100 200 300 400 500 600 700 800 900 1000}
```

Both specifications of the carrying capacities are identical, i.e. the macro *seq* translates its arguments to the lower specification.
The command *seq* can also be used for temporal parameters with a single number as argument:

```
patch_capacity (seq(0 100,90 1000, 10))
patch_capacity (0 100,
                10 200,
                20 300,
                30 400,
                40 500,
                50 600,
                60 700,
                70 800,
                80 900,
                90 1000)
```

Both specifications of the linear increase of the carrying capacities over time are identical. Again the macro *seq* translates its arguments into the lower specification.

Repetition
With the command *rep(value, number)* it is possible to specify a repetition of a single data point. *rep* works similar to *rep* in the statistical package R. The macro needs two arguments separated by a comma: the first one is the data point to be repeated, and the second argument specifies the number of repetitions. Example:

```
patch_ini_size {1000 rep(0, 9)}
patch_ini_size {1000 0 0 0 0 0 0 0 0 0}
```

Both specifications of the initial population sizes are identical, i.e. the macro *rep* translates its arguments to the lower specification. In this example only the first patch is populated at the start of the simulation, allowing to simulate a colonization scenario.

2.3.5 Parameter types

There are different types of arguments that a parameter may take. In the following the argument type is specified within square brackets. Example:

```
stat_log_time [integer]
```

2.3.5.1 Integer

Integers are whole-numbers, i.e. a dot-less number. The following forms are equivalent: 1000 or 1e3.

2.3.5.2 Decimal

Decimals are floating-point numbers. The following forms are equivalent: 0.0001, .0001 or 1e-4.

2.3.5.3 String

Strings are text arguments. If the string contains spaces the argument has to be enclosed within quotation marks "...". When a string is enclosed by quotation marks it may be written over several lines. Example of a string with a space:

```
folder "first simulation"
```

2.3.5.4 Matrix

Matrices allow to pass several numbers (integer or decimal) to a parameter. This may be necessary to specify carrying capacities (see section 1D Matrix), or to pass a dispersal matrix (see section 2D Matrix) to quantiNEMO. Matrices are enclosed between curly brackets '{ }', and numbers are separated by at least a space. Matrices may be written on several lines and may also contain comments. There is no *a priori* restriction on the size of the matrix.

1D Matrix
A one dimensional matrix (vector) consists of data in a single dimension. There are three different ways to write a one dimensional matrix, which are all equivalent:

```
patch_number 4
patch_capacity {  20   10   20   10  }
patch_capacity { {20   10   20   10} }
patch_capacity { {20} {10} {20} {10} }
```

2D Matrix
Some parameters need a second dimension for their argument. A second dimension is obtained by enclosing the inner rows of the matrix again within curly brackets '{ }'.

```
patch_number 4
disp_rate { {0.2  0.0  0.0  0.8}
            {0.4  0.2  0.0  0.4}
            {0.4  0.4  0.2  0.0}
            {0.0  0.4  0.4  0.2} }
```

This example shows the pairwise dispersal matrix for 4 patches (4x4). Each row specifies a source patch from which emigrants emigrate. Each column specifies the target patch receiving the immigrants. The diagonal of the matrix specifies the proportion of individuals remaining in the natal patch.

Matrix length adjustment
Usually matrices are defined in whole, i.e. the number of carrying capacities in the 1D matrix example above meets the number of patches (4 patches). However, if there is a repeating pattern in the matrix, it is also possible to define only the repetition. In this case the matrix will be repeated as needed. For instance the 1D matrix above could also be written in one of the following ways as there is a repetition in it:

```
patch_number 4
patch_capacity {20  10}
```

Note, that rows and/or columns are repeated as needed. If the number of columns (or rows) is not an entire subset a warning will be returned and the simulation will adjust the matrix as:

```
patch_number 5
patch_capacity {20  10}
```

In this example we have 5 patches, however the carrying capacities are only specified for 2 patches. As 2 is not an entire subset of 5 a warning will be returned and the patches will have the following carrying capacities: 20, 10, 20, 10, 20. If all values of a matrix are identical one may leave out the brackets. In the following example all three declarations result in the same simulation:

```
patch_number 4
patch_capacity {20  20  20  20}
patch_capacity {20}
patch_capacity  20
```

Unbalanced matrices
Usually a matrix is balanced, i.e. each row has the same number of columns. However, some parameters (such as the parameter `quanti_loci_positions`) allow to have unbalanced matrices, i.e. rows do not necessarily contain the same number of columns. Other parameters (such as the parameter `quanti_loci_positions`) allow to skip a row, i.e. some rows do not contain any data. A row can be skipped by explicitly indicating the rank of the row just after the beginning of the row followed by a colon ":". The ranking starts with 1. If a row has no explicit rank it is assumed to follow the preceding row. Here is an example for the genetic map of a quantitative trait:

```
quanti_loci 11
quanti_loci_positions { {1: 10}
                        {3: 10  40  60  80  100}
                        {   10  40  60  80  100} }
```

In this example the quantitative trait is defined by 11 loci located on three out of four chromosomes. The first chromosome contains a single locus at position 10 cM. No locus is located on the second chromosome. The third and fourth chromosome have the same structure: 5 loci are located on each of the two chromosomes, and the distance between adjacent loci is 20 cM.

2.3.6 Temporal parameters

An important feature of quantiNEMO is that some parameters may change over time during a simulation. Such parameters are indicated as "temporal" in this manual. Temporal arguments are enclosed within two parentheses '(...)'. For each change of

the argument over time a pair consisting of a generation index and a corresponding argument is needed. The values of a pair are separated by at least one space. Pairs are separated by a coma ',' or by a semi-colon ';'. The first value of a pair specifies the time, i.e. before which generation the change happens. The second value is the new argument. A parameter may change as often as any generation. A simulation starts at generation 1. Therefore the first change has to have the time value 1 otherwise the parameter cannot be initialized leading quantiNEMO to return an error. A temporal argument may be written over several lines.

```
patch_capacity (1   100,
                50  200,
                100 500)
```

Here, the carrying capacity is 100 for the first 49 generations, 200 from generation 50 on, and 500 from generation 100 on.

2.3.7 External files

In general arguments are written directly after the parameter name on a single line. However, arguments may be sometimes large (e.g. big matrices, temporal parameters, ...) leading to poorly readable settings file. Using external files for large arguments allows to keep the settings file clear and well readable. An external file may be used for any parameter. Instead of writing the argument directly after the parameter name, the name of an external file is written after the parameter name. In order for quantiNEMO to recognize the argument as a file name the prefix '$' has to be added before the file name. The external file must contain the argument of the parameter. Only a single argument per external file is possible, however the same external file may be used for several parameters. The format of the argument in the external file follows the same rules as in the settings file, except that line breaks are ignored, i.e the character '\' between lines is not necessary. Here is an example:

settings file:

```
disp_rate $dispersal_file.txt
```

external file named "dispersal_file.txt":

```
# dispersal rates
{ {0.2  0.0  0.0  0.4  0.4}
  {0.4  0.2  0.0  0.0  0.4}
  {0.4  0.4  0.2  0.0  0.0}
  {0.0  0.4  0.4  0.2  0.0}
  {0.0  0.0  0.4  0.4  0.2} }
```

2.4 Output files

2.4.1 Types of output

quantiNEMO can generate the following types of outputs:

summary statistics. quantiNEMO provides summary statistics for the different simulation components, including genetic variance estimates, quantitative trait analysis (e.g. h^2, V_A, genetic diversity), and F-statistics for all types of loci (neutral and QTL). Most of the summary statistics are available for juveniles and adults. The summary statistics can be computed for any generation during the simulation. The summary statistics can be saved in two ways: Either they are stored for each replicate separately and/or the summary statistics are averaged across replicates.

raw data. quantiNEMO can also produce files with the raw genetic and phenotypic data. The genotypes at all loci can be dumped to file in the FSTAT format (Goudet, 1995). Phenotypes, as well as the additive, dominance, and epistatic effect values can be written to a file and then analyzed with any population or quantitative genetic software, e.g. to get patterns of differentiation, study linkage disequilibrium, or scan for QTL. Genotypes and phenotypes can be saved for any generation during the simulation.

log files. quantiNEMO also generates log files allowing to reconstruct performed simulations. There are two types of log files. The first log file records the simulations performed with quantiNEMO and stores some general information. This log file is stored in the folder of the executable and allows to reconstruct the chronology of performed simulations and their main features. The other log file contains the used parameters, is generated for each simulation separately, and is stored in the simulation folder. This file is in principle a copy of the used settings file and contains the starting time and the duration time of the simulation. It contains also the seed (see parameter seed) used to initialize the simulation. This file can be used as settings file to exactly repeat the performed simulation. Note, that due to the seed in the file the random generator will be initialized in the same way leading to the exact same values in the output.

2.4.2 Naming convention

All files of a simulation are stored in a unique folder (see parameter folder). This simulation folder may contain a substructure. The name of the output files are based on the base name given by the parameter filename. Depending on the type of output different extensions are added to the base name. To avoid that recurring outputs overwrite previous outputs a counter is added to the file name between base

name and extension. There are two types of counters: the generation counter and the replication counter. A counter is only added if there is a risk of overwriting. For example the replication counter is only added if several replications are performed. The generation counter starts with "_g" and the replication counter with "_r". These characters are followed by the number of the generation and replication, respectively. Note, that generations and replications start at 1. The number has as many digits as are needed to represent the highest number in the simulation:

```
simulation_g0001_r01.dat
simulation_g0002_r01.dat
...
simulation_g5000_r10.dat
```

2.5 Minimal settings file

Most of the parameters have default values. This allows to make short and clear settings files. This section describes the parameters needed for a minimal settings file and describes the simulated model, respectively how the default values are set.

The following two parameters are needed in every settings file:

```
generations      500
patch_capacity   1000
```

This minimal settings file allows to perform a simulation with a single population consisting of 1000 hermaphrodites. The population evolves under neutral random mating for 500 generations keeping the population size constant at carrying capacity. No genetic data are simulated, since no quantitative traits or neutral markers are specified. The simulation generates no output apart from the log file. Although this simulation works it makes no sense, since no output is generated. This can be improved by specifying quantiNEMO to compute summary statistics on the demography:

```
generations      500
patch_capacity   1000
stat             {adlt.demo}
```

This settings file results in exactly the same simulation as the previous one, but now three additional files are generated. The first file lists the names of the computed summary statistics ("simulation_legend.txt"), while the two other files are almost identical containing the summary statistics for each generation. One of these latter files contains the summary statistics for each generation and replicate separately ("simulation.txt"), while the other one shows the summary statistics averaged across replicates ("simulation_mean.txt").

To simulate genetic data one has to add either a quantitative trait or a neutral marker to the simulation. This is done by specifying the number of loci to simulate (for quantitative traits and neutral markers separately). To force quantiNEMO to compute some summary statistics on quantitative traits, respectively on neutral markers one has also to specify corresponding summary statistics. For example:

```
generations       500
patch_capacity    1000
stat              {adlt.demo
                   quanti
                   n.adlt.fstat}
quanti_loci       1
ntrl_loci         1
```

In this example now selection acts on the reproduction stage (soft selection). The fitness is computed on the simulated quantitative trait. The trait consists of a single locus with up to 255 alleles. Allelic effects are normally distributed with a variance of 1. The phenotype of the quantitative trait is determined by pure additive effects of the alleles. Stabilizing selection acts on the phenotype with an optimum of 0 and a variance of 1. A single neutral marker locus is simulated with up to 2555 alleles. Both the quantitative trait locus and the neutral marker locus do not mutate. for both markers some common summary statistics are computed.

2.6 Simulation example

This section describes a more realistic simulation example and describes how the output is stored. The settings file of this example named "quantiNemo_example.ini" is included in the compressed folders of the downloads of quantiNEMO:

```
generations                 100

# metapopulation
patch_number                10
patch_capacity              1000
dispersal_rate              0.01

# mating
breed_model                 0
mating_system               0

# selection
patch_stab_sel_optima       10
patch_stab_sel_intensity    1

# quantitative trait
quanti_loci                 5
quanti_all                  255
```

```
quanti_mutation_model       0
quanti_mutation_rate        1e-4
quanti_save_genotype        2
quanti_genot_logtime        10
quanti_genot_dir            quanti_genotype
quanti_save_phenotype       2
quanti_phenot_logtime       10
quanti_phenot_dir           quanti_phenotype

# neutral marker
ntrl_loci                   5
ntrl_all                    10
ntrl_mutation_model         0
ntrl_mutation_rate          1e-4

ntrl_save_genotype          2
ntrl_genot_logtime          10
ntrl_genot_dir              ntrl_genotype

# statistics
stat                        {n.adlt.fstat}
stat_save                   1
stat_log_time               10
stat_dir                    stats
```

A simulation based on this settings file named "quantiNemo_example.ini" produces the following output to your terminal window:

```
************************************************

                q u a n t i N E M O
************************************************
*      Release: 1.0.0.0 [Jan 01 2008; 00:00:00]     *
*      Copyright (C) 2008 Samuel Neuenschwander     *
* http://www2.unil.ch/popgen/softwares/quantinemo *
************************************************

Reading settings file 'quantiNemo_example.ini' ...done!

SETTINGS
  Simulation:
    100 generations
    1 replicates

  Loaded traits:
    Neutral marker type: 5 loci; 10 alleles
    Quantitative trait: 5 loci; 255 alleles; stabilizing selection

  Life cycle sequence:
    1. breed
    2. save_stats
    3. save_files
```

```
      4. aging
      5. disperse
      6. regulation

    Metapopulation:
      10 populations
      Migration model: island
      Mating system: random mating (hermaphrodite)


SIMULATION
      replicate 1/1 [00:00:14] 100/100

------ SIMULATION done (CPU time: 00:00:15s) ------

quantiNEMO terminated successfully!
```

This output informs you that this simulation was parameterized by the settings file "quantiNemo_example.ini". The simulation consisted of one quantitative trait and one type of neutral markers. The simulated life cycle is indicated and shows that summary statistics and genotypes or phenotypes are output. Only one replicate of 100 generations was performed. For each replicate the elapsed time (hh:mm:ss) is printed out to the console and at the end of the simulations also the total elapsed time. The output of this simulation was stored in the following structure relative to the executable:

```
simulation_2008-01-01_00-00-00/     # automatically named folder
    simulation.log                  # log file
    quanti_phenotype/               # phenotype folder
        simulation_g010.phe         # phenotypes of generation 10
        simulation_g020.phe
        simulation_g030.phe
        simulation_g040.phe
        simulation_g050.phe
        simulation_g060.phe
        simulation_g070.phe
        simulation_g080.phe
        simulation_g090.phe
        simulation_g100.phe
    quanti_genotype/                # genotype folder (quantitative)
        simulation_g010.dat         # genotypes at generation 10
        simulation_g020.dat
        simulation_g030.dat
        simulation_g040.dat
        simulation_g050.dat
        simulation_g060.dat
        simulation_g070.dat
        simulation_g080.dat
        simulation_g090.dat
        simulation_g100.dat
```

```
    ntrl_genotype/                  # genotype folder (neutral)
        simulation_g010.dat         # genotype at generation 10
        simulation_g020.dat
        simulation_g030.dat
        simulation_g040.dat
        simulation_g050.dat
        simulation_g060.dat
        simulation_g070.dat
        simulation_g080.dat
        simulation_g090.dat
        simulation_g100.dat
    stats/                          # statistic files
        simulation.txt              # statistics for each replicate
        simulation_mean.txt         # statistics across replicates
        simulation_legend.txt       # statistic legends
```

2.7 Batch mode

quantiNEMO allows to perform multiple simulations by executing a single command. There are two ways to perform such batch simulations. Note, that these two types may not be mixed, i.e. be used at the same time.

2.7.1 Multiple settings files

A normal simulation can be launched by passing the settings file name as parameter to the executable. It is also possible to pass several settings file names to the executable. In this case a simulation for each settings file is executed consecutively:

```
> quantinemo.exe sim1.ini sim2.ini
```

In this example quantiNEMO is launched with two settings files (`sim1.ini` and `sim2.ini`). The simulations will be executed one after the other leading to the following console output:

```
...
Reading settings file 'sim1.ini' ...done!

Reading settings file 'sim2.ini' ...done!


------ SIMULATION 1/2 ------

...

------ SIMULATION 1/2 done (CPU time: 00:00:15s) ------
```

```
------ SIMULATION 2/2 ------

...

------ SIMULATION 2/2 done (CPU time: 00:00:16s) ------
```

2.7.2 Sequential parameters

A batch simulation may also be launched by a single settings file if so-called sequential parameters are used. Sequential parameters are any parameter with not only one but several arguments. Note, that a sequential parameter is not the same as a temporal parameter (see section 2.3.6).

```
patch_capacity 5 10 20
```

In this example patch_capacity is a sequential parameter with three arguments. If patch_capacity is the only sequential parameter three consecutive simulations will be launched with identical parameter arguments, except for the parameter patch_capacity which will be set to 5 for the first simulation, to 10 for the second simulation, and to 20 for the third simulation.

If several parameters are sequential parameters all combinations of the sequential parameters will be simulated. Example:

```
patch_number   10 50
patch_capacity 5 10 20
```

There are two sequential parameters in this example. This will result in 6 consecutive simulations with the following parameters:

	patch_number	patch_capacity
1. simulation	10	5
2. simulation	10	10
3. simulation	10	20
4. simulation	50	5
5. simulation	50	10
6. simulation	50	20

To prevent the output from overwriting the preceding simulation a unique base file name is given to each simulation. This unique base file name consists of the the base file name (parameter filename) plus a suffix which includes the rank of the simulation. If in the example above the parameter filename was set to "mysim" the base name for each simulation would be as follows:

	filename
1. simulation	mysim−1
2. simulation	mysim−2
3. simulation	mysim−3
4. simulation	mysim−4
5. simulation	mysim−5
6. simulation	mysim−6

However, the base file name can also be individualized by the user using expansion characters '%' in the base file name. The expansion characters allow to incorporate the changing argument value (of the sequential parameters) in the base file name. For this the expansion character '%' has to be followed by a number which corresponds to the rank of the sequential parameter. Note, that the rank corresponds to the alphabetical order of the sequential parameter names, starting with 1 for the first sequential parameter. The expansion characters and the rank numbers are then automatically replaced by the corresponding arguemnt values used in the simulation. Expansion characters can be applied to all types of arguments, however arguments of matrices and temporal parameters are not replaced by the argument value, but by the rank number of the argument (how they appear in the settings file after the parameter name) due to their big size. So each sequential parameter can be addressed by its rank allowing to build separate filenames. If not all sequential parameters are addressed by the filename, i.e if the base name is not unique for each simulation, the rank of the simualtion is added as suffix to the filename to avoid overwritting the output. Note, that after the rank number (which may consists of several digits) a character must follow which cannot be interpreted as a number by quantiNEMO. For example the first sequential parameter has to be called as "name_%1_4K" and not as "name_%14K". If the filename for the example above was set to "sim_%2pop_%1ind" the following base names would be generated (alphabetically patch_capacity comes before pach_number):

	filename
1. simulation	sim_10pop−5ind
2. simulation	sim_10pop−10ind
3. simulation	sim_10pop−20ind
1. simulation	sim_50pop−5ind
5. simulation	sim_50pop−10ind
6. simulation	sim_50pop−20ind

Chapter 3

Life Cycle

quantiNEMO is a discrete generation-based simulator. This means that a single individual undergoes only once the life cycle and that the generations are not overlapping. Depending on the parameterization in the settings file some events may be skipped. The life cycle has a fixed order of events. A simulation starts with "Breeding", i.e. only adults are present at the initialization of the simulation. The life cycle is repeated for each generation, i.e. the life cycle event "Breeding" follows the life cycle event "Extinction":

1 : Breeding. Adults mate and may produce offspring. Selection acts at this stage.

2 : Statistics. Adults and juveniles are present. It is the stage were summary statistics may be computed for adults and juveniles.

3 : Outputs. Adults and juveniles are still present. Genotypes and/or phenotypes may be dumped to files for adults and/or juveniles.

4 : Aging. It is the event where the adults are removed. Only the juveniles remain in the model.

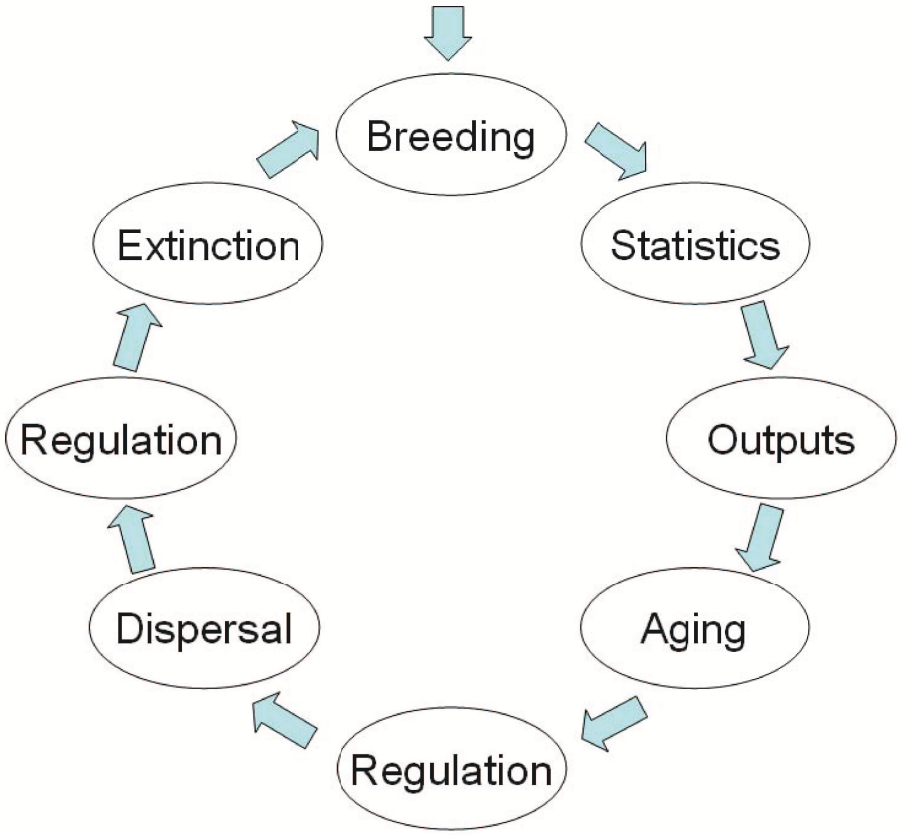
5 : Regulation. Before dispersal some patches may be overcrowded. This regulation stage allows to control the population sizes.

6 : Dispersal. Juveniles may migrate to other patches and become adults.

7 : Regulation. After dispersal some patches may be overcrowded. This regulation stage allows to control the population sizes.

8 : Extinction. Due to stochastic events populations may go extinct.

In the following the life cycle events and their options are described in details:



3.1 Breeding

This stage performs mating and breeding of the new offspring generation following the mating system chosen. Adults are not removed here (adults are removed in the event aging). This is the event were selection acts. The reproduction model implemented in quantiNEMO is composed of two phases: In the first phase for each patch the total number of offspring to be produced is computed. This number of offspring depends on the two parameters breed_model and mating_nb_offspring_model. In a second phase for each offspring to be produced a pair of local (of the same patch) parents is assigned. This assignment of the parents to the offspring depends on the mating system (parameter mating_system) and on the fitness of the parents if selection acts. Thereby, adults with a higher fitness have on average a higher reproductive success. If no selection acts, i.e. no quantitative trait undergoes selection, all parents have a fitness of 1, thus the assignment of the parents to the offspring depends solely on the mating system. The following parameters allow to parameterize this life cycle event:

breed_model [0,1,2] (default: 0)
: This parameter specifies if and how selection acts at the reproduction stage. Selection acts on the phenotype of the quantitative traits (see section 6 for

more details). If no selection acts, i.e. no quantitative trait undergoes selection, all breeding models result in the same outcome, respectively this parameter is ignored.

0 : soft selection. Selection acts locally at the patch level, i.e. patches do not interact (Wallace, 1975). The number of offspring of a patch depends solely on the parameter mating_nb_offspring_model. The parents for each offspring are randomly drawn taking into account their fitnesses (within a patch). Thus the reproductive output of the patch is independent of its mean fitness. Individual's fitness is relative to the mean fitness of its patch.

1 : soft/hard selection. Selection acts at the metapopulation level, i.e. the total number of offspring of the entire metapopulation is partitioned according to the mean fitnesses of the patches (Ravigne et al., 2004). Patches with a higher mean fitness produce on average more offspring. However, the total number of offspring of the entire metapopulation depends purely on the parameter mating_nb_offspring_model. Parents are then allocated to the offspring based on their fitnesses. In other words the fitness of the adults is relative to the mean fitness of the metapopulation.

2 : hard selection. The maximal number of offspring of a patch (i.e. if all individuals have a fitness of 1) is defined by the parameter mating_nb_offspring_model. The effective number of offspring produced in a patch is however adjusted by the mean fitness of the adults of this patch ($effective\_number = maximal\_number * mean\_fitness$). The parents for each offspring are randomly drawn taking into account their fitnesses (within a patch). Thus fitness is directly translated into a number of offspring.

An example illustrating the distribution of the offspring among two populations depending on the chosen breeding model (parameter breed_model). Both populations have a carrying capacity (K)of 1000 individuals, the parameter mating_nb_model is set to 0, and the mean fitnesses of the populations ($\bar{W}$) are 0.8 and 0.4, respectively:

breeding model	population 1 $K = 1000$ $\bar{W} = 0.6$	population 2 $K = 1000$ $\bar{W} = 0.2$	total
0 : soft	1000	1000	2000
1 : soft/hard	1500	500	2000
2 : hard	600	200	800

mating_nb_offspring_model [0,1,2,3,4,5] (default: 0)

This parameter specifies how the total number of offspring (N_{Off}) is determined. This parameter depends on the previous parameter breed_model. In

case of soft selection (breed_model set to 0) this parameter specifies how to compute the total number of offspring for each patch separately. In case of soft/hard selection (breed_model set to 1) this parameter specifies how to compute the total number of offspring of the entire metapopulation. This total number of offspring of the entire metapopulation is then distributed among the patches based on the mean fitnesses of the patches. In case of hard selection (breed_model set to 2) this parameter specifies the total number of offspring per patch assuming a maximal fitness of 1 for all adults.

0 : carrying capacity. $N_{Off} = K$
The total number of offspring (N_{Off}) is set to the carrying capacity (K, parameter patch_capacity) of the patch or the metapopulation, respectively.

1 : keep number. $N_{Off} = N$
The total number of offspring (N_{Off}) corresponds to the number of adults (N), i.e. the number of individuals is kept constant. Note that a regulation of the patch densities after dispersal can lead to an unwanted continuing reduction of the entire metapopulation size.

2 : fecundity. $N_{Off} = N_F f$
The number of offspring (N_{Off}) depends on the mean fecundity of the females (f) defined by the parameter mean_fecundity.

3 : fecundity (stochastic). $N_{Off} = Poisson(N_F f)$
Same as point 2, but the computation of the total number of offspring has a stochastic component.

4 : logistic regulation. $N_{Off} = \frac{NK(1+r)}{N(1+r)-N+K}$
The total number of offspring (N_{Off}) is logistically regulated (Beverton and Holt, 1957) and depends therefore on the carrying capacity (K) and on the parameter growth_rate (r).

5 : stochastic logistic regulation. $N_{Off} = Poisson(\frac{NK(1+r)}{N(1+r)-N+K})$
Same as point 4, but the computation of the total number of offspring has a stochastic component.

Models and their specific parameters

model	additional parameters
0 : carrying capacity	
1 : keep number	
2 : fecundity	mean_fecundity
3 : fecundity (stochastic)	mean_fecundity
4 : logistic regulation	growth_rate
5 : stochastic logistic regulation	growth_rate

growth_rate [decimal]
This parameter is mandatory (and only used) if logistic regulation is used to

specify the number of offspring (i.e. parameter breed_model set to 4 or 5). It specifies the growth rate of the population.

mean_fecundity [integer]
This parameter specifies the mean female fecundity. The parameter is mandatory (and only used) if the fecundity of the female specifies the number of offspring (i.e. parameter breed_model set to 2 or 3).

mating_system [0,1,2,3,4] (default: 0)
Five general mating systems are implemented in quantiNEMO. The assignment of the parents to the offspring is random depending on the fitness of the local parents (if no selection acts all individuals have a fitness of 1). Thereby, adults with a higher fitness have on average a higher reproductive success):

0 : random mating (hermaphrodite). For each offspring two hermaphrodite parents are randomly assigned. With probability $1/N$ these two hermaphrodites are identical which leads to selfing. Females are used to simulate hermaphrodites.

1 : selfing (hermaphrodite). For each offspring a hermaphrodite is randomly assigned to self fertilize. The parameter mating_proportion allows to set the proportion of outcrosses. quantiNEMO controls that the proportion of outcrosses is met, i.e. that outcrossing does not result by chance ($1/N$) in selfing. Females are used to simulate hermaphrodites.

2 : random mating (promiscuity). This is random mating with two sexes. For each offspring a father and a mother are randomly assigned.

3 : polygyny. Depending on the parameter mating_males only one (default) or several males per patch may reproduce. This fixed number of reproductive males are selected randomly depending on their fitnesses, i.e. the reproductive males have on average a higher fitness. Then for each offspring a mother and one of these reproductive males are randomly assigned depending on their fitnesses. Thus reproductive males with higher fitnesses have a higher reproductive success among the reproductive males. If no selection acts (parameter breed_model set to 3) the reproductive males are randomly chosen (all males have the same probability) and each male has the same probability to father an offspring. The parameter mating_proportion allows to set the proportion of random matings between any male and female, i.e. also males of the non-reproductive group may get the chance to reproduce.

4 : monogamy. For each female a male is randomly assigned to be its partner for all offspring. If there are less females than males present in the patch, not all males will mate. In contrast, if there are more females than males present in the patch, males may belong to several mating pairs. For each offspring a parent pair is randomly assigned depending on the fitness of the female (if no selection acts the parent pairs have the same probability

to be selected). Thus parent pairs, where the females have a higher fitness have on average a higher reproductive success. The parameter mating_proportion allows to set the proportion of random matings.

Models and their specific parameters

model	additional parameters
0 : random mating (herma.)	
1 : selfing (herma.)	mating_proportion
2 : random mating (prom.)	sex_ratio
3 : polygyny	sex_ratio / mating_proportion / mating_males
4 : monogamy	sex_ratio / mating_proportion

mating_proportion [decimal] (default: 1)

This parameter allows to specify the ratio of a special mating system in relation to random mating. A value of 1 (default) means that only the special mating occurs and a value of 0 means that only random mating occurs. For example if we want to simulate a plant with a selfing rate of 90% we have to set the parameter mating system to 1 (selfing) and this parameter mating_proportion to 0.9. These settings will lead to 90% selfing and 10% random mating. Note, that quantiNEMO controls that the ratio is met, i.e. that selfing does not occur by chance (probability would be $1/N$) when random mating should occur.

mating_males [integer] (default: 1)

This parameter sets the number of males that will be available for mating within each patch. The parameter will only be used if the mating system is polygyny (parameter mating_system set to 3). The range of values is between 1 (a single male mates) and the carrying capacity of the males (all males may mate).

sex_ratio [decimal] (default: 1)

This parameter allows to specify the ratio of males to females of the offspring in a patch. If hermaphrodites are simulated (parameter mating_system is set to 0 or 1) the sex ratio is not considered, respectively set to 0 (females are used to simulate hermaphrodites).

3.2 Statistics

After breeding has taken place it is possible to record summary statistics specified by the parameter stat. Most of the summary statistics can be recorded for offspring and/or adults and for females and/or males. It is also possible to set the frequency (parameter log_time) of the recording. At the end of a simulation the summary statistics are written to a text file. There is the choice to print the summary statistics individually per replicate and/or summed up across replicates (mean and variance

across replicates). In this latter case an additional statistic named *alive.rpl* will be added which contains the number of alive replicates, i.e. the number of simulations where the populations did not get extinct. If no summary statistics are computed this event is skipped.

stat_save [0,1,2,3,4,5] (default: 0)

This parameter specifies if the summary statistics should be computed and how they should be dumped to file. The summary statistics may be dumped to file for each specified generation and replicate separately (file "generic_name_stats.txt"), or summary statistics may be summed up across replicates by their mean (file "generic_name_stats.txt") and their variance (file "generic_name_var.txt").

0 : All. Output includes all types of summary statistic (files "generic_name_stats.txt", "generic_name_mean.txt", and "generic_name_var.txt").

1 : Detailed. Output includes only the file containing the summary statistics for each replicate separately (file "generic_name_stats.txt").

2 : Summed up. Output includes the files containing the summary statistics summed up by their mean and variance across replicates (files "generic_name_mean.txt", and "generic_name_var.txt").

3 : Mean. Output includes only the file containing the summary statistics summed up by their mean across replicates (file "generic_name_mean.txt").

4 : Variance. Output includes only the file containing the summary statistics summed up by their variance across replicates (file "generic_name_var.txt").

5 : None. No summary statistics are written. The life cycle event "Statistics" is skipped.

Whenever the summary statistics are output (parameter stat_save not set to 3) a file named "generic_name_legend.txt" containing a small description of the summary statistics is also generated.

stat_log_time [integer] (temporal/default: 1)

This is the time interval at which summary statistics are recorded. The interval must range between 1 and the number of generations. Since the parameter may change over time (temporal parameter) the summary statistics may be computed for any generation:

```
stat_log_time (1 1, 10 10, 100 100)
```

In this example for the first 9 generations the summary statistics are computed every generation, from the tenth until generation 99 they are computed at every tenth generation, and from the generation 100 every hundred generation.

stat_dir [string] (default: "")
This parameter is used to specify a subdirectory within the simulation folder (parameter folder) where the summary statistic files will be stored. If the parameter is not set, the files will be stored in the simulation folder.

stat [string/matrix]
This parameter allows to specify the summary statistics to be computed. The arguments are key words standing for one or multiple summary statistics. If multiple key words are passed they have to be written as a matrix within brackets separated by space. The available key words and their corresponding summary statistics are listed in the corresponding simulation component section in this manual. Summary statistics about the demography may be found in section 5.3, summary statistics about quantitative traits may be found in section 6.7, and summary statistics about neutral markers may be found in section 7.6. If no arguments are specified this event will be skipped. If summary statistics are specified, which cannot be computed since the corresponding component is missing (e.g. F-statistics of neutral loci can only be computed if neutral markers are simulated) a warning will be given.

```
stat {n.adlt.fstat
      # q.adlt.fstat
      quanti
      adlt.demo}
```

In this example the summary statistics for the key words *n.adlt.fstat*, *quanti*, and *adlt.demo* are considered by quantiNEMO, while the key word *q.adlt.fstat* is commented out.

3.3 Output

quantiNEMO can also produce files with the raw genetic and phenotypic data. Genotypes of neutral markers and quantitative traits can be dumped to file in the FSTAT format (Goudet, 1995). For quantitative traits, the phenotypes may also be written to a file for any generation, sex or age. The selection of the various outputs is done in the corresponding simulation components (neutral genotype see section 7.5, quantitative trait genotype see section 6.6.1, and quantitative trait phenotype see section 6.6.3). If no output is desired this event is skipped. After each output a script may be launched to process the output.

3.4 Aging

This life cycle event simply removes the adults present.

3.5 Regulation offspring

This event performs population regulation before dispersal, i.e. at the offspring stage. Due to dispersal some patches may be overcrowded. This life cycle event allows to regulate the population sizes down to carrying capacity. In fact the regulation should only be used if there is no regulation at the breeding stage, i.e. if the parameter breed_model is set to 1 (keep number), 2 (fecundity), or 3 (fecundity stochastic).

regulation_model_offspring [0,1] (default: 0)

0 : no regulation. No population size regulation takes place at the offspring stage, i.e. overcrowding can occur.

1 : random regulation. For each patch quantiNEMO regulates the population size down to its carrying capacity if the population size exceeds carrying capacity. Individuals are thereby randomly removed. No regulation takes place if the population size is lower than carrying capacity.

3.6 Dispersal

This life cycle event allows the exchange of individuals between populations. The dispersal rates may vary among patches, sexes and time. Several dispersal models are available (see parameter dispersal_model). It is also possible to specify a dispersal matrix, which will have precedence over other dispersal parameters. After dispersal, individuals become adults. By default (if none of the following parameters are set) individuals do not disperse among patches.

dispersal_rate
dispersal_rate_fem
dispersal_rate_mal [decimal/matrix] (temporal/default: 0)

These parameters allow to set the emigration rate. If the argument is a single value the dispersal model used depends on the other parameters of this section. But it is also possible to specify the dispersal rate explicitly between each pair of patches (for both directions) if the argument is a matrix. A dispersal matrix has precedence over all other dispersal settings. The matrix must be `patch_number` x `patch_number` in dimensions. Each d_{ij} element of this matrix is the dispersal probability from patch i to patch j, where i specifies the row and j the column of the matrix. Consequently the values in a row must sum up to 1. The dispersal rates can either be specified for both sexes in general or for each sex separately. If the dispersal rates are sex specific the dispersal rates for both sexes have to be specified and they have to be in the same format

(matrix or a single dispersal rate). Sex specific dispersal rates have precedence over a general dispersal rate. Note, that the dispersal matrix has to be fully specified, i.e. the matrix is not adjusted to the number of patches as for other parameters.

dispersal_model [0,1,2,3] (default: 0)
The following dispersal models can be specified, if the dispersal rate is a single rate:

0 : Migrant-pool Island model. If the dispersal rate is m and the number of patches is n_p, the probability to disperse to any $n_p - 1$ non-natal patch is $\frac{m}{n_p-1}$ while the probability to stay at home is $1 - m$.

1 : Propagule-pool Island model. In that modified version of the Island model a proportion of emigrants from a patch (parameter dispersal_propagule_prob, φ) disperse to the same non-natal patch. This propagule patch varies among patches and is reassigned at each generation. Each offspring of a patch has a probability $m\varphi$ to migrate to this propagule patch. With probability $\frac{m(1-\varphi)}{n_p-2}$, it will disperse to any patch but its natal or propagule patch. With a probability of $1 - m$ it will stay at home.

2 : 1D Stepping-Stone model. In the one dimensional Stepping Stone model patches are placed on a line and migrants can only move to one of the two adjacent patches. If the dispersal rate is m, the probability to disperse to one of the adjacent patches is $m/2$ while the probability to stay at home is $1-m$. The parameter `dispersal_border_model` allows to specify how to treat the border patches.

3 : 2D Stepping-Stone model. In the two dimensional Stepping-Stone model patches are placed on a grid (or lattice) and migrants can move to 4 or 8 adjacent patches (set by the dispersal_lattice_range parameter below). If the dispersal rate is m, the probability to disperse to one of the adjacent patches is $m/4$ or $m/8$ depending on the the parameter dispersal_lattice_range, while the probability to stay at home is $1 - m$. The parameter dispersal_border_model allows to specify how to treat the border patches and the parameter dispersal_lattice_dims allows to specify the dimensions of the grid.

Models and their specific parameters

model	additional parameters
0 : Migrant-pool Island	
1 : Propagule-pool Island	dispersal_propagule_prob
2 : 1D Stepping-Stone	dispersal_border_model
3 : 2D Stepping-Stone	dispersal_border_model / dispersal_lattice_range / dispersal_lattice_dims

dispersal_lattice_range [0,1] (default: 0)

This parameter sets the number of neighboring patches used for dispersal. The dispersal probabilities to these adjacent patches are $m/4$ in the first case and $m/8$ in the second. This parameter is only used in the 2D Stepping-Stone model (parameter dispersal_model set to 3).

0 : 4 neighbors. 4 adjacent patches (up, down, left and right)

1 : 8 neighbors. 8 adjacent patches (as before plus the diagonals)

dispersal_border_model [0,1,2] (default: 0)

This parameter specifies how the patches at a border of the Stepping Stone model should be treated:

0 : Circle/Torus. In the 1D Stepping-Stone model the first and last patches are connected to each other by migration leading to a circle. In the 2D Stepping-Stone model individuals of an edge patch may migrate to the other side leading to a torus (donut world). This means that there are no edges, eliminating any such effects.

1 : Reflective boundaries. The borders are reflective. Dispersers from the border patches cannot move beyond the border. Border cells have thus less cells connected to them and their dispersal probabilities to the adjacent patches are higher (e.g. m for the 1D Stepping-Stone model, $m/3$ (corners $m/2$) for the 2D Stepping-Stone model with four adjacent cells, and $m/5$ (corners $m/3$) for the 2D Stepping-Stone model with eight adjacent cells). No dispersers are lost.

2 : Absorbing boundaries. Dispersers of the border patches are lost if they choose to move beyond the border. The dispersal probabilities of a border patch are not modified.

dispersal_lattice_dims [matrix]

This parameter allows to specify the length and width of the 2D Stepping-Stone lattice (only used when dispersal_model is set to 3). The argument is an integer matrix with two values. The first value stands for the number of rows, and the second value for the number of columns. The product of the two values results in the number of patches and thus must match the parameter patch_number. If the parameter is not set quantiNEMO assumes that the 2D Stepping-Stone lattice is quadratic. If this is not possible due to the number of patches an error is returned.

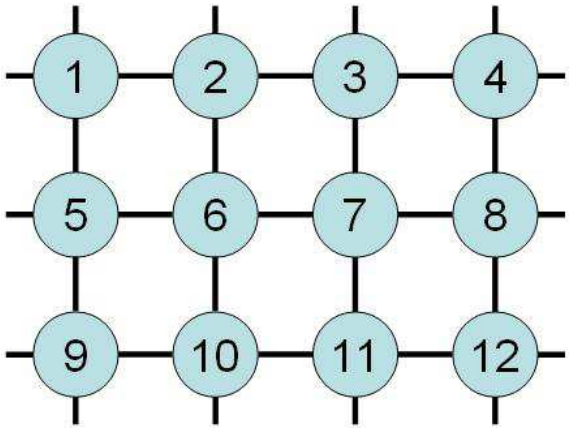
dispersal_propagule_prob [decimal] (temporal/default: 1)

This parameter is only used for the Propagule-pool Island model (parameter dispersal_model set to 1). It specifies the probability that a migrant will move to the propagule-assigned patch, i.e. this is also the proportion of emigrants of a patch which migrate to the same non-natal patch. A probability of 1

means that all emigrants migrate to the same non-natal patch, while a value of 0 means that all emigrants migrate to any patch, but the natal and the propagule patch.

Example
The following example defines a metapopulation of 12 patches arranged in a two dimensional grid shown below:

```
patch_number            12
dispersal_model         3
dispersal_lattice_dims  {4 3}
dispersal_rate          0.1
```

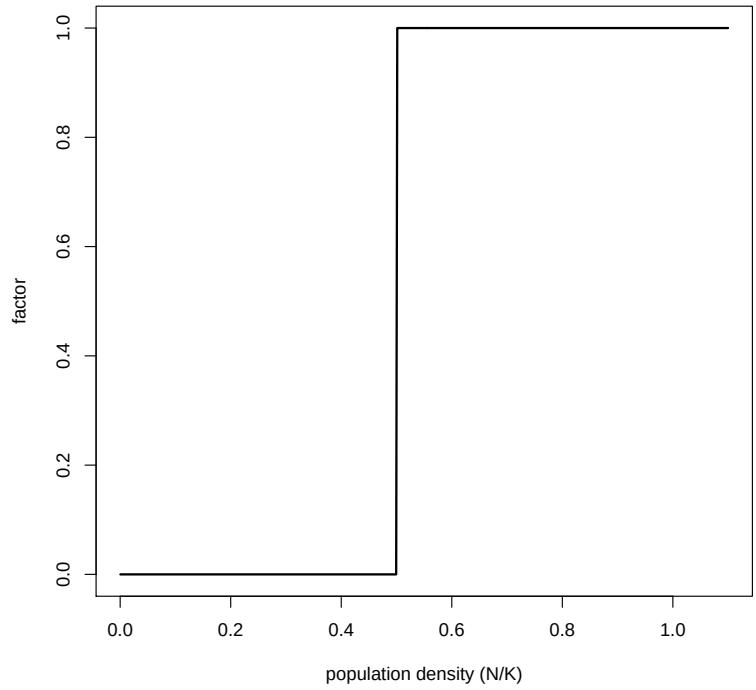


3.6.1 Density dependent dispersal rate

By default the dispersal rate is not influenced by the population size. The following parameters allow to define a generalized logistic function (Richards, 1959) to specify a relationship between the dispersal rate and the population density (population size divided by the carrying capacity). The generalized logistic function defines a factor (f) which is then multiplied with the dispersal rate (parameter dispersal_rate):

$$f = min + \frac{max - min}{(1 + s * e^{r(D_{r_{Max}} - D)})^{1/s}}$$

Where min is the lower asymptote (parameter dispersal_min), max is the upper asymptote (parameter dispersal_max), r is the growth rate (parameter dispersal_growth_rate), $D_{r_{Max}}$ is the population density with the maximal slope (parameter dispersal_max_growth), D is the current population density (N/K), and s defines the symmetry of the curve (parameter dispersal_symmetry).



In this figure the growth rate of the curve has a very high value of $1e^4$ (default value of parameter dispersal_growth_rate). This results in a sharp change of the factor from the lower asymptote (parameter dispersal_k_min) to the upper asymptote (parameter dispersal_k_max): Below the threshold (parameter dispersal_max_growth), i.e. when the population density is low no migration occurs since the factor is zero. If the population size is larger than the half of the carrying capacity the patch sends emigrants at the set dispersal rate (parameter dispersal_rate) since the factor is 1.

dispersal_k_min [decimal] (temporal/default: 0)
This parameter specifies the lower asymptote of the slope.

dispersal_k_max [decimal] (temporal/default: 1)
This parameter specifies the upper asymptote of the slope.

dispersal_k_max_growth [decimal] (temporal/default: -1)
This parameter specifies the population density with the maximal change.

dispersal_k_growth_rate [decimal] (temporal/default: $1e^4$)
This parameter specifies the slope of the curve. The high default value of $1e^4$ implies that the slope is more or less vertically, i.e. that the change between lower and higher asymptote is instantaneously. Thus the parameter dispersal_k_max_growth serves as threshold of the population density below which

the factor is set to the value of the parameter dispersal_k_min. Above the threshold the factor is set to the value of the parameter dispersal_k_max.

dispersal_k_symmetry [decimal] (temporal/default: 1)
This parameter specifies the symmetry of the curve. The default value of 1 results in a symmetric curve.

3.7 Regulation adults

This event performs population regulation after migration. Due to dispersal some patches may be overcrowded. This life cycle event allows to regulate the population sizes down to carrying capacity. In fact the regulation should only be used if there is no regulation at the breeding stage, i.e. if the parameter breed_model is set to 1 (keep number), 2 (fecundity), or 3 (fecundity stochastic).

regulation_model_adults [0,1] (default: 0)

0 : no regulation. No population size regulation takes place at the adults stage, i.e. overcrowding can occur.

1 : random regulation. For each patch quantiNEMO regulates the population size down to its carrying capacity if the population size exceeds carrying capacity. Individuals are thereby randomly removed. No regulation takes place if the population size is lower than carrying capacity.

3.8 Extinction

This event allows to randomly wipe out populations. The probability that a population goes extinct is specified by the parameter extinction_rate. If a patch goes extinct, all individuals of the patch are removed. If the extinction rate is zero this event is skipped.

extinction_rate [decimal] (default: 0)
Extinction probability of a patch at each generation.

Chapter 4

Simulation

This section describes general parameters of a simulation:

generations [integer]
Number of generations to perform per replicate. This parameter is mandatory, and has no default.

replicates [integer] (default: 1)
Number of replicates to perform per simulation. Replicates are identical simulations (in terms of parametrization), but their outcome may differ due to stochastic events.

folder [string] (default: "simulation_yyyy-mm-dd_hh-mm-ss")
This parameter specifies the folder for the output. The default argument of this parameter differs from the rules, as the default folder name is dynamic, i.e. comprises the start time and date of the simulation. The default folder name is "simulation_" with the date and time as suffix in the format "yyyy-mm-dd_hh-mm-ss" (Year-Month-Day_Hour-Minute-Second). This dynamic default folder name allows to store each simulation separately, avoiding that previous outputs are overwritten. If the parameter is set, the passed argument will be used as folder name (i.e. without any addition of the time). In this case it may happen that the new output wants to overwrite previous outputs. The parameter overwrite (see below) allows to specify the rules for overwriting other outputs. If the output should be stored directly in the working directory this parameter has to be listed in the settings file followed by an empty argument "" (for this special parameter an empty argument is not identical to a missing one).

overwrite [integer] (default: 0)
This parameter specifies how present output is treated if there is a danger of overwriting:

0 : no. If the output of the running simulation will overwrite present files, quantiNEMO will ask the user how to proceed (*Do you want to overwrite all the files that use it ? (y(es)/n(o)/s(kip)):*), and will suspend the simulation until the user responds to the question.

1 : yes. Present output is overwritten without asking.

filename [string] (default: "simulation")
This name will be used as the base filename for all outputs. The output file extensions are added to this base filename. If a file is written on a replicate-periodic basis, the replicate number will be added between the base name and the extension, so that the same file is not overwritten periodically. The same is true concerning generation-periodic files (see section 2.4).

The base name may include the special expansion character '%' used to build filenames when sequential parameters are present in the input parameter file. See the discussion on sequential parameters in section 2.7.2.

logfile [string] (default: "quantinemo.log")
This is the file name (including the extension) for the log file in which the simulation logs are recorded. This log file is stored next to the executable and records the main information of each simulation, such as the elapsed time for the simulation and the replicates and the time of the start and end of a simulation. By default, quantiNEMO will save all this information in a file named "quantinemo.log".

logfile_type [0,1,2] (default: 0)
For each simulation a log file is created containing the settings of the simulation. The file is created for each simulation and is stored in the simulation folder (parameter folder). The name of the log file is composed of the base name (parameter filename) and the suffix ".log". There are three possibilities to store this file:

0 : as input. The file contains the same parameters as the settings file, plus the parameter seed.

1 : minimal. The file contains a minimal set of parameters still able to perform the same simulation. All parameters of the settings file which were set to the default value are not reported.

2 : maximal. The file contains all parameters, including the ones not set in the settings file. For each parameter the type, the default value, if it is a temporal parameter, and a possible range limit is added as comment.

seed [integer/matrix] (default: -1)
This parameter specifies the seed which will be used to initialize the random number generator. It is possible to pass a single number as seed in the range

of 0 to 4,294,967,295 depending on the computer system (corresponding to an unsigned long in C++). To increase the number of possible seeds it is possible to pass an array of seeds (up to 624) to quantiNEMO if specified as a one dimensional matrix. If the seed is not set (i.e. is set to -1) the random generator is initialized by the time, i.e. a matrix with two seeds is used. Thereby the first number is the time in seconds since 1.1.1970 (function *time(NULL)* in C++), and the second number is the number of clock ticks elapsed since quantiNEMO started (function *clock()* in C++).

```
seed {2354135 20234510 30345120 456300 50363450}
```

postexec_script [string] (default: "")

genetic_map_output [0,1] (default: 0)

quantiNEMO allows simulating linked neutral markers and QTLs. The sections 6.5 (for QTLs) and 7.4 (for neutral loci) describe how to specify the corresponding genetic map. The parameter genetic_map_output allows to specify if the underlying genetic map is dumped to file or not.

0 : output. The genetic map is dumped to file for every replicate if the genetic map was simulated, i.e. if at least one of the parameters quanti_loci_positions, quanti_loci_positions_random, ntrl_loci_positions, and ntrl_loci_positions_random are set. The output is named "genetic_map.txt" and is stored in the simulation folder. If multiple replicates are simulated a replicate counter (example _r3) is inserted before the extension.

1 : no output. The genetic map is never dumped to file.

sampled_patches [integer/matrix] (default: 0)

By default summary statistics and all other outputs (genotypes, genotypic and phenotypic values) consider all patches. The parameter **sampled_patches** allows to make a selection among the patches. In this case the summary statistics are only computed among the sampled patches and genotypes, genotypic and phenotypic values are only dumped to file for these selected patches. This parameter thus allows for example to investigate different sampling schemes. There are different methods to define the sampled patches:

matrix. Using a one dimensional matrix as argument allows to define explicitly the patches to sample. Please note that for this parameter a matrix has to be defined fully, i.e. all patches to sample have to be listed in the matrix and a single number is not expanded to a matrix.

number. If the argument is a single number the patches to sample are drawn randomly. In this case the passed number specifies the total number of patches to sample. Patches are drawn randomly for each replicate, but remain the same during a simulation.

0. This is the default value. Also a single number, but in this case all patches are sampled.

Chapter 5

Metapopulation

This section describes general parameters concerning the metapopulation:

patch_number [integer] (default: 1)
This parameter specifies the number of patches in the metapopulation.

5.1 Population sizes

This section allows to specify the carrying capacities of the patches and the initial populations.

patch_capacity
patch_capacity_fem
patch_capacity_mal [integer/matrix] (temporal)
These parameters allow to set the carrying capacities of the patches. The carrying capacity of a patch is the maximal number of individuals that a patch may support. The carrying capacities may vary among sexes, patches, and time. The carrying capacities have to be specified either for each sex separately (parameters patch_capacity_fem and patch_capacity_mal), or for both sexes together (parameter patch_capacity). In the first case both sex specific parameters have to be set if two sexes are simulated. In the latter case the carrying capacities for females and males are assumed to be identical, i.e. the carrying capacity of females and males is *patch_capacity*/2. If hermaphrodites are simulated the parameters patch_capacity and patch_capacity_fem are identical. In case all three parameters are set, only the sex specific parameters will be used as they are more informative. To set the carrying capacities individually for each patch a matrix is needed. The matrix is adjusted to the number of patches if necessary (see section 2.3.5.4). If all patches have the same carrying capacities a single number as argument is sufficient to specify the

carrying capacities. If the initial population sizes (parameter patch_ini_size) are not specified the simulation will start with all the populations set at carrying capacity.

Examples
The following example defines a metapopulation of 5 patches each with a carrying capacity of 100 individuals:

```
patch_number    5
patch_capacity  100
```

If the carrying capacities vary between patches the following definition with a matrix may be used:

```
patch_number    5
patch_capacity  {100  200  300  400  500}
```

In the following example, however the matrix of the parameter patch_capacity will be adjusted by quantiNEMO to meet the number of patches. This will result in 10 patches with the carrying capacities 100, 200, 300, 400, 500, 100, 200, 300, 400, and 500.

```
patch_number    10
patch_capacity  {100  200  300  400  500}
```

If the carrying capacities are defined for each sex separately both sex specific parameters must be present if two sexes are simulated:

```
patch_number        5
patch_capacity_fem  {100  200  300  400  500}
patch_capacity_mal  {200  400  600  800  1000}
```

In the following example the carrying capacity is specified with too many parameters. As the sex specific parameters are more informative, they have precedence over the global setting of carrying capacities (1000) which will therefore be ignored:

```
patch_number        5
patch_capacity      1000
patch_capacity_fem  {100  200  300  400  500}
patch_capacity_mal  {200  400  600  800  1000}
```

patch_ini_size
patch_ini_size_fem
patch_ini_size_mal [integer/matrix]

These parameters allow to set the initial population sizes of the patches, i.e. the populations sizes present at the beginning of the simulation. These parameters are optional. If none of these parameters is set, i.e. the initial population

sizes are not set, the simulation will start with all the population sizes set at carrying capacity. The initial population sizes may vary among sexes, and patches. The initial population sizes have to be specified either for each sex separately (parameters patch_ini_size_fem and patch_ini_size_mal), or for both sexes together (parameter patch_ini_size). In the first case both sex specific parameters have to be set if two sexes are simulated. In the latter case the initial population sizes for females and males are assumed to be identical, i.e. the initial population size of females and males is $patch\_ini\_size/2$. If hermaphrodites are simulated the parameters patch_ini_size and patch_ini_size_fem are identical. In case all three parameters are set the sex specific parameters will be used as they are more informative. To set the initial population sizes individually for each patch a matrix is needed. The matrix is adjusted to the number of patches if necessary (see section 2.3.5.4). If all patches have the same initial population sizes a single number as argument is sufficient to specify the initial population sizes.

5.2 Selection pressure

Phenotypes for quantitative traits (see section 6) may be under selection. Selection pressures may vary among quantitative traits, sexes, patches, and time. To specify the selection pressure individually for quantitative traits and patches, matrices may be used. They are adjusted to the number of quantitative traits and to the number of patches if needed (see section 2.3.5.4). If a parameter does not change among quantitative traits and patches, a single value may be used as argument. Selection pressures have to be specified either for each sex separately (parameters with the suffix "_fem" for females and "_mal" for males), or for both sexes together (parameters without a suffix). In the first case both sex specific parameters have to be set if two sexes are simulated. In the latter case the selection pressure of females and males are assumed to be identical. Each row of the matrix corresponds to a quantitative trait, each column to a patch:

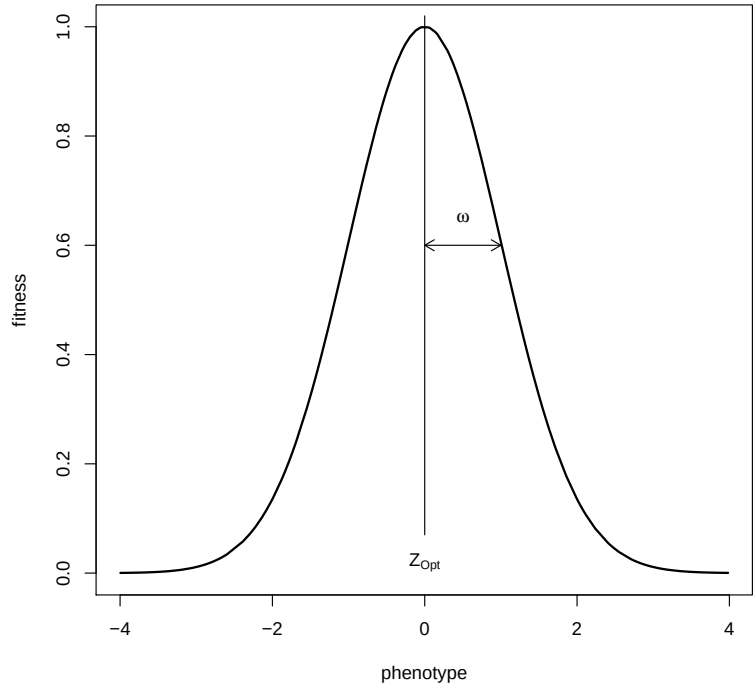
```
{ {patch_1 patch_2 . . . patch_n}    # trait 1
  {patch_1 patch_2 . . . patch_n}    # trait 2
  ...
  {patch_1 patch_2 . . . patch_n} }  # trait m
```

quantiNEMO supports two types of selection, stabilizing and directional selection. The type of selection may vary among quantitative traits, but not among patches and sexes. Selection pressure may change through time, allowing simulating a dynamic environment such as global warming.

5.2.1 Stabilizing selection

Stabilizing selection may act on the phenotype (P) of quantitative traits. The selection pressure is defined by the parameters patch_stab_sel_optima (Z_{Opt}) and patch_stab_sel_intensity (ω). The fitness (W) of a quantitative trait is computed using the following standard Gaussian function for stabilizing selection:

$$W = e^{\left(-\frac{\left(P-Z_{Opt}\right)^2}{2\omega^2}\right)}$$



patch_stab_sel_optima
patch_stab_sel_optima_fem
patch_stab_sel_optima_mal [decimal/matrix] (temporal/default: 0)
These parameters allow to set the selection optimum z_{Opt} for each patch and quantitative trait.

patch_stab_sel_intensity
patch_stab_sel_intensity_fem
patch_stab_sel_intensity_mal [decimal/matrix] (temporal/default: 1)
These parameters allow to set the selection intensity ω for each patch and quantitative trait. A small value results in a strong selection pressure, whereas a large value results in a weak selection pressure.

patch_stab_sel_optima_var [decimal/matrix] (default: 0)
This parameter specifies the variance of the normal distribution by which the selection optimum varies between generations (e.g. annual fluctuations of the mean temperature). By default the local selection optimum does not vary.

patch_stab_sel_intensity_var [decimal/matrix] (default: 0)
This parameter specifies the variance of the normal distribution by which the selection intensity varies at each generation (e.g. annual fluctuations of the mean temperature). By default the local selection intensity does not vary.

Example

```
patch_number                2
quanti_nb_trait             3
patch_stab_sel_optima       {{-0.1   0.1}
                             {  0.2   0.2}
                             {-0.3   0.3}}
patch_stab_sel_intensity    1
```

In this example the environment consist of two patches with varying selection pressures. Three quantitative traits are simulated. The first trait has a selection optimum at -0.1 in patch 1 and at 0.1 in patch 2. The selection optimum of the second trait is the same in both patches (0.2). The third trait has an optimum at -0.3 in patch 1 and at 0.3 in patch 2. The intensity of the selection is identical for all three traits and in both patches.

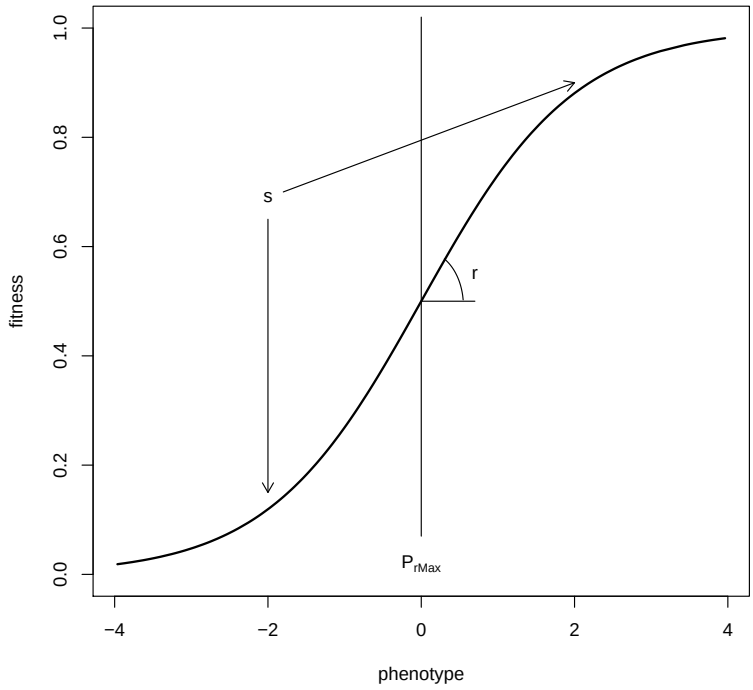
5.2.2 Directional selection

Directional selection may act on quantitative traits. The fitness (W) of a quantitative trait is computed using the following generalized logistic function (Richards, 1959):

$$W = min + \frac{max - min}{(1 + s * e^{r(P_{r_{Max}} - P)})^{1/s}}$$

Where min is the lower asymptote (parameter patch_dir_sel_min), max is the upper asymptote (parameter patch_dir_sel_max), r is the growth rate (parameter patch_dir_sel_growth_rate), $P_{r_{Max}}$ is the phenotype with the maximal slope (parameter patch_dir_sel_max_growth), and s defines the symmetry of the curve (parameter patch_dir_sel_symmetry; the curve is symmetric by default (value 1)).

patch_dir_sel_min
patch_dir_sel_min_fem



patch_dir_sel_min_mal [**decimal/matrix**] (**temporal/default: 0**)
: These parameters allow to set the lower asymptote of the selection curve for each patch and quantitative trait.

patch_dir_sel_max
patch_dir_sel_max_fem
patch_dir_sel_max_mal [**decimal/matrix**] (**temporal/default: 1**)
: These parameters allow to set the upper asymptote of the selection curve for each patch and quantitative trait.

patch_dir_sel_growth_rate
patch_dir_sel_growth_rate_fem
patch_dir_sel_growth_rate_mal [**decimal/matrix**] (**temporal/default: 1**)
: These parameters allow to set the slope of the selection curve for each patch and quantitative trait. If the argument is positive larger phenotypes have a higher fitness, while if negative smaller phenotypes have a higher fitness.

patch_dir_sel_max_growth
patch_dir_sel_max_growth_fem
patch_dir_sel_max_growth_mal [**decimal/matrix**] (**temporal/default: 0**)
: These parameters allow to set the phenotype with the maximal growth.

patch_dir_sel_symmetry

patch_dir_sel_symmetry_fem
patch_dir_sel_symmetry_mal [decimal/matrix] (temporal/default: 1)
These parameters allow to set the symmetry of the curve. The default value of 1 results in a symmetric slope.

patch_dir_sel_growth_rate_var [decimal/matrix] (default: 0)
This parameter specifies the variance of the normal distribution by which the selection slope varies at each generation (e.g. annual fluctuations of the mean temperature). By default the local selection slope does not vary.

patch_dir_sel_max_growth_var [decimal/matrix] (default: 0)
This parameter specifies the variance of the normal distribution by which the phenotype with maximal growth varies at each generation (e.g. annual fluctuations of the mean temperature). By default the local phenotype with maximal growth does not vary.

patch_dir_sel_symmetry_var [decimal/matrix] (default: 0)
This parameter specifies the variance of the normal distribution by which the symmetry of the curve varies at each generation (e.g. annual fluctuations of the mean temperature). By default the symmetry of the slope does not vary.

Example

```
patch_dir_sel_growth_rate 1
patch_dir_sel_max_growth  0
patch_dir_sel_symmetry    1
```

In this example the selection pressure for all patches and quantitative traits are identical and set to the default values. The specified directional selection pressure favors larger phenotypes (parameter patch_dir_sel_growth_rate is positive). This means that individuals with larger phenotypes have on average higher fitnesses and thus higher reproductive successes.

5.2.3 Multiple traits with varying types of selection

A special case arises if multiple quantitative traits are simulated with varying types of selection. The problem is how to specify the individual selection pressures. This is a more technical problem which I would like to illustrate here. First, quantiNEMO investigates which types of selection will be simulated based on the settings file. Then, quantiNEMO sets for each simulated type of selection the corresponding parameters assuming one selection type after the other that all quantitative traits have the same type of selection: assuming for example first that all quantitative traits are under stabilizing selection, then in a second step assuming that all quantitative traits are under directional selection. This detail is important to understand since

this makes it clear how a matrix is treated, i.e. how a matrix is expanded if needed. Of, course finally the selection pressure parameters are only set where needed, i.e. if the type of selection for a given quantitative trait requires the parameter. In other words the matrix of a selection pressure has to have the number of rows of the total number of traits, and not only of the traits with the corresponding selection pressure (this is controlled by quantiNEMO returning a warning if not met). If multiple quantitative traits are simulated with varying selection pressures it is handy to use the row indicator for the rows of the matrix. Example:

```
patch_number 3
quanti_nb_trait 5
quanti_selection_model 0
quanti_selection_model_3 1
patch_stab_sel_optima      {{1: 1 2 3}{2: 2 3 4}}
patch_dir_sel_growth_rate {{3: 4 5 6}{4: 7 8 7}{5: 2 3 2}}
```

In this example the first two quantitative traits are under stabilizing selection, whereas the three last quantitative traits are under directional selection. Using the row indicator it is possible to set the selection pressure directly for the required quantitative trait. However the following parameterization is equivalent to the upper one:

```
patch_number 3
quanti_nb_trait 5
quanti_selection_model 0
quanti_selection_model_3 1
patch_stab_sel_optima      {{1 2 3}{2 3 4}{9 9 9}{9 9 9}{9 9 9}}
patch_dir_sel_growth_rate {{9 9 9}{9 9 9}{4 5 6}{7 8 7}{2 3 2}}
```

In the example above the rows consisting of the number nine are read but then not taken into account, since the rows do not correspond to the correct quantitative traits. Caution if you are using matrix expansions since this may lead to unwanted configurations as shown below:

```
patch_number 3
quanti_nb_trait 5
quanti_selection_model 0
quanti_selection_model_3 1
patch_stab_sel_optima      {{1 2 3}}
patch_dir_sel_growth_rate {{4 5 6}{2 3 7}{2 8 3}}
```

In this example the growth rate are as follows:

```
1. trait: stabilizing selection optima: {1 2 3}
2. trait: stabilizing selection optima: {1 2 3}
3. trait: directional selection growth rate: {2 8 3}
4. trait: directional selection growth rate: {4 5 6}
5. trait: directional selection growth rate: {2 3 7}
```

Maybe this behavior was desired, but it could also well be that one anticipated the following specification which is wrong:

```
1. trait: stabilizing selection optima: {1 2 3}
2. trait: stabilizing selection optima: {1 2 3}
3. trait: directional selection growth rate: {4 5 6}
4. trait: directional selection growth rate: {2 3 7}
5. trait: directional selection growth rate: {2 8 3}
```

Why is this not the case? Assume that all quantitative traits are under directional selection. Thus the matrix of the growth rate has to be repeated leading to the following full matrix: {{4 5 6}{2 3 7}{2 8 3}{4 5 6}{2 3 7}}. This matrix expansion leads to a warning indicating that the number of rows is not an entire subset of the number of quantitative traits. Based on this matrix it is now obvious that the growth rate of the third quantitative trait (thus the first trait under directional selection) has the growth rates {2 8 3} and not {4 5 6}.

5.3 Summary statistics

The summary statistics listed in the table below are available for demography of the populations. The column Stat name contains the name of the summary statistic used to specify which summary statistics are computed (parameter stat, for details see section 3.2). These names appear also in the output file. The column Description contains a short description of the summary statistics.

Some of the summary statistics are available for adults and offspring (indicated by (adlt/off)). To obtain a certain summary statistic for adults the prefix adlt. has to be added to the summary statistic name (e.g. adlt.allnb), respectively the prefix off. to obtain the summary statistic for offspring (e.g. off.allnb).

Some of the summary statistics are computed for each patch separately (indicated by *(computed for each patch)* in the table). Depending on the number of patches this may lead to the computation of a large number of summary statistics. The names of such summary statistics are marked in the output file with the index of the patch: "_pX", respectively "_pX-Y" for pairwise summary statistics. X and Y stand for the patch index starting with 1.

The summary statistics are computed for every quantitative trait. If several quantitative traits are simulated the postfix "_tY" is added to the summary statistic name in the output file.

The summary statistic name (column Stat name) may be used to specify the summary statistic to be computed (e.g. adlt.nblnd). Similar summary statistics (within a thematic group) may be obtained at once using the name within square brackets after the group title (e.g. adlt.demo). Using this group statistic name all summary statistics of the thematic group marked with a star (*) will be computed.

Table 5.1: Summary statistics available for the demographic structure

Stat name	Description
Demography [(adlt/off).demo]	
(adlt/off).nbInd	total number of individuals in the metapopulation*
(adlt/off).nbFem	total number of females in the metapopulation*
(adlt/off).nbMal	total number of males in the metapopulation*
(adlt/off).meanInd	mean number of individuals per inhabited patch*
(adlt/off).meanFem	mean number of females per inhabited patch*
(adlt/off).meanMal	mean number of males per inhabited patch*
(adlt/off).sexRatio	sex ratio ($\frac{males}{females}$)*
(adlt/off).nbPops	number of inhabited patches*
(adlt/off).nbInd_p	number of individuals in patch i
(adlt/off).nbFem_p	number of females in patch i
(adlt/off).nbMal_p	number of males in patch i
Patch extinction [ext.rate]	
ext.rate	proportion of extinct patches in the metapopulation*
Fecundity [fecundity, available only for adults]	
fem.meanFec	mean realized female fecundity*
fem.varFec	mean variance of realized female fecundity*
mal.meanFec	mean realized male fecundity*
mal.varFec	mean variance of realized male fecundity*
Kinship [(adlt/off).kinship]	
(adlt/off).fsib	mean proportion of full-sib*
(adlt/off).phsib	mean proportion of paternal half-sib*
(adlt/off).mhsib	mean proportion of maternal half-sib*
(adlt/off).nsib	mean proportion of non-sib*
(adlt/off).self	mean proportion of selfed offspring*
Migration [migration, available only for adults]	
emigrants	mean number of emigrants per patch*
immigrants	mean number of immigrants per patch*
residents	mean number of residents per patch*
immigrate	mean effective immigration rate per patch* ($\frac{immigrants}{immigrants+residents}$)
colonisers	mean number of colonizers per extinct patch*
colon.rate	mean effective colonization rate of extinct patches*
Fitness [fitness, available only for adults]	
VwW	variance of the fitness of adults within patches*

Table 5.1 continued on next page

Stat name	Description
VwB	variance of the fitness of adults between patches*
meanW_p	mean fitness of adults in patch i (computed for each patch)
varW_p	variance of the fitness of adults in patch i (computed for each patch)

Table 5.1: Summary statistics available for the demographic structure continued

Chapter 6

Quantitative traits

quantiNEMO allows the simulation of multiple quantitative traits each having its own specifications. Each quantitative trait is defined by one to many loci each with up to 256 alleles. The allelic effects at each locus can be drawn from a normal distribution or can be set explicitly. Mutations are implemented with several models. The trait determinism can be purely additive, or include dominance and/or epistatic interactions among loci. Environmental effects can also be set in different ways:

$$P = G + E$$

$$G_{11'22'...ii'...} = \epsilon_{11'22'...ii'...} + \sum_{i=1}^{nbLoci} a_i + a_{i'} + k_{ii'}|a_{i'} - a_i|$$

Where P is the phenotype of the trait, G the genotypic value, and E the environmental contribution to the trait. $G_{11'22'...ii'...}$ is the genotypic value of genotype $11'22'...ii'...$ (1 and $1'$ are the alleles of locus 1, 2 and $2'$ are the alleles of locus 2, ...), a_i is the effect of the first allele of locus i, $a_{i'}$ is the effect of the second allele of locus i, $k_{ii'}$ is the dominance value between allele i and i', and $\epsilon_{11'22'...ii'...}$ is the epistatic value of genotype $11'22'...ii'...$.

Each quantitative trait can have its own selection pressure, which may be stabilizing or directional selection, or the trait can be neutral. The selection pressure may vary in time and space. A quantitative trait is at least specified by the number of loci and number of alleles. In this minimal definition the genotypic value of the quantitative trait is purely additive and the traits does not undergo any mutation. By default only additive genetic effects are simulated and each new population is initiated by assigning random allelic values within the range `[1, quanti_all]` to each locus thus assuring a very large initial variance.

6.1 Architecture

quanti_loci [integer]
This parameter specifies the number of loci defining the quantitative trait. This parameter is mandatory for the simulation of a quantitative trait.

quanti_all [1 to 256] (default: 255)
This parameter specifies the maximal number of alleles per locus (same number for each locus).

6.1.1 Allelic effects

Each allele has an allelic effect, its contribution to the genotype. There are two possibilities to define these effects. Either they are defined explicitly for each allele using a separate file (see parameter quanti_allelic_file), or they can be defined by specifying the variance of the normal distribution of the allelic effects (see parameter quanti_allelic_var). If the allelic effects are defined in both ways the explicitly defined effects are used. Note, that all effects have to be defined in the same way, i.e. explicitly or by their distribution.

quanti_allelic_file [string] (default: "")
This parameter allows to pass the name of a file containing allelic informations, such as the allelic effects, the mutation frequency, and/or the initial frequency. The information passed by this file has precedence over other settings, however the number of alleles and loci has to be in line with the parameters quanti_loci and quanti_all. The information can be set globally for all loci, if they have the same specifications, or for each locus separately. The allelic file has the following format:

```
#Allelic file
###############
[FILE_INFO]{
  col_locus 1
  col_allele 2
  col_allelic_value 3
  col_mut_freq 4
  col_ini_freq 5
}

#locus  allele  value   mut_freq  ini_freq
1       1       -1.0    0.2       {0  0.2}
1       2       -0.5    0.2       {0  0.2}
1       3       0.0     0.2       {1  0.2}
1       4       0.5     0.2       {0  0.2}
1       5       1.0     0.2       {0  0.2}
2       1       -2      0.2       {0  0.2}
```

```
2   2   −1   0.2   {0   0.2}
2   3   0    0.2   {1   0.2}
2   4   1    0.2   {0   0.2}
2   5   2    0.2   {0   0.2}
```

The file has to start with a file information box [FILE_INFO]{...}. This box contains informations about the structure of the following table and thus allowing flexibility in the format of the table. For example the order of the columns in the table may vary, or some columns may be ignored. The file information box starts with the key word [FILE_INFO]. This key word is followed by brackets "{...}" within which the user has to specify the contents of the columns to be considered by quantiNEMO. Each column is specified by a pair consisting of a key word (e.g. col_locus followed by the column number (the ordering starts with 1). Each column definition has to be on a new line. The following column keywords are available:

col_locus This keyword specifies the column containing the locus index. If this column is not declared in the file information box, quantiNEMO will use the same settings for all loci. In this case the number of lines of the table must be equal to (quanti_all). If the keyword col_locus is declared the number of rows of the table must be equal to the number of alleles times the number of loci (quanti_loci * quanti_all).

col_allele This keyword specifies the column containing the allele index. This column is mandatory. The index of the allele goes from 1 to quanti_all.

col_allelic_value This keyword specifies the column containing the allelic effects. If this column is not set the allelic effects will be drawn randomly from a normal distribution with the variance defined by the parameter quanti_all_effect_var.

col_mut_freq This keyword specifies the column containing the probability to mutate to this allele given that a mutation occurs.

col_ini_freq This keyword specifies the column containing the initial frequencies of the alleles. This column allows to explicitly set the allele frequencies at the start of a simulation. The frequencies can be set for each patch separately in which case the column consists in fact of one dimensional matrices, with each value of the matrix corresponding to one population. The lines of the matrix are enclosed in braces. In the example above, individuals of the first population are initially fixed for the allele 3 at the first locus as well as at the second locus. In the second population all alleles have the same initial frequency of 0.2. Note, that the matrix is adjusted in length if the number of populations does not correspond to the length of the matrix. If this column is not given the initial allele frequencies are set globally depending on the parameter quanti_ini_allele_model.

Note, that as in all input files for quantiNEMO it is possible to add comments (also in the file information box) using the hash character: '#' or '#/ ...any text... /#'.

quanti_allelic_var [decimal/matrix] (default: 1)
This parameter allows to specify the variance of the normal distribution from where the allelic effects are drawn randomly. The normal distribution is centered around 0 (mean phenotype is by definition 0). By using a matrix it is possible to define different variances of the normal distributions for each locus, i.e. specifying different contributions of the QTLs to the trait. This parameter is taken into account only if the allelic effects are not defined explicitly by the allelic file.

quanti_ini_allele_model [0,1] (default: 0)
This parameter allows to define the allele frequencies of the populations at the start of a simulation. Apart from these two models it is also possible to define the initial allele frequencies explicitly per population, locus and allele using the allelic file (parameter allelicFile, column keyword col_ini_freq). This parameter is ignored if the initial allele frequencies are defined by the allelic file.

0 : polymorph. The populations are maximally polymorph in respect to allele frequencies at the start of a simulation.

1 : monomorph. The populations are monomorph in respect to allele frequencies at the start of a simulation. All individuals are fixed for a single allele, which is the "middle" allele, i.e. the allele with the index $\lfloor \text{ntrl\_all}/2 \rfloor$.

6.1.2 Dominance effects

It is possible to simulate dominance effects between alleles at a locus. The following equation is used to compute the genotypic value at a given locus:

$$G_i = a_i + a_{i'} + k_{ii'}|a_{i'} - a_i|$$

Where G_i is the genotypic value of locus i, a_i and $a_{i'}$ are the effects of the two alleles at locus i, and $k_{ii'}$ is the dominance value between allele i and i'. k can have the following effects:

- $k < -1$: underdominance
- $k = -1$: allele a_i is dominant

- $k = 0$: no dominance: purely additive
- $k = 1$: allele $a_{i'}$ is dominant
- $k > 1$: overdominance

There are two possibilities to define these dominance effects. Either they are defined explicitly for each pair of alleles using a separate file (see parameter quanti_dominance_file), or by defining the normal distribution from where the dominance effect are randomly drawn (see parameters quanti_dominance_mean and quanti_dominance_var). If the dominance effects are defined in both ways the explicitly defined effects are used. Note, that all effects have to be defined in the same way, i.e. explicitly or by their distribution. By default, the parameters quanti_dominance_mean and quanti_dominance_var are set to zero resulting in a purely additive genotypic value of a locus.

quanti_dominance_file [string] (default: "")

This parameter allows to pass the name of a file containing dominance effects. The number of alleles and loci has to be in line with the parameters quanti_loci and quanti_all. The information can be set globally for all loci, if they have the same specifications, or for each locus separately. The dominance file has a similar format as the allelic file:

```
#Dominance file
#####################
[FILE_INFO]{
  col_locus          1
  col_allele1        2
  col_allele2        3
  col_dominance      4
}

#locus  allele1  allele2  value
1       1        2        0.298
1       1        3        0.435
1       1        4        0.224
1       2        3        0.104
1       2        4        0.974
1       3        4        0.808
```

The file has to start with a file information box [FILE_INFO]{...}. This box contains informations about the structure of the following table and thus allowing flexibility in the format of the table. For example the order of the columns in the table may vary, or some columns may be ignored. The file information box starts with the key word [FILE_INFO]. This key word is followed by brackets "{...}" within which the user has to specify the contents of the columns to be considered by quantiNEMO. Each column is specified by a pair consisting of

a key word (e.g. col_locus followed by the column number (the ordering starts with 1). Each column definition has to be on a new line. The following column keywords are available:

col_locus This keyword specifies the column containing the locus index. If this column is not declared in the file information box, quantiNEMO will use the same settings (dominance effect and/or fitness factor) for all loci.

col_allele1 This keyword specifies the column containing the index of the allele with the smaller allelic effect. This column is mandatory. The index of the allele goes from 1 to quanti_all.

col_allele2 This keyword specifies the column containing the index of the allele with the larger allelic effect. This column is mandatory. The index of the allele goes from 1 to quanti_all.

col_dominance This keyword specifies the column containing the dominance effects. For any pair of alleles for which the dominance effect is not specified a dominance effect will be drawn from the normal distribution defined by the parameters quanti_dominance_mean and quanti_dominance_var.

Note, that as in all input files for quantiNEMO it is possible to add comments (also in the file information box) using the hash character: '#' or '#/ ...any text... /#'.

quanti_dominance_mean [decimal] (default: 0)
This parameter allows to specify the mean of the normal distribution from where the dominance effects are randomly drawn. Note that a_1 has the smaller allelic effect than a_2. This parameter is only taken into account if the dominance effects are not defined explicitly by the dominance file.

quanti_dominance_var [decimal] (default: 0)
This parameter allows to specify the variance of the normal distribution from where the dominance effects are randomly draw. This parameter is taken into account only if the dominance effects are not defined explicitly by the dominance file.

6.1.3 Epistatic effects

It is possible to simulate epistatic effects between alleles at different loci. The following equation is used to compute the genotype:

$$G_{11'22'...ii'...} = \epsilon_{11'22'...ii'...} + \sum_{i=1}^{nbLoci} G_i$$

Where $G_{11'22'...ii'...}$ is the genotypic value of genotype $11'22'...ii'...$ (1 and $1'$ are the alleles of locus 1, 2 and $2'$ the alleles of locus 2, ...), G_i is the genotypic effect of locus i (additive and dominance effects), and $\epsilon_{11'22'...ii'...}$ is the epistatic effect of genotype $11'22'...ii'...$ (unique for each multilocus genotype). There are two possibilities to define these epistatic effects. Either they are defined explicitly for each genotype using a separate file (see parameter quanti_epistatic_file), or by defining the variance of the normal distribution from where the epistatic effects are randomly drawn (see parameter quanti_epistatic_var). Using the parameter quanti_epistatic_file it is also possible to define the genotypic effects directly. If the epistatic effects are defined in both ways the explicitly defined effects are used. Note, that all effects have to be defined in the same way, i.e. explicitly or by their distribution. By default, the parameter quanti_epistatic_var is set to zero resulting in simulations without any epistatic effects.

quanti_epistatic_file [string] (default: "")

This parameter allows to pass the name of a file containing the epistatic effects. The number of defined genotypes has to be in line with the parameters quanti_loci and quanti_all. The epistatic file has a similar format as the allelic file:

```
#Epistatic file
####################
[FILE_INFO]{
  col_genotype 1
  col_epistatic_value 2
  # col_genotypic_value 3
}

#genotype        epistaticVal  genotypicVal
{0101 0101}      0.242984      1.87009
{0101 0102}      0.580787      0.834811
...
{5050 5048}      0.264001      1.24981
{5050 5049}      0.118982      -0.55701
{5050 5050}      0.071359      -2.26644
```

The file has to start with a file information box [FILE_INFO]{...}. This box contains informations about the structure of the following table and thus allowing flexibility in the format of the table. For example the order of the columns in the table may vary, or some columns may be ignored. The file information box starts with the key word [FILE_INFO]. This key word is followed by brackets "{...}" within which the user has to specify the contents of the columns to be considered by quantiNEMO. Each column is specified by a pair consisting of a key word (e.g. col_locus followed by the column number (the ordering starts with 1). Each column definition has to be on a new line. The following column keywords are available:

col_genotype This keyword specifies the column containing the genotype. This column is mandatory. The genotype is enclosed by brackets "{...}" and the genotype itself has to be in the FSTAT format (Goudet, 1995). The two alleles of a locus are written consecutively without any space. For all alleles the same number of digits are needed. Two different loci are separated by a space. The above example consists of two loci, each with 50 alleles. Each allele is written using two digits. The table must include all possible genotypes, consequently the table has $(\_nb\_all * (\_nb\_all + 1)/2)^{\_nb\_loci}$ rows, which can be a very large number if there are many loci.

col_epistatic_value This keyword specifies the column containing the epistatic effects. If this column is set the epistatic effect is added to the genotypic effect computed as described above.

col_genotypic_value This keyword specifies the column containing directly the genotypic effects. If this column is set, the genotypic effect of each genotype is set directly without taking into account allelic, dominance, and/or epistatic effects. If both keywords col_epistatic_value and col_genotypic_value are specified in the information box, only the column col_genotypic_value is considered.

Note, that as in all input files for quantiNEMO it is possible to add comments (also in the file information box) using the hash character: '#' or '#/ ...any text... /#'.

quanti_epistatic_var [decimal] (default: 0)
This parameter allows to specify the variance of the normal distribution from where the epistatic effects are drawn randomly. The normal distribution is centered around 0. This parameter is only taken into account if the epistatic effects are not defined explicitly by the epistatic file.

6.1.4 Environment

The environment may also contribute to the phenotype. There are several possibilities to set the contribution of the environment to the phenotype globally or for each patch separately. Either the contribution of the environment to the phenotype is defined directly by the variance of the environmental effect (model 0), by the narrow-sense heritability h^2 (model 1 and 2), or by the broad-sense heritability H^2 (model 3 and 4). The heritability is later translated into a corresponding environmental variance V_E:

$$h^2: \qquad V_E = \frac{1-h^2}{h^2} * V_A$$

$$H^2 : \qquad V_E = \frac{1 - H^2}{H^2} * V_G$$

Where V_A is the additive genetic variance computed following (Lynch and Walsh, 1998, p85-87), and V_G the genetic variance.

quanti_environmental_model [0,1,2,3,4] (default: 0)

This parameter specifies how the environmental variance is defined. The following models are available:

0 : set V_E directly. The variance of the environment is set directly by the parameter quanti_heritability, which is in this case not the heritability, but the environmental variance.

1 : V_E defined by the narrow-sense heritability (V_E constant). The variance of the environment (V_E) is set at the beginning of a simulation (generation 1) and is based on the narrow-sense heritability (h^2, parameter quanti_heritability) and the additive genetic variance (V_A at generation 1). Note, that in this case the environmental variance remains constant over time, but not the heritability.

2 : V_E defined by the narrow-sense heritability (h^2 constant). This is the same as model 1, but the environmental variance is readjusted at each generation. Thus the narrow-sense heritability remains constant over time, but not the environmental variance.

3 : V_E defined by the broad-sense heritability (V_E constant). The variance of the environment (V_E) is set at the beginning of a simulation (generation 1) and is based on the broad-sense heritability (H^2, parameter quanti_heritability) and the genetic variance (V_G at generation 1). Note, that in this case the environmental variance remains constant over time, but not the heritability.

4 : V_E defined by the broad-sense heritability (H^2 constant). This is the same as model 3, but the environmental variance is readjusted at each generation. Thus the broad-sense heritability remains constant over time, but not the environmental variance.

quanti_heritability [decimal/matrix] (default: 1)

This parameter depends on the environmental model chosen (parameter quanti_environmental_model): If V_E is directly set (model 0) this parameter is the environmental variance. For the environmental model 1 and 2 this parameter is the narrow-sense heritability (h^2):

$$h^2 = V_A/V_P$$

For the environmental model 3 and 4 this parameter is the broad-sense heritability (H^2):

$$H^2 = V_G/V_P$$

Note, that a heritability (narrow and broad sense) of 0 (no genetic component) makes no sense for models 1 to 4 in this simulation framework. Therefore a value of 0 is not accepted by quantiNEMO for models 1 to 4, resulting in an error message. If the parameter quanti_environmental_model is set to 0 the parameter quanti_heritability is no longer the heritability, but the environmental variance directly, and can be set to 0. Using a matrix as argument it is possible to set the environmental variance, respectively heritability for each patch separately.

quanti_environmental_proportion [decimal] (default: 1)
This parameter specifies which environment affects the phenotype of the quantitative trait: the natal or the current (at the adult stage) environment? The argument specifies the relative weight of the current patch effect on the phenotype: if the value is 1 (default value) only the environmental variance of the current patch affects the phenotype while if the value is 0 only the environmental variance of the natal patch affects the phenotype.

quanti_va_model [0,1] (default: 0)
This parameter specifies how the additive genetic variance (V_A) of a quantitative trait is computed. If purely additive effects are simulated (no dominance end epistatic effects) the additive genetic variance is identical to the genetic variance (V_G) and thus this parameter is not considered. The additive genetic variance of a trait is used in several statistics. For instance it is used to set the environmental variance of the patches if this variance is specified by the narrow-sense heritability ($h2$, parameter quanti_environmental_model set to 2 or 3, see also parameter quanti_heritability), the additive genetic variance may also be directly output as summary statistic (stat option $q.varA$), or indirectly in the population differentiation measurement Q_{ST} (stat option $q.qst$). The additive genetic variance (V_A) is computed following Lynch and Walsh (1998, p85-87). For a quantitative trait determined by a single locus this is:

$$V_A = 2 \sum_{i=1}^{nbAllele} p_i \alpha_i \alpha_i^*$$

Where p_i is the allele frequency of allele i, α_i is the additive effect of allele i, and α_i^* is the average excess of allele i. Note, that we show here the computation for a single locus for simplicity reasons. In quantiNEMO a full version for traits determined by multiple loci is implemented. However, the implementation does not take into account linkage between loci, thus the additive genetic variance is underestimated when loci are linked.

0 : for any case. If this model is chosen the additive effect (α_i) and the average excess (α_i^*) are computed. While the average excess is simple to

compute, the computation of the additive effect requires a time consuming least-square regression. This formula is valid for any mating system, but its computation is rather slow.

Note, that the least-square regression has not always a solution thus the additive effects α, are not always computable. In this case the locus in question is skipped from the analysis and a warning is returned:

```
***WARNING*** Va could not be correctly estimated (1. time
 at generation 20, see manual parameter 'quanti_va_model')
```

This warning is returned the first 10 times the problem occurs and then every hundredth time. It is up to the user to decide whether the problem occurs to often or if these "missing points" are acceptable. If the environmental variance is set by the narrow-sense heritability (parameter quanti_environmental_model set to 1 or 2) and the additive genetic variance cannot be computed quantiNEMO uses the additive genetic variance computed using the algorithm for random matings (this parameter set to 1).

1 : limited to random mating. In case of random mating the additive effect (α_i) and the average excess (α_i^*) are identical. Therefore the time consuming computation of the additive effect can be omitted. Note, that even when the mating system is set to random mating (parameter mating_system set to 0 or 2) mating is not randomly if selection acts, as the choice of the parents depends on their fitnesses. Its up to the user to decide whether this quick way to compute the additive genetic variance is appropriate or not.

6.2 Mutation

Mutation rates may be defined for each locus individually by explicitly defining the individual mutation rates (parameter quanti_mutation_rate) or by defining the gamma distribution from which the individual mutation rates are drawn (parameters quanti_mutation_rate and quanti_mutation_var). Depending on the mutation model (parameter quanti_mutation_model) the mutant effect is the effect of the drawn allele (model 0), or the effect of the drawn allele is added to the current allelic effect to get the mutant effect (model 1). Using the allelic file (see section 6.1.1) it is possible to specify for each allele its effect and the probability to mutate to this allele given that there is a mutation. A minimal definition for mutations requires the setting of a common mutation rate (parameter quanti_mutation_rate has a single value). In this case all loci have the same mutation rate and the mutation model is RMM.

quanti_mutation_model [0,1] (default: 0)

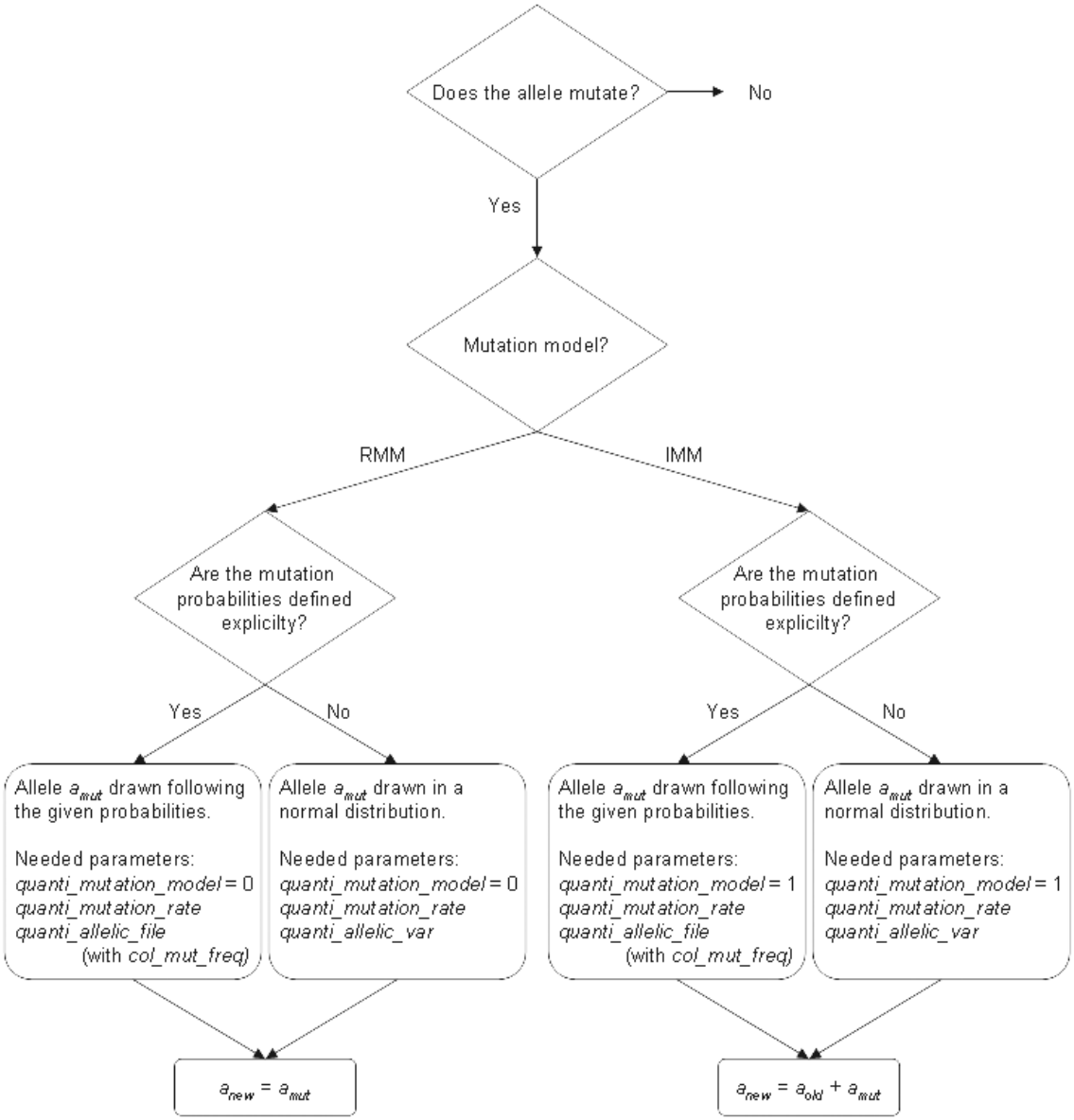


Figure 6.1: Schematic representation of the mutation process for a specific allele

This parameter allows to specify the mutation model. The following mutation models are available:

0 : RMM (Random Mutation Model)
At a given mutation a new allele is drawn randomly.

$$a_{new} = a_{mut}$$

Where a_{new} is the effect of the new allele, and a_{mut} is the effect of the drawn allele. The probability to mutate to a certain allele depends on its probability to mutate to it given that there is a mutation. These probabilities and also the effects of the alleles can explicitly be set by the allelic file (see section 6.1.1). If this is not the case quantiNEMO allocates the allelic effects and the probabilities to mutate to the alleles given that there is a mutation automatically: The effects of the alleles are regularly spaced between -6σ and 6σ, if there are more than 5 alleles, and the range is down regulated if the number of alleles is lower. Note, that in this later case the variance of the allelic effects in a population may not meet the specified variance. Where σ is the square root of the variance of the normal distribution from where the allelic effects are drawn randomly (see parameter quanti_allelic_var). The probability to mutate to a given allele given that there is a mutation is identical to the frequency of this allele defined by the distribution of the allelic effects (see parameter quanti_allelic_var), i.e. follow a normal distribution.

1 : IMM (Increment Mutation Model)
At a given mutation an allele effect is drawn randomly and is added to the current allelic effect.

$$a_{new} = a_{old} + a_{mut}$$

Where a_{new} is the effect of the new allele, a_{mut} is the effect of the drawn allele, and a_{old} is the effect of the old allele, before the mutation. The probability to mutate to a certain allelic effect depends on its probability to mutate to it given that there is a mutation. These probabilities and also the effects of the alleles can explicitly be set by the allelic file (see section 6.1.1). If this is not the case quantiNEMO allocates the allelic effects and the probabilities to mutate to the alleles given that there is a mutation automatically: The effects of the alleles are regularly spaced between -20σ and 20σ, where σ is the square root of the variance of the normal distribution from where the allelic effects are drawn randomly (see parameter quanti_allelic_var). The probability to mutate to a given allelic effect given that there is a mutation is identical to its allele frequency, i.e. follow a normal distribution. This mutation model requires that the allelic effects are regularly spaced around zero, that the number of alleles

is odd (thus the allele with index $\lfloor$quanti_all$/2\rfloor$ is zero), and that there are at least 51 possible alleles (parameter quanti_all).

Note, that for both mutation models the number of alleles have to be odd if the allelic effects and the probabilities to mutate to the alleles given that there is a mutation are set automatically. In this case also a warning will be drawn if the number of alleles is below 200, informing that this number of alleles may not well represent the normal distribution of the allelic effects.

quanti_mutation_rate [decimal/matrix] (default: 0)
This parameter specifies the mutation rate per locus and generation. If the argument is a single value the mutation rate for all loci is the same. By passing a matrix of mutation rates it is possible to set the mutation rate for each single locus individually. By default no mutations occur.

quanti_mutation_shape [integer] (default: 0)
This parameter allows to specify the shape of the gamma distribution from which the mutation rates for each locus are drawn. The mean mutation rate is given by the parameter quanti_mutation_rate:

$$\Gamma(shape, mut\_rate/shape)$$

Where Γ is the gamma function requiring two parameters. The first parameter defines the shape of the gamma distribution (parameter quanti_mutation_shape), and the second parameter the scale to the gamma distribution. The scale parameter is set such does the mean of the gamma distribution is equal to the mean mutation rate (mut_rate, respectively parameter quanti_mutation_rate). If the shape is 0 (default value) all loci have the same mutation rate defined by the parameter quanti_mutation_rate. The parameter quanti_mutation_shape is ignored if the mutation rates (parameter quanti_mutation_rate) are explicitly defined for each locus by a matrix.

6.3 Multiple traits

quantiNEMO allows to simulate several quantitative traits simultaneously. Each trait has its own architecture and may be under different selection pressures.

quanti_nb_trait [integer] (default: 1)
This parameter defines the number of quantitative traits.

Each trait may have its own specifications, but it is also possible to specify parameters for some quantitative traits together. If several quantitative traits are used it is

possible to address a certain trait by its number. For instance to specify a parameter for the fifth trait one has to append a "_5" to the parameter name. In contrast if for the fifth trait no parameter with the suffix "_5" is passed quantiNEMO checks if the parameter is passed for the fourth trait (suffix "_4"). If this is also not the case quantiNEMO checks if the parameter is passed for the third trait (suffix "_3"), and so forth until a parameter is found. Note, that a parameter without a suffix is the same as the parameter with the suffix "_1". This behavior of quantiNEMO allows to specify parameters for a group of traits. An example may make it clearer:

```
quanti_nb_trait        12

quanti_loci            5
quanti_loci_7          10

quanti_all_1           10
quanti_all_4           20
quanti_all_7           10
quanti_all_10          20

quanti_mutation_rate   0.0001
```

In this example we simulate 12 quantitative traits. Traits 1 to 3 consist of 5 loci with up to 10 alleles, traits 4 to 6 consist of 5 loci with up to 20 alleles, traits 7 to 9 consist of 10 loci with up to 10 alleles, and traits 10 to 12 consist of 10 loci with up to 20 alleles. All traits have the same mutation rate of 0.0001.

6.4 Selection models

In the current version, only selection at the reproductive stage is implemented. However, it is possible to modify the code in a way that selection may also act on other life cycles, such as during migration or regulation. The fitness of an individual defines its reproductive success. The higher its fitness, the more offspring are produced. The fitness itself depends on the selection pressure and on the phenotype of the individual. The better an individual is adapted to the local environment the higher its fitness is. The phenotype of an individual is characterized by one to several quantitative traits (see section 6.3). Each quantitative trait has its individual architecture, and may be selected for a different selection pressure. Selection types may differ between quantitative traits. The selection pressure has to be defined in the patch component (see section 5). If several quantitative traits are simulated the fitness of an individual is the product of the quantitative trait's fitnesses:

$$W = \prod_{t}^{traits} W_t$$

Where W is the total fitness of the individual, and W_t is the fitness of quantitative trait t.

quanti_selection_model [0,1,2] (default: 0)
This parameter allows to define the selection model for each quantitative trait. Section 5 describes how to set the selection pressure at the patch level.

0 : stabilizing selection
Stabilizing selection acts on the quantitative trait. The fitness W of quantitative trait t is computed using the following standard Gaussian function for stabilizing selection:

$$W_t = e^{\left(-\frac{\left(P-Z_{Opt}\right)^2}{2\omega^2}\right)}$$

Z_{Opt} is the selection optimum of the current trait and patch (parameter patch_stab_sel_optima), P is the phenotypic value of the individual, and ω is the intensity of the selection (parameter patch_stab_sel_intensity).

1 : directional selection.
Directional selection acts on the quantitative trait. A generalized logistic curve (Richards, 1959) is implemented in quantiNEMO to characterize the directional selection pressure:

$$W_t = \left(1 + s * e^{r(P_{r_{Max}}-P)}\right)^{-1/s}$$

Where r is the growth rate (parameter patch_dir_sel_growth_rate), $P_{r_{Max}}$ is the phenotype with the maximal slope (parameter patch_dir_sel_max_growth), and s defines the symmetry of the slope (parameter patch_dir_sel_symmetry; slope is symmetric by default (value 0.5)).

2 : neutral selection.
The quantitative trait is not under selection. The fitness of this quantitative trait is set to 1 not taking into account the phenotype:

$$W_t = 1$$

6.5 Genetic map

quantiNEMO has an underlying genetic map, which may consist of several chromosomes. This allows to explicitly position all types of loci on the map (quantitative trait loci (QTL) and neutral markers). The unit of the genetic map is centi Morgans (Haldane, 1919, cM). If the genetic map of a trait is not specified by either of the following parameters, the loci for that trait are assumed to be unlinked and independent.

quanti_loci_positions [matrix] (default: "")

For each trait this parameter allows to specify explicitly the positions of the loci in centi Morgans from the beginning of the chromosome. The brackets separate the chromosomes. Within a chromosome, the positions of the different loci are separated by a space. Note, that chromosomes may contain different numbers of loci, but that the total number of loci must correspond to the parameter quanti_loci. It is possible to skip chromosomes if they don't contain loci (see section 2.3.5.4):

```
quanti_loci 11
quanti_loci_positions { {1: 10}
                        {3: 10 40 60 80 100}
                        {   10 40 60 80 100} }
```

In the example above, the quantitative trait is defined by 11 loci located on three of at least four chromosomes. The first chromosome contains a single locus at position 10 cM. No loci are located on the second chromosome. The third and fourth chromosomes have 5 loci at positions 10, 40, 60, 80, and 100 cM.

quanti_loci_positions_random [matrix] (default: "")

This parameter allows to specify the number of chromosomes and their lengths. The loci are then randomly positioned on the chromosomes. The matrix contains the length of each chromosome in cM. It is possible to skip chromosomes if they don't contain loci (see section 2.3.5.4):

```
quanti_loci 11
quanti_locus_positions_random { {1: 100} {3: 100} {100} }
```

In this example the quantitative trait is defined by 11 loci located on three out of four chromosomes. The loci of the quantitative trait are randomly positioned on the first 100 cMs of each chromosome. No loci of this quantitative trait are located on chromosome 2.

For every simulated replicate where a genetic map is simulated the entire genetic map (including QTLs and neutral loci) is dumped to a file named "genetic_map.txt" stored in the simulation folder. If multiple replicates are simulated a replicate counter (example _r3) is inserted before the extension. Using the parameter **genetic_map_output** it is possible to suppress this output.

6.6 Output

Apart from the genotypes and the phenotypes it is also possible to obtain certain summary statistics, the allelic, the dominance, and the epistatic effects. For each

type of output it is possible to define individually the time interval at which the output is generated, which may also vary over time.

6.6.1 Genotype

The genotype of all individuals may periodically be dumped to files. The output files will be stored in the folder given by the parameter quanti_genot_dir and will have the name of the base file name (see parameter filename in section 4). The extension is ".dat". A counter for the generation (e.g. _g05) and the replicate (e.g. _r4) is inserted before the extension. An example of such a file name is "simulation_g05_r4.dat".

quanti_save_genotype [0,1,2] (default: 0)

This parameter specifies the output of the quantitative genotype at the QTLs.

0 : None. No output is generated.

1 : FSTAT. The output contains the genotypes in the FSTAT format (Goudet, 1995).

2 : FSTAT extended. Same as point 1, but the file contain the following six additional columns: the age class (1 = offspring, 2 = adult), the sex (0 = male; 1 = female), the ID of the individual, the ID of the mother, the ID of the father, and the fitness of the individual. The ID is a unique identifier for each individual of a simulation in the format "345_23", meaning that this is the 345th individual born in patch 23. The IDs of the individual, the mother and the father allow to extract pedigree informations, if the output is stored for each generation, and also to investigate the migration behavior of the individual and its parents.

An example of such a file (with quanti_save_genotype set to 2):

```
5 4 20 2
loc-1
loc-2
loc-3
loc-4
1 1415 1019 2002 0820 1 1 10_1 1_1 0_1 0.345
1 0814 0219 2002 2020 1 1 11_1 8_1 2_4 0.334
1 0808 0217 1902 0820 1 1 12_1 5_3 5_1 0.123
...
5 1004 0917 1404 1007 1 1 16_5 9_5 3_2 0.999
5 2017 1010 2013 1812 1 0 17_5 3_2 9_2 1.000
5 2017 1008 2013 1811 1 1 11_4 8_2 9_2 0.678
```

The first line contains the number of patches (5 patches here), the number of loci (4), the highest possible allele index (20), and the number of digits used to write each allele (2). The next four lines contain the locus names. The

following lines contain the individual's info, one individual per line. The first number is the patch number of the individual, followed by the genotype. Each column represents a locus, and the first half of the locus (first 2 digits) represents the first allele index, while the second half of the locus (last 2 digits) the second allele at the given locus. As, in this example, we are using two digits per allele, the first two digits of a locus genotype number are the first allele (e.g. allele 14 for the first allele of the first locus of the first individual) while the two next digits are the second allele (e.g. allele 15 for the second allele of the first locus of the first individual). Each line ends with six columns consisting supplementary information on the individual (see above) if the parameter quanti_save_genotype is set to 2.

quanti_genot_dir [string] (default: "")
This parameter allows to specify the subdirectory where the genotypes are stored. This directory has to be specified relative to the simulation folder (parameter folder), and may also contain subdirectories. If not specified (default) the output is stored in the simulation folder (parameter folder).

quanti_genot_logtime [integer] (temporal/default: 1)
This parameter specifies the time interval of the genotype output. Since the parameter may change over time the output may be generated at any generation.

quanti_genot_script [string] (default: "")
It is possible to launch a script just after the genotype file is generated. The argument of the parameter is the file name of the script. The name of the genotype file is passed as unique parameter to the script.

quanti_genot_sex [0,1,2] (default: 0)
This parameter allows to choose which sex is output.

0 : Both. Output includes both sexes.

1 : Females. Output includes only female genotypes.

2 : Males. Output includes only male genotypes.

quanti_genot_age [0,1,2] (default: 0)
This parameter allows to choose which age is output.

0 : Adults. Output includes only adult genotypes.

1 : Juveniles. Output includes only juvenile genotypes.

2 : Both. Output includes juveniles and adults genotypes.

6.6.2 Genotypic value

Similar to the genotypes the genotypic values may be periodically dumped to files. The output files will be stored in the folder given by the parameter quanti_geno_value_dir and will have the name of the base file name (see parameter filename in section 4). The extension is ".gen". A counter for the generation (e.g. _g05) and the replicate (e.g. _r4) is inserted before the extension. An example of such a file name is "simulation_g05r4.gen".

quanti_save_geno_value [0,1,2] (default: 0)

This parameter specifies the output of the phenotype.

0 : None. No output is generated.

1 : FSTAT. The output contains the phenotypes in the FSTAT-like format (Goudet, 1995).

2 : FSTAT extended. Same as point 1, but the file contain the following six additional columns: the age class (1 = offspring, 2 = adult), the sex (0 = male; 1 = female), the ID of the individual, the ID of the mother, the ID of the father, and the fitness of the individual. The ID is a unique identifier for each individual of a simulation in the format "345_23", meaning that this is the 345th individual born in patch 23. The IDs of the individual, the mother and the father allow to extract pedigree informations, if the output is stored for each generation, and also to investigate the migration behavior of the individual and its parents.

An example of such a file (quanti_save_geno_value is set to 2):

```
2 5
genotypic_value_trait_1
genotypic_value_trait_2
genotypic_value_trait_3
genotypic_value_trait_4
genotypic_value_trait_5
1 0.0493 -3.203 -2.441 0.0683 -3.199 2 1 10_1 1_1 0_1 0.345
1 0.4924 -3.803 -0.869 -2.002 -2.594 2 1 11_1 8_1 2_2 0.334
1 2.2342 -2.931 -0.725 -0.750 -0.698 2 1 12_1 5_2 5_1 0.123
...
2 0.8623 0.6525 -0.857 1.7483 -4.194 2 1 16_2 9_2 3_2 0.999
2 1.7752 -2.223 -3.117 0.3409 -2.003 2 1 17_2 3_2 9_2 1.000
2 0.2081 -2.803 -0.146 -0.456 -5.137 2 1 11_1 8_1 9_1 0.678
```

The first line contains the number of patches (2 patches here), and the number of traits (5). The next five lines contain the five trait names. The following lines contain the individual's info, one individual per line. The first number is the patch number of the individual, followed by the genotypic value for each trait. Each line ends with six columns consisting supplementary information

on the individual (see above) if the parameter quanti_save_geno_value is set to 2.

quanti_geno_value_dir [string] (default: "")
This parameter allows to specify the subdirectory where genotypic values are stored. This directory has to be specified relative to the simulation folder (parameter folder), and may also contain subdirectories. If not specified (default) the output is stored in the simulation folder (parameter folder).

quanti_geno_value_logtime [integer] (temporal/default: 1)
This parameter specifies the time interval of the genotypic value output. Since the parameter may change over time the output may be generated at any generation.

quanti_geno_value_script [string] (default: "")
It is possible to launch a script just after the genotypic value file is generated. The argument of the parameter is the file name of the script. The name of the genotypic value file is passed as unique parameter to the script.

quanti_geno_value_sex [0,1,2] (default: 0)
This parameter allows to choose which sex is output.

0 : Both. Output includes both sexes.

1 : Females. Output includes only female genotypic values.

2 : Males. Output includes only male genotypic values.

quanti_geno_value_age [0,1,2] (default: 0)
This parameter allows to choose which age is output.

0 : Adults. Output includes only adult genotypic values.

1 : Juveniles. Output includes only juvenile genotypic values.

2 : Both. Output includes juveniles and adults genotypic values.

6.6.3 Phenotypic value

Similar to the genotypes the phenotypic values of the adults may be periodically dumped to files. The phenotype of juveniles cannot be output, as the phenotype is only computed when selection acts, and this is at the reproduction stage, i.e. when individuals are adults. The output files will be stored in the folder given by the parameter quanti_phenot_dir and will have the name of the base file name (see parameter filename in section 4). The extension is ".phe". A counter for the generation (e.g. _g05) and the replicate (e.g. _r4) is inserted before the extension. An example of such a file name is "simulation_g05r4.phe".

quanti_save_phenotype [0,1,2] (default: 0)

This parameter specifies the output of the phenotype.

0 : None. No output is generated.

1 : FSTAT. The output contains the phenotypes in the FSTAT-like format (Goudet, 1995).

2 : FSTAT extended. Same as point 1, but the file contain the following six additional columns: the age class (1 = offspring, 2 = adult), the sex (0 = male; 1 = female), the ID of the individual, the ID of the mother, the ID of the father, and the fitness of the individual. The ID is a unique identifier for each individual of a simulation in the format "345_23", meaning that this is the 345th individual born in patch 23. The IDs of the individual, the mother and the father allow to extract pedigree informations, if the output is stored for each generation, and also to investigate the migration behavior of the individual and its parents.

An example of such a file (quanti_save_phenotype is set to 2):

```
2 5
phenotypic_value_trait_1
phenotypic_value_trait_2
phenotypic_value_trait_3
phenotypic_value_trait_4
phenotypic_value_trait_5
1 0.0493 -3.203 -2.441 0.0683 -3.199 2 1 10_1 1_1 0_1 0.345
1 0.4924 -3.803 -0.869 -2.002 -2.594 2 1 11_1 8_1 2_2 0.334
1 2.2342 -2.931 -0.725 -0.750 -0.698 2 1 12_1 5_2 5_1 0.123
...
2 0.8623 0.6525 -0.857 1.7483 -4.194 2 1 16_2 9_2 3_2 0.999
2 1.7752 -2.223 -3.117 0.3409 -2.003 2 1 17_2 3_2 9_2 1.000
2 0.2081 -2.803 -0.146 -0.456 -5.137 2 1 11_1 8_1 9_1 0.678
```

The first line contains the number of patches (2 patches here), and the number of traits (5). The next five lines contain the five trait names. The following lines contain the individual's info, one individual per line. The first number is the patch number of the individual, followed by the phenotype value for each trait. Each line ends with six columns consisting supplementary information on the individual (see above) if the parameter quanti_save_phenotype is set to 2.

quanti_phenot_dir [string] (default: "")

This parameter allows to specify the subdirectory where phenotypes are stored. This directory has to be specified relative to the simulation folder (parameter folder), and may also contain subdirectories. If not specified (default) the output is stored in the simulation folder (parameter folder).

quanti_phenot_logtime [integer] (temporal/default: 1)
This parameter specifies the time interval of the phenotype output. Since the parameter may change over time the output may be generated at any generation.

quanti_phenot_script [string] (default: "")
It is possible to launch a script just after the phenotype file is generated. The argument of the parameter is the file name of the script. The name of the phenotype file is passed as unique parameter to the script.

quanti_phenot_sex [0,1,2] (default: 0)
This parameter allows to choose which sex is output.

0 : Both. Output includes both sexes.

1 : Females. Output includes only female phenotypes.

2 : Males. Output includes only male phenotypes.

quanti_phenot_age [0,1,2] (default: 0)
This parameter allows to choose which age is output. Phenotypes are only computed at the reproduction stage, thus juveniles have no phenotypes, respectively are indicated as NaN (not a number) in the output. This parameter is only present for uniformity reasons to obtain the same output format.

0 : Adults. Output includes only adult phenotypes.

1 : Juveniles. Output includes only juvenile phenotypes, respectively NaNs.

2 : Both. Output includes juveniles and adults phenotypes.

6.6.4 Architecture

Depending on the definition of the architecture of the quantitative trait it is possible to obtain the allelic file, the dominance file, and/or the epistatic file. All three files follow the structure of the homonymous input files, except that for the input all possible combinations have to be present, while in the output only the used combinations are output. Therefore the output files of allelic, dominance, and epistatic cannot always be used as input files. The files have the names "allelic_values.txt", "dominance_values.txt, and "epistatic_values.txt" and are stored in the simulation folder (parameter folder). If several replicates are simulated the files are generated for each replicate and a replicate counter (e.g. _r4) is inserted before the extension.

quanti_output [0,1] (default: 0)

0 : None. The files are not generated.

1 : Output. The allelic, the dominance, and/or the epistatic files are stored in the simulation folder (parameter folder), if they were used during the simulation.

6.7 Summary statistics

The summary statistics listed in the table below are available for quantitative traits. The column Stat name contains the name of the summary statistic used to specify which summary statistics are computed (parameter stat, for details see section 3.2). These names appear also in the output file. The column Description contains a short description of the summary statistic.

Some of the summary statistics are available for adults and offspring (indicated by (adlt/off)). To obtain a certain summary statistic for adults the prefix adlt. has to be added to the summary statistic name (e.g. adlt.allnb), respectively the prefix off. to obtain the summary statistic for offspring (e.g. off.allnb).

Some of the summary statistics are computed for each patch separately (indicated by *(computed for each patch)* in the table), others for each pair of patches (indicated by *(all pairwise combinations computed)*). Depending on the number of patches this may lead to the computation of a large number of summary statistics. The names of such summary statistics are marked in the output file with the index of the patch: "_pX", respectively "_pX-Y" for pairwise summary statistics, where X and Y are the index of the patches (starting with 1).

The summary statistics are computed for every quantitative trait. If several quantitative traits are simulated the postfix "_tT" is added to the summary statistic name in the output file, where T is the index of the trait.

The summary statistic name (column Stat name) may be used to specify the summary statistic to be computed (e.g. q.varA). Similar summary statistics (within a thematic group) may be obtained at once using the name within square brackets after the group title (e.g. quanti). Using this group statistic name all summary statistics of the thematic group marked with a star (*) will be computed.

Table 6.1: Summary statistics available for quantitative traits

Stat name	Description
Quantitative trait statistics	[quanti, available only for adults]
q.VgW	genetic variance within patches*
q.VgB	genetic variance between patches*
q.VpW	phenotypic variance within patches*

Table 6.1 continued on next page

Stat name	Description
q.VpB	phenotypic variance between patches*
q.VaW	additive genetic variance within patches
q.qst	Q_{ST}
q.varA_p	additive genetic variance of patch i following Lynch and Walsh (1998, p85-87) (computed for each patch)
q.meanG_p	genetic mean of patch i (computed for each patch)
q.varG_p	genetic variance of patch i (computed for each patch)
q.meanP_p	phenotypic mean of patch i (computed for each patch)
q.varP_p	phenotypic variance of patch i (computed for each patch)
Genotype coancestry [q.(adlt/off).coa]	
q.(adlt/off).theta	mean within patch coancestry*
q.(adlt/off).alpha	mean between patch coancestry*
q.(adlt/off).thetaFF	mean within patch, within females coancestry*
q.(adlt/off).thetaMM	mean within patch, within males coancestry*
q.(adlt/off).thetaFM	mean within patch, between sexes coancestry*
q.(adlt/off).coa.fsib	mean coancestry within full-siblings*
q.(adlt/off).coa.phsib	mean coancestry within paternal half-siblings*
q.(adlt/off).coa.mhsib	mean coancestry within maternal half-siblings*
q.(adlt/off).coa.nsib	mean coancestry within non-siblings*
q.(adlt/off).theta_p	mean coancestry within patch i (computed for each patch)
q.(adlt/off).alpha_pair	mean coancestry between patch i and j (all pairwise combinations computed)
Genetic diversity [q.(adlt/off).gendiv]	
q.(adlt/off).nbAll	mean number of alleles per locus across the entire metapopulation*
q.(adlt/off).meanAll	mean number of alleles per locus and patch*
q.(adlt/off).nbFixLoc	number of fixed loci across the entire metapopulation*
q.(adlt/off).meanFixLoc	mean number of fixed loci per patches*
q.(adlt/off).ho	observed heterozygosity following Nei and Chesser (1983)*
q.(adlt/off).hs	expected heterozygosity following Nei and Chesser (1983)*
q.(adlt/off).ht	total expected heterozygosity following Nei and Chesser (1983)*
q.(adlt/off).nbAll_p	mean number of alleles per locus within patch i (computed for each patch)
q.(adlt/off).nbFixLoc_p	number of fixed loci within patch i (computed for each patch)

Table 6.1 continued on next page

Stat name	Description
q.(adlt/off).ho_p	observed heterozygosity within patch i following Nei and Chesser (1983) (computed for each patch)
q.(adlt/off).hs_p	expected heterozygosity within patch i following Nei and Chesser (1983) (computed for each patch)
F-statistics following Nei and Chesser (1983) [q.(adlt/off).fstat]	
q.(adlt/off).fst	global F_{ST}*
q.(adlt/off).fis	global F_{IS}*
q.(adlt/off).fit	global F_{IT}*
q.(adlt/off).fst_pair	pairwise F_{ST} between patch i and j (all pairwise combinations computed)
F-statistics following Weir and Cockerham (1984) [q.(adlt/off).fstat.wc]	
q.(adlt/off).fst.wc	global F_{ST}*
q.(adlt/off).fis.wc	global F_{IS}*
q.(adlt/off).fit.wc	global F_{IT}*
q.(adlt/off).fst.wc_pair	pairwise F_{ST} between patch i and j (all pairwise combinations computed)

Table 6.1: Summary statistics available for quantitative traits continued

Chapter 7

Neutral markers

quantiNEMO also allows the simulation of neutral markers, such as microsatellites or SNPs with different mutation models (K allele and Stepwise). Different types of neutral markers can be combined within the same simulation. The initial allele frequencies can be defined for each population separately.

7.1 Architecture

ntrl_loci [integer]
This parameter specifies the number of neutral marker loci per individual. This parameter is mandatory for the simulation of a neutral marker.

ntrl_all [1 to 256] (default: 255)
This parameter specifies the maximal number of alleles per locus (same number for each locus).

ntrl_allelic_file [string] (default: "")
This parameter allows to pass the name of a file containing allele informations, such as the initial allele frequencies, and/or the mutation probability to an allele. The number of alleles and loci has to be in line with the parameters ntrl_loci and ntrl_all. The information can be set globally for all loci, if they have the same specifications, or for each locus separately. The allelic file for neutral markers has the same format as the allelic file for quantitative traits, but does not allow to specify the allelic effects:

```
#Allelic file
###############
[FILE_INFO]{
  col_locus 1
  col_allele 2
  col_mut_freq 3
```

```
    col_ini_freq 4
}

#locus   allele    mut_freq   ini_freq
1        1         0.20       {0   0.2}
1        2         0.25       {0   0.2}
1        3         0.20       {1   0.2}
1        4         0.20       {0   0.2}
1        5         0.20       {0   0.2}
2        1         0.20       {0   0.2}
2        2         0.25       {0   0.2}
2        3         0.20       {1   0.2}
2        4         0.20       {0   0.2}
2        5         0.20       {0   0.2}
```

The file has to start with a file information box [FILE_INFO]{...}. This box contains the information of the structure of the following table allowing a flexible structure of the table. For example the order of the columns in the table may vary, or columns may be ignored. The file information box starts with the key word [FILE_INFO] and the information is enclosed by brackets "{...}". Line by line the index of the columns to be read have to be declared. Thereby a keyword for the specific setting is followed by the column index (the ordering starts with 1). The following column keywords are available:

col_locus This keyword specifies the column containing the locus index. If this column is not declared in the file information box, quantiNEMO will use the same settings for all loci. In this case the length of the table must meet the number of alleles (ntrl_all). If this keyword is declared the length of the table must meet the number of alleles times the number of loci (ntrl_loci * ntrl_all).

col_allele This keyword specifies the column containing the allele index. This column is mandatory. The index of the allele goes from 1 to ntrl_loci.

col_mut_freq This keyword specifies the column containing the mutation probabilities, i.e. the probability to mutate to this allele when a mutation occurs. The behavior of this mutation probability depends on the mutation model (see parameter (ntrl_mutation_model).

col_ini_freq This keyword specifies the column containing the initial frequencies of the alleles. This column allows to explicitly set the allele frequencies at the start of a simulation. The frequencies can be set for each patch separately using a matrix. In the example above, individuals of the first population are initially fixed for the allele 3 at the first locus as well as at the second locus. In the second population all alleles have the same initial frequency of 0.2. Note, that the matrix is adjusted in length if the number of populations does not correspond to the length of the matrix.

If this column is not given the initial allele frequencies are set globally depending on the parameter ntrl_ini_allele_model.

Note, that as in all input files for quantiNEMO it is possible to define comments (also in the file information box) using the hash character: '#' or '#/ ...any text... /#'.

ntrl_ini_allele_model [0,1] (default: 0)

This parameter allows to define the allele frequencies of the populations at the start of a simulation. Apart from these two models it is also possible to define the initial allele frequencies explicitly per population, locus and allele using the allelic file (parameter allelicFileNTRL, column keyword col_ini_freq). This parameter is ignored if the initial allele frequencies are defined by the allelic file.

0 : polymorph. The populations are maximally polymorph in respect to allele frequencies at the start of a simulation.

1 : monomorph. The populations are monomorph in respect to allele frequencies at the start of a simulation. All individuals are fixed for a single allele, which is the "middle" allele, i.e. the allele with the index $\lfloor \text{ntrl\_all}/2 \rfloor$.

7.2 Mutation

Mutation rates may be defined for each locus individually by explicitly defining the individual mutation rates (parameter ntrl_mutation_rate) or by defining the gamma distribution from which the individual mutation rates are drawn (parameters ntrl_mutation_rate and ntrl_mutation_var). There are two mutation models available. Using the allelic file for neutral markers (see section 7.1) it is possible to specify the probability to mutate to a certain allele explicitly for all alleles. Depending on the mutation model the new allele is the drawn one (KAM), or its index distance from the allele index in the middle is added to the current allele index (SSM). A minimal definition for mutations requires the setting of a common mutation rate (parameter ntrl_mutation_rate has a single value). In this case all loci have the same mutation rate and the mutation model is KAM.

ntrl_mutation_model [0,1] (default: 0)

This parameter allows to specify the mutation model. The following mutation models are available:

0 : KAM (K-Allele Model)

At each mutation the existing allele is randomly exchanged by another

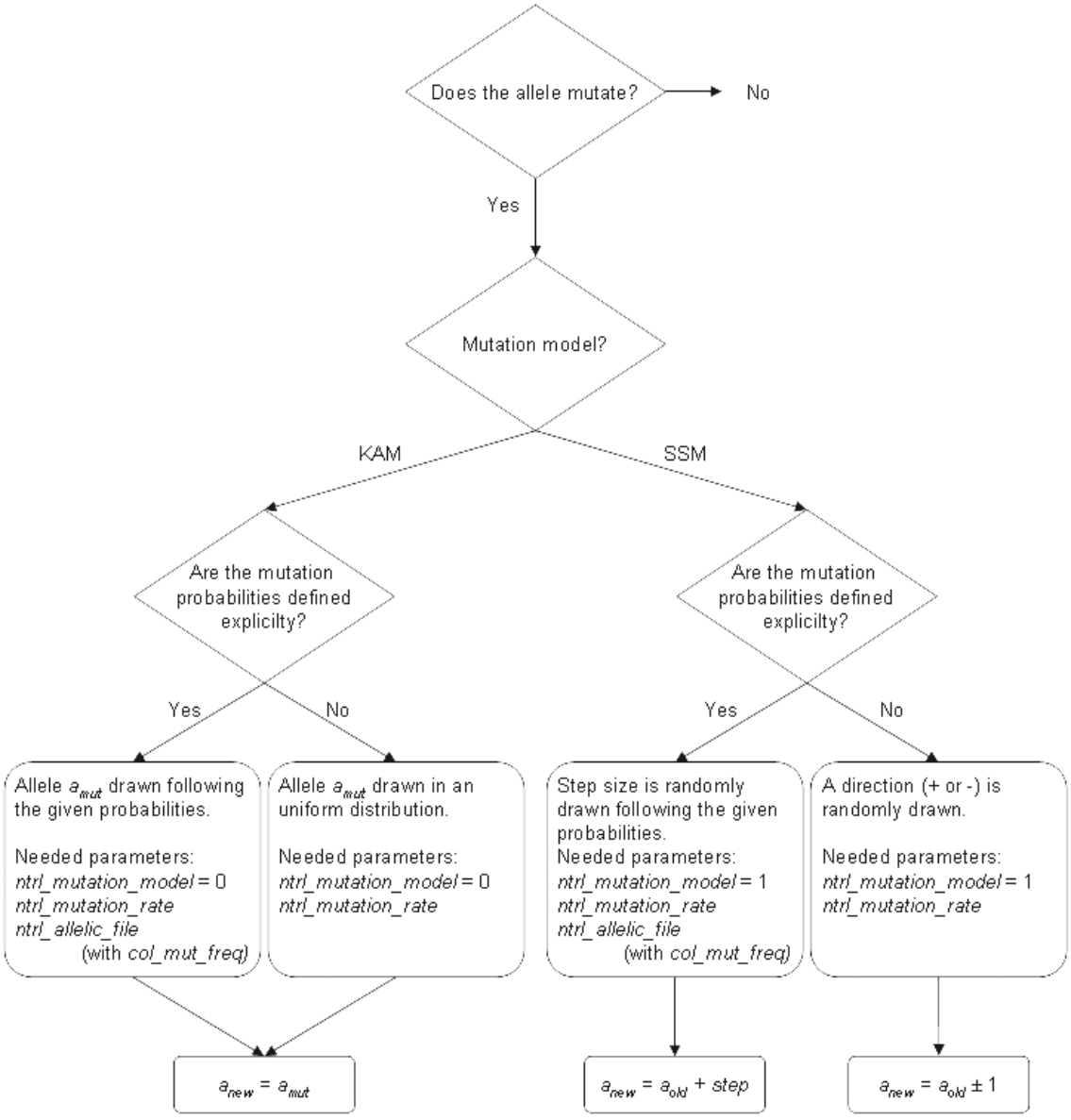


Figure 7.1: Schematic representation of the mutation process for a specific allele

allele within the range of alleles (i.e. $[1; ntrl\_all]$). By default the probability to mutate to any allele is the same. However, if the mutation probabilities are specified explicitly by the allelic file (see section 7.1) the probability to mutate to a certain allele depends on the specified probability to mutate to this allele.

1 : SSM (Single Step Mutation)
In contrast to the K-Allele-Model the new allele depends on the current allele. When a mutation occurs the current allele is replaced by one of its neighboring alleles (concerning the allele index). For example, if the allele with the index 12 mutates, it changes either to the allele with the index 13, or to the allele with the index 11. The boundaries are reflexive, i.e. the allele index can not exceed the range of alleles (i.e. $[1; ntrl\_all]$). In line with the Increment Mutation Model (IMM) for quantitative traits it is possible to specify mutation probabilities of the "steps to mutate" using the allelic file (see section 7.1). In this case the number of alleles (ntrl_all) has to be odd (as for the IMM). Then similar to the IMM the probabilities of the step to mutate has to be defined using the allelic file, where the step size is measured from the middle allele (concerning its index, i.e. $ntrl\_all/2$). The middle allele (with index $ntrl\_all/2$) has to have a mutation probability of 0 as a mutation of step zero is not a mutation. This allows generating any mutation pattern, as for instance the Generalized stepwise mutation model (Estoup et al., 2002).

ntrl_mutation_rate [decimal/matrix] (default: 0)
This parameter specifies the mutation rate per locus and generation. If the argument is a single value the mutation rate for all loci is the same. By passing a matrix of mutation rates it is possible to set the mutation rate for each single locus individually. By default no mutations occur.

ntrl_mutation_shape [integer] (default: 0)
This parameter allows to specify the shape of the gamma distribution from which the mutation rates for each locus are drawn. The mean mutation rate is given by the parameter ntrl_mutation_rate:

$$\Gamma(shape, mut\_rate/shape)$$

Where Γ is the gamma function requiring two parameters. The first parameter defines the shape of the gamma distribution (parameter ntrl_mutation_shape), and the second parameter the scale to the gamma distribution. The scale parameter is set such does the mean of the gamma distribution is equal to the mean mutation rate (mut_rate, respectively parameter ntrl_mutation_rate). If the shape is 0 (default value) all loci have the same mutation rate defined by the parameter ntrl_mutation_rate. The parameter ntrl_mutation_shape is ignored

if the mutation rates (parameter ntrl_mutation_rate) are explicitly defined for each locus by a matrix.

7.3 Multiple traits

Different types of neutral markers can be combined within the same simulation. Each type may have its own parameterization.

ntrl_nb_trait [integer] (default: 1)
This parameter defines the number of different types of neutral markers.

Each type of neutral markers may have its own specifications, but it is also possible to specify parameters for some types of neutral markers together. If several types of neutral markers are used it is possible to address a certain type by the number of the type. For instance to specify a parameter for the fifth type one has to add a "_5" to the parameter name. In contrast, if for the fifth trait no parameter with he suffix "_5" is passed quantiNEMO checks if the parameter is passed for the fourth type (suffix "_4"). If this is also not the case quantiNEMO checks if the parameter is passed for the third type (suffix "_3"), and so fort until a parameter is found. Note, that a parameter without a suffix is the same as the parameter with the suffix "_1". This behavior of quantiNEMO allows to specify parameters for a group of types of neutral markers. An example may make it more clear:

```
ntrl_nb_trait 2

# SNP (type 1)
ntrl_loci_1    100
ntrl_all_1     2

# STR (type 2)
ntrl_loci_2    10
ntrl_all_2     20

# same mutation rate for SNPs and STRs
ntrl_mutation_rate 0.0001
```

In this example we simulate 2 types of neutral markers: SNPs and STRs (microsatellites). We simulate 100 SNPs each with two alleles, and 10 STRs each with up to 20 alleles. The mutation rate is assumed to be the same for both types of markers, i.e. 0.0001.

7.4 Genetic map

quantiNEMO has an underlying genetic map, which may consist of several chromosomes. This allows to explicitly position all types of loci on the map (quantitative trait loci (QTL) and neutral markers). The unit of the genetic map is centi Morgans (Haldane, 1919, cM). If the genetic map is not specified by either of the following parameters, the loci are assumed to be unlinked and independent.

ntrl_loci_positions [matrix] (default: "")

For each trait this parameter allows to specify explicitly the loci position in centi Morgans from the beginning of the chromosome. The brackets separate the chromosomes. Within a chromosome loci are separated by a space. Note, that chromosomes may contain different numbers of loci, but that the total number of loci must correspond to the parameter ntrl_loci. It is possible to skip chromosomes if they don't contain loci (see section 2.3.5.4):

```
ntrl_loci 11
ntrl_loci_positions { {1: 10}
                      {3: 10 40 60 80 100}
                      {   10 40 60 80 100} }
```

In the example above, the neutral marker is defined by 11 loci located on three of at least four chromosomes. The first chromosome contains a single locus at position 10 cM. No loci are located on the second chromosome. The third and fourth chromosomes have 5 loci at positions 10, 40, 60, 80, and 100 cM.

ntrl_loci_positions_random [matrix] (default: "")

This parameter allows to specify the number of chromosomes and their lengths. The loci are then randomly positioned on the chromosomes. The matrix contains the length of each chromosome in cM. It is possible to "jump" chromosomes if they don't contain loci (see section 2.3.5.4):

```
ntrl_loci 11
ntrl_locus_positions_random { {1: 100} {3: 100} {100} }
```

11 loci of this type of neutral marker are located on three out of four chromosomes. The loci are randomly positioned on the first 100 cMs of each chromosome. No loci of this type of neutral marker are located on chromosome 2.

For every simulated replicate where a genetic map is simulated the entire genetic map (including QTLs and neutral loci) is dumped to a file named "genetic_map.txt" stored in the simulation folder. If multiple replicates are simulated a replicate counter (example _r3) is inserted before the extension. Using the parameter **genetic_map_output** it is possible to suppress this output.

7.5 Genotype output

The neutral genotype of all or a part of the individuals may periodically be stored to files similar to the genotype of quantitative traits. The output files will be stored in the folder given by the parameter ntrl_genot_dir and will have the name of the base file name (see parameter filename in section 4). The extension is ".dat". A counter for the generation (e.g. _g05) and the replicate (e.g. _r4) is inserted before the extension. An example of such a file name is "simulation_g05_r4.dat".

ntrl_save_genotype [0,1,2] (default: 0)
This parameter specifies the output of the neutral genotype.

0 : None. No output is generated.

1 : FSTAT. The output contains the genotypes in the FSTAT format (Goudet, 1995).

2 : FSTAT extended. Same as point 1, but the file contain the following six additional columns: the age class (1 = offspring, 2 = adult), the sex (0 = male; 1 = female), the ID of the individual, the ID of the mother, the ID of the father, and the fitness of the individual. The ID is a unique identifier for each individual of a simulation in the format "345_23", meaning that this is the 345th individual born in patch 23. The IDs of the individual, the mother and the father allow to extract pedigree informations, if the output is stored for each generation, and also to investigate the migration behavior of the individual and its parents.

An example of such a file (ntrl_save_genotype is set to 2):

```
5 4 20 2
loc-1
loc-2
loc-3
loc-4
1 1415 1019 2002 0820 1 1 10_1 1_1 0_1 0.345
1 0814 0219 2002 2020 1 1 11_1 8_1 2_4 0.334
1 0808 0217 1902 0820 1 1 12_1 5_3 5_1 0.123
...
5 1004 0917 1404 1007 1 1 16_5 9_5 3_2 0.999
5 2017 1010 2013 1812 1 0 17_5 3_2 9_2 1.000
5 2017 1008 2013 1811 1 1 11_4 8_2 9_2 0.678
```

The first line contains the number of patches (5 patches here), the number of loci (5), the highest possible allele index (20), and the number of digits used to write each allele (2). The next four lines contain the locus names. The following lines contain the individual's info, one individual per line. The first number is the patch number of the individual, followed by the genotype. Each

column represents a locus, and the first half of the locus represent the first allele index, while the second half of the locus the second allele at the given locus. In this example, we are using two digit per allele, the first two digits of a locus genotype number are the index of the first allele (e.g. allele 14 for the first allele of the first locus of the first individual) while the two next digits are the index of the second allele (e.g. allele 15 for the second allele of the first locus of the first individual). Each line ends with six columns consisting supplementary information on the individual (see above) if the parameter ntrl_save_genotype is set to 2.

ntrl_genot_dir [string] (default: "")
This parameter allows to specify the subdirectory where the genotypes are stored. If not specified (default) the output is stored in the simulation folder (parameter folder).

ntrl_genot_logtime [integer] (temporal/default: 1)
This parameter specifies the interval of the genotype output. Since the parameter may change over time the output may be generated at any generation.

ntrl_genot_script [string] (default: "")
It is possible to launch a script just after the genotype file is generated. The argument of the parameter is the file name of the script. The name of the genotype file is passed as unique parameter to the script.

ntrl_genot_sex [0,1,2] (default: 0)
This parameter allows to choose which sex is output.

0 : Both. Output includes both sexes.

1 : Females. Output includes only female genotypes.

2 : Males. Output includes only male genotypes.

ntrl_genot_age [0,1,2] (default: 0)
This parameter allows to choose which age is output.

0 : Adults. Output includes only adult genotypes.

1 : Juveniles. Output includes only juvenile genotypes.

2 : Both. Output includes juveniles and adults genotypes.

7.6 Summary statistics

The summary statistics listed in the table below are available for neutral markers. The column Stat name contains the name of the summary statistic used to specify which summary statistics are computed (parameter stat, for details see section 3.2).

These names appear also in the output file. The column Description contains a short description of the summary statistic.

Some of the summary statistics are available for adults and offspring (indicated by (adlt/off)). To obtain a certain summary statistic for adults the prefix adlt. has to be added to the summary statistic name (e.g. adlt.allnb), respectively the prefix off. to obtain the summary statistic for offspring (e.g. off.allnb).

Some of the summary statistics are computed for each patch separately (indicated by *(computed for each patch)* in the table), others for each pair of patches (indicated by *(all pairwise combinations computed)*). Depending on the number of patches this may lead to the computation of a large number of summary statistics. The names of such summary statistics are marked in the output file with the index of the patch: "_pX", respectively "_pX-Y" for pairwise summary statistics. X and Y stand for the patch index starting with 1.

The summary statistics are computed for every type of neutral marker. If several quantitative traits are simulated the postfix "_tY" is added to the summary statistic name in the output file.

The summary statistic name (column Stat name) may be used to specify the summary statistic to be computed (e.g. adlt.fst). Similar summary statistics (within a thematic group) may be obtained at once using the name within square brackets after the group title (e.g. adlt.fstat). Using this group statistic name all summary statistics of the thematic group marked with a star (*) will be computed.

Table 7.1: Summary statistics available for neutral markers

Stat name	Description
Genotype coancestry [n.(adlt/off).coa]	
n.(adlt/off).theta	mean within patch coancestry*
n.(adlt/off).alpha	mean between patch coancestry*
n.(adlt/off).thetaFF	mean within patch, within females coancestry*
n.(adlt/off).thetaMM	mean within patch, within males coancestry*
n.(adlt/off).thetaFM	mean within patch, between sexes coancestry*
n.(adlt/off).coa.fsib	mean coancestry within full-siblings*
n.(adlt/off).coa.phsib	mean coancestry within paternal half-siblings*
n.(adlt/off).coa.mhsib	mean coancestry within maternal half-siblings*
n.(adlt/off).coa.nsib	mean coancestry within non-siblings*
n.(adlt/off).theta_p	mean coancestry within patch i (computed for each patch)
n.(adlt/off).alpha_pair	mean coancestry between patch i and j (all pairwise combinations computed)

Table 7.1 continued on next page

Stat name	Description
Genetic diversity [n.(adlt/off).gendiv]	
n.(adlt/off).nbAll	mean number of alleles per locus across the entire metapopulation*
n.(adlt/off).meanAll	mean number of alleles per locus and patch*
n.(adlt/off).nbFixLoc	number of fixed loci across the entire metapopulation*
n.(adlt/off).meanFixLoc	mean number of fixed loci per patches*
n.(adlt/off).ho	observed heterozygosity following Nei and Chesser (1983)*
n.(adlt/off).hs	expected heterozygosity following Nei and Chesser (1983)*
n.(adlt/off).ht	total expected heterozygosity following Nei and Chesser (1983)*
n.(adlt/off).nbAll_p	mean number of alleles per locus within patch i (computed for each patch)
n.(adlt/off).nbFixLoc_p	number of fixed loci within patch i (computed for each patch)
n.(adlt/off).ho_p	observed heterozygosity within patch i following Nei and Chesser (1983) (computed for each patch)
n.(adlt/off).hs_p	expected heterozygosity within patch i following Nei and Chesser (1983) (computed for each patch)
F-statistics following Nei and Chesser (1983) [n.(adlt/off).fstat]	
n.(adlt/off).fst	global F_{ST}*
n.(adlt/off).fis	global F_{IS}*
n.(adlt/off).fit	global F_{IT}*
n.(adlt/off).fst_pair	pairwise F_{ST} between patch i and j (all pairwise combinations computed)
F-statistics following Weir and Cockerham (1984) [n.(adlt/off).fstat.wc]	
n.(adlt/off).fst.wc	global F_{ST}*
n.(adlt/off).fis.wc	global F_{IS}*
n.(adlt/off).fit.wc	global F_{IT}*
n.(adlt/off).fst.wc_pair	pairwise F_{ST} between patch i and j (all pairwise combinations computed)

Table 7.1: Summary statistics available for neutral markers continued

Bibliography

Beverton, R. J. H. and Holt, S. J. (1957), On the dynamics of exploited fish populations, Technical report, U.K. Ministry of Agriculture and Fisherie.

Bürger, R. (2000), *The Mathematical Theory of Selection, Recombination, and Mutation*, Wiley, Cichester, UK.

Estoup, A., Jarne, P. and Cornuet, J. M. (2002), 'Homoplasy and mutation model at microsatellite loci and their consequences for population genetics analysis', *Molecular Ecology* **11**(9), 1591–1604.

Goudet, J. (1995), 'Fstat (version 1.2): A computer program to calculate f-statistics', *Journal of Heredity* **86**(6), 485–486.

Guillaume, F. and Rougemont, J. (2006), 'Nemo: an evolutionary and population genetics programming framework', *Bioinformatics* **22**(20), 2556–2557.

Haldane, J. B. S. (1919), 'The combination of linkage values, and the calculation of distances between loci of linked factors. journal of genetics', *Journal of Genetics* **8**, 299–309.

Lynch, M. and Walsh, B. (1998), *Genetics and analysis of quantitative traits*, Publisher Sinauer Associates Inc.

Nei, M. and Chesser, R. K. (1983), 'Estimation of fixation indexes and gene diversities', *Annals of Human Genetics* **47**(Jul), 253–259.

Ravigne, V., Olivieri, I. and Dieckmann, U. (2004), 'Implications of habitat choice for protected polymorphisms', *Evolutionary Ecology Research* **6**(1), 125–145.

Richards, F. (1959), 'A flexible growth function for empirical use', *J. Exp. Bot* **10**, 290–300.

Wallace, B. (1968), Polymorphism, population size, and genetic load, *in* R. C. Lewontin, ed., 'Population biology and evolution', Syracuse University Press, Syracuse, N. Y., pp. 87–108.

Wallace, B. (1975), 'Hard and soft selection revisited', *Evolution* **29**, 465–473.

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

Index

breed_model, 21

dispersal_border_model, 30
dispersal_k_growth_rate, 32
dispersal_k_max, 32
dispersal_k_max_growth, 32
dispersal_k_min, 32
dispersal_k_symmetry, 33
dispersal_lattice_dims, 30
dispersal_lattice_range, 30
dispersal_model, 29
dispersal_propagule_prob, 30
dispersal_rate, 28
dispersal_rate_fem, 28
dispersal_rate_mal, 28

extinction_rate, 33

filename, 35
folder, 34

generations, 34
genetic_map_output, 36
growth_rate, 23

logfile, 35
logfile_type, 35

mating_males, 25
mating_nb_offspring_model, 22
mating_proportion, 25
mating_system, 24
mean_fecundity, 24

ntrl_all, 76
ntrl_allelic_file, 76
ntrl_genot_age , 84
ntrl_genot_dir, 84
ntrl_genot_logtime, 84
ntrl_genot_script, 84
ntrl_genot_sex, 84
ntrl_ini_allele_model, 78
ntrl_loci, 76
ntrl_loci_positions, 82
ntrl_loci_positions_random, 82
ntrl_mutation_model, 78
ntrl_mutation_rate, 80
ntrl_mutation_shape, 80
ntrl_nb_trait, 81
ntrl_save_genotype, 83

overwrite, 34

patch_capacity, 38
patch_capacity_fem, 38
patch_capacity_mal, 38
patch_dir_sel_growth_rate, 43
patch_dir_sel_growth_rate_fem, 43
patch_dir_sel_growth_rate_mal, 43
patch_dir_sel_growth_rate_var, 44
patch_dir_sel_max, 43
patch_dir_sel_max_fem, 43
patch_dir_sel_max_growth, 43
patch_dir_sel_max_growth_fem, 43
patch_dir_sel_max_growth_mal, 43
patch_dir_sel_max_growth_var, 44
patch_dir_sel_max_mal, 43
patch_dir_sel_min, 42
patch_dir_sel_min_fem, 42
patch_dir_sel_min_mal, 43
patch_dir_sel_symmetry, 43
patch_dir_sel_symmetry_fem, 44
patch_dir_sel_symmetry_mal, 44
patch_dir_sel_symmetry_var, 44

patch_ini_size, 39
patch_ini_size_fem, 39
patch_ini_size_mal, 39
patch_number, 38
patch_stab_sel_intensity, 41
patch_stab_sel_intensity_fem, 41
patch_stab_sel_intensity_mal, 41
patch_stab_sel_intensity_var, 42
patch_stab_sel_optima, 41
patch_stab_sel_optima_fem, 41
patch_stab_sel_optima_mal, 41
patch_stab_sel_optima_var, 42
postexec_script, 36

quanti_all, 50
quanti_allelic_file, 50
quanti_allelic_var, 52
quanti_dominance_file, 53
quanti_dominance_mean, 54
quanti_dominance_var, 54
quanti_environmental_model, 57
quanti_environmental_proportion, 58
quanti_epistatic_file, 55
quanti_epistatic_var, 56
quanti_geno_value_age, 70
quanti_geno_value_dir, 70
quanti_geno_value_logtime, 70
quanti_geno_value_script, 70
quanti_geno_value_sex, 70
quanti_genot_age, 69
quanti_genot_dir, 68
quanti_genot_logtime, 68
quanti_genot_script, 68
quanti_genot_sex, 68
quanti_heritability, 58
quanti_ini_allele_model, 52
quanti_loci, 50
quanti_loci_positions, 66
quanti_loci_positions_random, 66
quanti_mutation_model, 62
quanti_mutation_rate, 63
quanti_mutation_shape, 63
quanti_nb_trait, 64
quanti_output, 72
quanti_phenot_dir, 72
quanti_phenot_logtime, 72
quanti_phenot_script, 72
quanti_phenot_sex, 72
quanti_save_geno_value, 69
quanti_save_genotype, 67
quanti_save_phenotype, 71
quanti_sel_coef_heterozygote, 60
quanti_sel_coef_homozygote, 60
quanti_selection_model, 65
quanti_va_model, 58

regulation_model_adults, 33
regulation_model_offspring, 28
replicates, 34

sampled_patches, 36
seed, 35
sex_ratio, 25
stat, 27
stat_dir, 27
stat_log_time, 26
stat_save, 26