AlphaSim Documentation

Release 1.01

Anne-Michelle Faux, Gregor Gorjanc and John M. Hickey

September 15, 2015

CONTENTS

Contents:

CHAPTER ONE

ALPHASIM

Table of Contents

1.1 Introduction

AlphaSim is a software package for simulating sequence, genotypic, phenotypic, and pedigree data in a wide range of population types (e.g., animal breeding, plant breeding, human genetics, natural populations). In developing AlphaSim major emphasis was placed on flexibility, computational efficiency, and ease of use. Populations can have almost any pedigree structure, almost any type of selection, a wide range of genotyping platforms, and multiple traits with a wide range of alternatives for genetic architecture. Additionally, AlphaSim also includes features for genome editing, manipulating the recombination rate and its genetic architecture, performing optimal contribution selection, and other "tricks".

1.2 Program history

AlphaSim is a new program that unifies new features with previously developed of simulation packages written by John Hickey (JH) and collaborators into a single flexible software package. The first version was written by JH in 2007 and simulated pedigrees, breeding values and phenotypes under the classical infinitesimal model of quantitative genetics. In 2008 functionality to simulate genomic data (i.e., haplotypes and genotypes), including efficient methods for storing and representing sequence and genomic data implicitly were added. In 2009 functionality to simulate breeding values and phenotypes under the various QTN (quantitative trait nucleotide) models were added and the program was renamed AlphaDrop. Functionality for selection was added in 2011. In 2012 AlphaSimPlant, a program for simulating a sub set of plant breeding scenarios was developed in collaboration between JH and Gregor Gorjanc (GG). AlphaSimPlant used a large portion of AlphaDrop source code but some additional features were also added including an efficient method to store and track identical-by-descent information. In 2013 JH, Emma Huang, and GG developed AlphaMPSim, based to a large extent on AlphaDrop and AlphaSimPlant source code. AlphaMPSim had new functionality to simulate the typical pedigree structures of multiparent populations that are commonly used in in plant genetics and model species (e.g., MAGIC, NAM, etc). In 2014 functionality for the promotion of alleles by genome editing (PAGE), manipulation of recombination rate, and de-novo mutations was added to AlphaDrop in collaboration with contributions from JH, GG, Janez Jenko (JJ) and Gabor Meszaros (GM). At this point we realised that the various incarnations of the original AlphaDrop program were beginning to diverge and that supporting the simulation suite was becoming unsustainable. Anne-Michelle Faux (AMF), GG and JH collaborated to undertake a major rewrite of the whole code and thus develop a single unified simulation package called AlphaSim. During this rewrite several new features were added, including multiple traits, extra genetic architectures, a more general framework for pedigree structures (thus enabling complex plant breeding pedigrees be simulated), incorporation of optimal contributions, dominance effects, and several ease of use, flexibility and computational efficiency improvements. AMF wrote the present user manual.

1.3 List of contributors to the development and testing of AlphaSim

John M Hickey, Gregor Gorjanc, Anne-Michelle Faux, Serap Gonen, David Ludwig Wilson, Janez Jenko, Stefan Hoj-Edwards, Chris Gaynor, Gabor Meszaros, Emma Huang, Sarah Hearne, and Sam Clark.

1.4 Funding

The development of AlphaSim has been largely not funded by specific grants, instead the work has been undertaken as part of other projects which have been funded by The Irish Cattle Breeding Federation, Teagasc, The Australian Research Council, Aviagen LTD, Genus PLC, Pfizer Inc., The Sheep CRC, CIMMYT, Advanta Semillas, and the Seeds of Discovery. CIMMYT, through The Seeds of Discovery Project, and the ISPG funding from the BBSRC to The Roslin Institute have funded the work of Anne-Michelle Faux.

1.5 Availability

AlphaSim is available from: http://www.alphagenes.roslin.ed.ac.uk/software-packages/alphasim/

Material available includes the compiled programs for 64 bit Linux and Mac OSX machines, together with a User Manual.

Please report bugs or suggestions on how the program / user interface / manual could be improved or made more user friendly to John.Hickey@roslin.ed.ac.uk.

1.6 Conditions of use

AlphaSim is available to the scientific community free of charge. Users are required, however, to credit its use in any publications. Commercial users should contact John Hickey (John.Hickey@roslin.ed.ac.uk).

We are currently preparing an updated manuscript that will describe the new features of AlphaSim. We would suggest that you would use this future manuscript as the reference when using AlphaSim. Until this paper is published we suggest you use the original AlphaDrop paper:

> Hickey, J.M., Gorjanc, G., 2012. Simulated Data for Genomic Selection and Genome-Wide Association Studies Using a Combination of Coalescent and Gene Drop Methods. G3 2, 425–427)

If you use some of the special features of AlphaSim we suggest that you also cite the specific paper that developed that particular feature:

1. When using the data sets simulated using AlphaSim which are suggested as benchmark data sets by Genetics and G3 please also cite: Hickey, J.M., Gorjanc, G., 2012. Simulated Data for Genomic Selection and Genome-Wide Association Studies Using a Combination of Coalescent and Gene Drop Methods. G3 2, 425–427
2. When using the selection options please also cite: Gorjanc, G., Bijma, P., and Hickey, J.M. (2015) Reliability of pedigree and genomic evaluation in selected populations. Genetics Selection Evolution (Submitted)
3. When using the default option for generating founder haplotypes please also cite: Chen, G.K., Marjoram, P., Wall, J.D., 2009. Fast and flexible simulation of DNA sequence data. Genome Res. 19, 136–142. doi:10.1101/gr.083634.108
4. Papers describing manipulation or recombination rate and the use of optimal contributions are under preparation.
5. For promotion of alleles by genome editing (PAGE) please also cite: Jenko, J., Gorjanc, G., Cleveland, M.A., Varshney, R.K., Whitelaw, C.B.A., Woolliams, J.A., and Hickey, J.M. (2015) Potential of promotion of alleles by genome editing for improving quantitative traits in livestock breeding programs. Genetics Selection Evolution (Submitted)

1.7 Disclaimer

While every effort has been made to ensure that AlphaSim does what it claims to do, there is absolutely no guarantee that the results provided are correct. Use of AlphaSim is entirely at your own risk!

1.8 Advertisement

AlphaSim is part of the AlphaSuite collection of software programs that we have developed. The AlphaSuite collection can perform many of the common tasks in animal breeding, plant breeding, and human genetics including genomic prediction, breeding value estimation, variance component estimation, GWAS, imputation, phasing, optimal contributions, simulation, field trial designs, and various data recoding and handling tools.

The AlphaSuite [1] is available at this link: http://www.alphagenes.roslin.ed.ac.uk/software-packages/

[1] People often ask as to the origins on the AlphaSuite naming convention. The original program in the AlphaSuite was AlphaBayes, which was named after the "Bayesian Alphabet" that it implemented. The name was suggested by Brian Kinghorn. Subsequent programs evolved via mutation with a conserved primer!

1.9 Description of the method

The method implemented in AlphaSim to simulate pedigree, sequence, SNP (single nucleotide polymorphism) and QTN (quantitative trait nucleotide) data is described in Hickey and Gorjanc (2012). In simple terms AlphaSim takes in founder haplotypes (which can be whole genome sequence) and drops them through a pedigree, extracts QTN and marker genotypes, and simulates phenotypes, true breeding values, and selection.

The **pedigree** can be provided or simulated internally. In the latter case, a pedigree is generated according to specified generation sizes and numbers of male and female parents; these numbers can be made constant or variable. When working with plant populations, AlphaSim allows selfing and creating doubled haploid lines.

The default system for generating the founder **haplotypes** is a system call to MaCS, a coalescent simulation program which simulates a sample of haplotypes with sequence information for each chromosome according to the ancestral population and mutation and recombination rates specified by the user (Chen et al., 2009). The call to MaCS is via a shell script that can be modified to replace with any other program or to take in a pre-existing set of haplotypes. AlphaSim then samples haplotypes with replacement from the final generation of the ancestral population simulated using MaCS or from the set of external haplotypes, and drops them according to a user-defined recombination rate in the first generation of the pedigree.

AlphaSim efficiently stores sequence information, and this makes the simulation of sequence data in large pedigrees computationally feasible. Gametes comprise strings of 0s and 1s, representing segregating sites. Gametes can therefore be thought of as large binary numbers and represented as integers. AlphaSim breaks gametes into distinct blocks, each of them including a constant number of segregating sites, and represents each block as a long integer. These long integers are only decompressed into their binary numbers where a recombination occurs. The segregating sites that remain after breaking gametes into blocks are discarded. Output files reporting information on the segregating sites that are included in blocks are indicated by the word ‘Included’ in section Chromosomes.

Next, the segregating sites are sampled to become **SNP markers** to form distinct **SNP chips** characterized by distinct densities. The frequency of the minor allele of the SNP markers can be restricted to a specified range, and the SNP chips can be nested to each other or not. Both genotypes and haplotypes are generated for all SNP chips. The full **sequence** data can also be generated.

AlphaSim then selects two samples of segregating sites to possibly become **QTN**. These are called candidate QTN. The first set comprises a user-specified number of candidate QTN, selected at random, from across the genome. The second set comprises a user-specified number of candidate QTN, selected at random, from across the genome, with the restriction that the minor allele frequency must be comprised in a specified range. This restriction facilitates the possibility that QTN have lower minor allele frequency than SNP. The allele substitution effect at each QTN can be sampled from a normal distribution with a mean of zero and standard deviation of one unit, or from a Gamma distribution with a user-specified shape and scale parameter and a 50% chance of being positive or negative. In total, four different QTN models are generated assuming an additive genetic model, differing from each other by the distribution of allele substitution effects and the use of restriction on the minor allele frequency of QTN or not.

Mutations can be inserted in the sequence data. The number of mutations per chromosome and the mutational heritability are specified by the user. AlphaSim considers that all new mutations are QTN, so that the total number of QTN is increased by the total number of mutations.

Selection can be carried out using true breeding values or genomic estimated breeding values. In the latter case, distinct strategies for selecting a training population have been implemented. Also, external training data can be imported to compute gEBVs. AlphaSim performs selection by **truncation** (the best performing individuals are selected) or by **optimal contribution** (individuals are selected according to both their breeding value and the average co-ancestry among them). In addition, the program allows **selecting for recombination** as well as **editing selected male parents**.

Distinct traits can be simulated; each of them characterized by a distinct heritability and genetic variance, and the genetic and residual correlations among the traits can be specified. Selection is then performed using an **index** that combines the distinct traits using user-specified weights.

AlphaSim can be run sequentially, i.e., it can be stopped after a given number of generations and then restarted. This **flexibility** allows the value of some parameters to be modified, such as the type of training population or the method of selection – truncation or optimal contribution –, from one to another generation. Also, AlphaSim allows working with an internal pedigree while making the selection and mating decisions outside the program based on the simulated breeding values and importing the externally created pedigree. This feature is flexible in terms of generations.

Output files, including the full sequence and phased data of all individuals across the pedigree, can be created and compressed in zip files, if requested. Also, the genome can be compressed by dropping only a percentage of the total amount of simulated segregating sites into the pedigree. This **genome compression** option allows, in particular, simulating large genomes with properties corresponding to a given effective population size, while restricting the amount of segregating sites that are dropped through the pedigree and, in fine, reducing the required computational resources.

More details about the methods used by AlphaSim are provided in section AlphaSimSpec.txt below, which describes the parameters file.

Note: Manipulation of the recombination rate, gene editing, and optimal contribution selection are not enabled until such time as the papers describing the methods have been published. These options refer to Box 8, Box 9 and Box 11 in the parameters file. We apologize for the inconvenience.

1.10 Using AlphaSim

1.10.1 Input files

Running AlphaSim basically requires a file providing the program parameters, founder haplotypes and a seed. The parameters file, *AlphaSimSpec.txt*, is described in section AlphaSimSpec.txt. By default, founder haplotypes are supplied by calling the external software package MaCS. The MaCS simulation parameters are controlled internally by AlphaSim, or though a user-specified input file. A description of the different MaCS simulations scenarios available in AlphaSim are described below in the description of the *AlphaSimSpec.txt* file. The seed for initialising the MaCS program is provided in a file called *Seed.txt* (see section Seed.txt).

Additional files are required when working with an external pedigree (see section PedigreeFiles), importing an external pedigree for a given selection generation while working with an internal pedigree (see section PedigreeFiles), performing gene editing on a user-specified selection of QTN (section UserSelectedQtnToBeEdited.txt), or using external solutions for the SNP effects (section SnpSolutionsFiles).

In addition to these input files, running AlphaSim requires several additional programs. Details about these programs are provided in section AdditionalPrograms.

AlphaSimSpec.txt (mandatory)

An example of *AlphaSimSpec.txt* is shown in Figure 1. Text to the left of the comma should not be changed. The program is controlled by changing the input right of the first comma:

```
###############################################################
BOX 1 - PEDIGREE
PedigreeStatus                                   ,Internal
NumberGenerationsBurnIn                          ,3
NumberGenerationsSelection                       ,3
NumberOfIndividualsInEachGeneration              ,Variable,500,50,25,20,20,20
###############################################################
BOX 2 - ANIMAL BREEDING OPTIONS
AnimalBreedingOnOff                              ,Off
NumberOfSiresInEachGeneration                    ,Constant,25
NumberOfDamsInEachGeneration                     ,Constant,25
###############################################################
```

```
BOX 3 - PLANT BREEDING OPTIONS
PlantBreedingOnOff                                ,On
PerformCrossesConsideringTheGenderYesNo           ,Yes
NumberOfMaleParentsInEachGeneration               ,Variable,50,25,0,0,0
NumberOfFemaleParentsInEachGeneration             ,Variable,0,25,0,0,0
NumberOfParentsRegardlessTheGenderInEachGen       ,Constant,10
PerformSelfingInAtLeastOneGenerationYesNo         ,Yes
NumberOfSelfedParentsInEachGeneration             ,Variable,0,0,5,5,5
NumberOfGenerationsIncludingDoubledHaploids       ,1
CreateDoubledHaploidsInTheseGenerations           ,2
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                        ,Off
StartStopGeneration                               ,1,4
SelectionFlexibilityOnOff                         ,Off
ImportAnExternalPedigreeForThisGeneration         ,4
ImportStagePut1ToComputeBV2ToContinue             ,1
MergeTypes                                        ,none
NumberOfMergeRuns                                 ,0
MergeFiles                                        ,/path/
###############################################################
BOX 5 - CHROMOSOMES
NumberOfChromosomes                               ,2
ChromosomeLengthMorgans                           ,1.0
NumberOfHaplotypes                                ,1000
ChromosomeSource                                  ,internaltest
###############################################################
BOX 6 - SNP CHIPS
UseSnpChipOnOff                                   ,On
NumberOfSnpChips                                  ,2
nSnpPerChipPerChromosme                           ,200,100
MinMaxAlleleFreqForSnp                            ,0.1,0.5
SnpChipIncludesQtnYesNoRandom                     ,No
SnpChipsAreNestedYesNo                            ,Yes
IdOfChipUsedForSelection                          ,1
###############################################################
BOX 7 - QTN
NumberOfQtnPerChromosomeNormalDist                ,100
NumberOfQtnPerChromosomeGammaDist                 ,50
NumberOfMutationsPerChr                           ,0
MutationalVariance                                ,0.1
MinMaxAlleleFreqForModel2And4                     ,0.0,0.3
ShapeAndScaleOfGammaDistForModel3And4             ,0.30,0.40
IncludeDominanceEffectsOnOff                      ,On
DominanceDegreeMeanAndVariance                    ,0.5,0.10
###############################################################
BOX 8 - RECOMBINATION
RecombinationOnOff                                ,Off
HeritabilityOfRecombination                       ,0.6
NumberOfQtnPerChromForRecombination               ,100
InitialRecombinationIndexWeight                   ,0.8
NumberGenerationsSelectionOnRecombination         ,2
LaterGenRecombinationIndexWeight                  ,0.2
DistributionOfQtnForRecombination                 ,Normal
ShapeAndScaleOfGammaDist                          ,0.30,0.40
###############################################################
BOX 9 - GENE EDITING
GeneEditingOnOff                                  ,Off
```

```
EditTopOrBottom                                   ,Top
NumberOfEditedIndividuals                         ,5
NumberOfGeneEditsPerIndividual                    ,20
PerformGEOnUserQtnSelectionYesNo                  ,No
###############################################################
BOX 10 - SELECTION
SelectionBasedOnRestrictedOrUnrestrictedQtn       ,Unrestricted
DistributionOfQtnOfSelectionTraits                ,Normal
SelectionMethod                                   ,GBLUP
AddQtnToTheSnpChipYesNo                           ,Yes
UseExternalSnpSolutionsYesNo                      ,No
SelectionPhenotypingStrategy                      ,RandomPhenotypes
FirstAndLastTrainingGenForRandomPhenotypesStrat   ,1,1
TrainingSetSizeForRandomPhenotypesStrat           ,400
FixTheTrainingDataAcrossGenerationsOnOff          ,On
UseTrainingDataFromGeneration                     ,4
ComputeAverageCoancestryYesNo                     ,Yes
###############################################################
BOX 11 - OPTIMAL CONTRIBUTION SELECTION
OptimalContributionSelectionOnOff                 ,Off
SizeOfXvectPopulationAndMaxNbOfGenerations        ,50,5000
PercentageDifferenceInbreeding                    ,0.05
###############################################################
BOX 12 - TRAITS
NumberOfTraits                                    ,2
IndexWeights                                      ,0.7,0.3
TraitHeritability                                 ,0.8,0.6
TraitVariances                                    ,1.0,1.0
###############################################################
BOX 13 - TRAIT CORRELATIONS
Genetic Correlation Matrix
1.0
0.6 1.0
Residual Correlation Matrix
1.0
0.4 1.0
###############################################################
BOX 14 - OUTPUTS
KeepChromosomesFolderYesNo                        ,Yes
WriteFullSequenceOutOnOff                         ,On
ZipSequenceOutputFilesYesNo                       ,Yes
###############################################################
BOX 15 - GENOME COMPRESSION
GenomeCompressionOnOff                            ,Off
GenomeCompression                                 ,50.00
###############################################################
```

Figure 1. Example of AlphaSimSpec.txt

AlphaSimSpec.txt includes fifteen boxes.

Note: Each line including a `...OnOff` parameter is followed by sub-parameters that are only read if the `...OnOff` parameter is set to *On*. The number of sub-parameters varies from one case to another one; please read carefully the user manual to fill in properly the parameters file. If the `...OnOff` parameter is set to *Off*, AlphaSim does not read the sub-parameters, and no value has to be provided for these sub-parameters.

Box 1: Pedigree

AlphaSim is able to deal with both external and internal pedigrees. The pedigree status is defined by the **PedigreeStatus** parameter, which is either the name of the external file containing the pedigree, such as *PedigreeFile.txt*, or set to internal via keyword *Internal*. If using an external pedigree, details on the format of the file are given in section PedigreeFiles. If the pedigree is internal, AlphaSim simulates a pedigree by using additional information related to the number of generations and the number of individuals per generation:

NumberGenerationsBurnIn is the number of burn-in generations, i.e., the number of generations in which no selection is performed.

NumberGenerationsSelection is the number of generations in which selection is performed.

For convenience, the numbers of burn-in and selection generations are thereafter referred to as *nGen* and *nGenSel*, respectively.

The parameter **NumberOfIndividualsInEachGeneration** includes a character string specifying if the number of individuals per generation is constant or variable, followed by one or several number(s) indicating the generation sizes, each of them being separated by a comma. The character string is *Constant* or *Variable*. If the number of individuals per generation is constant, the term *Constant* must be followed by only one value of generation size. If the number of individuals per generation is *Variable*, the size of each generation has to be specified, and thus the term *Variable* must followed by a total of *nGen+nGenSel* values. The number of individuals in a given generation *g* is thereafter referred to as *nIndivInThisGen(j)*.

When using an internal pedigree, either **AnimalBreedingOnOff** (Box 2) or **PlantBreedingOnOff** (Box 3) must be `On`.

Note:

- **Constructing a pedigree that integrates both external and internally simulated generations** AlphaSim allows the construction of a pedigree that combines internally simulated generations with external generations read in an external file. This flexibility of the software can be used to incorporate generations including very specific crosses. Combining external and internally simulated generations is made possible thanks to the flexibility of AlphaSim in terms of generation. An example is provided below in Box 4 using the option **GenerationFlexibility**.
- **Performing selection in a pedigree**
  - Selection in **internal pedigrees** is straightforward. To this purpose, the user specifies the number of selection generations and their size (Box 1), the number of individuals to be selected (Box 2 or Box 3), and the selection parameters (Box 10 and Box 11).
  - AlphaSim allows **adding selection generations to external pedigrees**. To perform selection only in the last generation of the external pedigree, the pedigree needs to be imported in two steps, the last generation being imported apart from the ones that precede. As when using an internal pedigree, the user specifies the number of selection generations to be simulated and their size (Box 1), the number of individuals to be selected (Box 2 or Box 3), and the selection parameters (Box 10 and Box 11). An example of how to add selection generations to an external pedigree is provided in Box 4 using the option **GenerationFlexibility**.
  - In addition, AlphaSim allows **importing externally created selection generations**. This option enables the application of any alternative selection method through the ability of the software to read in external pedigrees. An example of how to import an external selection generation is provided in Box 4 using the option **SelectionFlexibility**.

Box 2: Animal breeding options

AnimalBreedingOnOff can be *On* or *Off*. If **AnimalBreedingOnOff** is *On*, then Box 2 must be filled in as follows.

NumberOfSiresInEachGeneration determines the number of male animals from a given generation used to produce the next generation. The format of this parameter is the same as for **NumberOfIndividualsInEachGeneration** with the exception of the number of values that must be provided if the number of selected male animals varies among generations. In that case, the term Variable must be followed by a total of *nGen+nGenSel-1* numbers, one for each generation with the exception of the last one in which selection is no longer performed.

NumberOfDamsInEachGeneration determines the number of female animals from a given generation used to produce the next generation. The format of this parameter is the same as for **NumberOfSiresInEachGeneration**.

Note: For a given generation *g*, **NumberOfSiresInEachGeneration** and **NumberOfDamsInEachGeneration** must be smaller or equal to *nIndivInThisGen(j)/2*.

Box 3: Plant breeding options

PlantBreedingOnOff can be on *On* or *Off*. If *On*, then Box 3 must be filled in as follows.

PerformCrossesConsideringTheGenderYesNo can be either *Yes* or *No*. If *Yes*, then the gender of each individual is considered when performing crosses, i.e., a given individual cannot be used as both male and female parent. If *No*, an individual can be used as a male parent in a given cross and as a female parent in another cross while selfing is not allowed.

NumberOfMaleParentsInEachGeneration and **NumberOfFemaleParentsInEachGeneration** must be provided only if **PerformCrossesConsideringTheGenderYesNo** is *Yes*.

NumberOfMaleParentsInEachGeneration and **NumberOfFemaleParentsInEachGeneration** determine the number of male and female plants, respectively, that are inter-crossed in a given generation to produce the next generation. The format of these parameters is the same as for **NumberOfSiresInEachGeneration**, i.e., a character string, *Constant* or *Variable*, followed by 1 or *nGen+nGenSel-1* values indicating the number of plants to be used as parents in each generation.

Note: For a given generation *g*, **NumberOfMaleParentsInEachGeneration** and **NumberOfFemaleParentsInEachGeneration** must be smaller or equal to *nIndivInThisGen(j)/2*.

NumberOfParentsRegardlessTheGenderInEachGen must be provided only if **PerformCrossesConsideringTheGenderYesNo** is *No*.

PerformSelfingInAtLeastOneGenerationYesNo can be *Yes* or *No*. The value must be *Yes* if the breeding program includes at least one generation of selfing.

NumberOfSelfedParentsInEachGeneration determines the number of plants that are selfed in each generation to produce the next generation. The format of this parameter is the same as for **NumberOfSiresInEachGeneration**, i.e., a character string, *Constant* or *Variable*, followed by 1 or *nGen+nGenSel-1* values indicating the number of plants to be selfed in each generation.

Note: Let *nSelfParInThisGen(j)* be the number of plants to be selfed from generation *g*. If one wants to self plants from generation *g* and inter-cross their progeny, i.e., generation *g+1*, *nSelfParInThisGen(j)* must be set to the desired number of plants to be selfed while *nSelfParInThisGen(j+1)* must be zero. A non-null value for *nSelfParInThisGen(j+1)* will result in selfing whatever the values specified for the parameters **NumberOfMaleParentsInEachGeneration** and **NumberOfFemaleParentsInEachGeneration** in generation *g+1*.

NumberOfGenerationsIncludingDoubledHaploids must be an integer comprised between 0 to *nGen+nGenSel*. This parameter determines the number of generations in which double-haploids (DH) must be created. The generations in which DH are created are specified by the parameter **CreateDoubledHaploidsInTheseGenerations**.

CreateDoubledHaploidsInTheseGenerations must be a vector of length equal to **NumberOfGenerationsIncludingDoubledHaploids**, i.e., the number of generations in which DH must be created. Each element of the vector must include an integer identifying a generation in which DH must be created.

AlphaSim creates DH by selecting one haplotype from a male plant and copying it after recombination and insertion of eventual mutations. The number of plants that are used to create DH in generation *g* is specified by the value of **NumberOfMaleParentsInEachGeneration** for generation *g-1*, with **PerformCrossesConsideringTheGenderYesNo** set on *Yes*. If *g* is 1, DH are created in the base generation using the founders haplotypes. Note that when creating DH in generation *g* the value of **NumberOfFemaleParentsInEachGeneration** for generation *g-1* does not matter and can be set to 0.

Box 4: Flexibility

Three flexibility options have been integrated into AlphaSim. The first two allow AlphaSim to be run sequentially, i.e., running AlphaSim until a user-defined stop and then restarting AlphaSim. In between, the user can introduce some changes to the parameters file. The third option enables the user to generate a dataset across different AlphaSim runs, and then continue running AlphaSim using this merged dataset.

(1) First flexibility option: GenerationFlexibility

The first flexibility option – **GenerationFlexibility** – allows running AlphaSim from a given start generation to a given stop generation, instead of running AlphaSim from the very first generation to the very last one.

GenerationFlexibilityOnOff can be *On* or *Off*. The value must be *Off* when using an external pedigree.

If **GenerationFlexibilityOnOff** is *On*, start and stop generations must be provided by setting the **StartStopGeneration** parameter to *StartGeneration*, *StopGeneration*. *StartGeneration* and *StopGeneration* must be comprised range between 1 and *nGen+nGenSel*, with $StartGeneration \leq StopGeneration$. When starting a simulation, *StartGeneration* must be 1. Indeed, the base information is simulated during the first generation of the breeding program: pedigree and genomic data are generated, segregating sites are extracted, SNP chips and QTN arrays are created, and QTN effects are simulated. This information is stored in distinct files, which will be thereafter read to generate the next generations. Set *StartStopGeneration* to *“g,g”* to run generation *g* only.

Note:

- **GenerationFlexibilityOnOff can be used to:**
  - simulate several pedigrees all derived from a common base generation, i.e., when restarting AlphaSim from generation g, the number of generations, the number of individuals per generation (Box 1), and the number of individuals to be selected (Box 2 or Box 3) can be modified for generation *g* to the last generation of the pedigree.
  - modify the value of a parameter from a generation to another, e.g., the selection strategy;
  - integrate external generations and internally simulated generations (Example 1);
  - add selection generations to an external pedigree (Example 2);
  - import externally created selection generations together with the option **SelectionFlexibilityOnOff** to (Example 3);
  - . . .
- Once the last generation of the pedigree has been simulated, the Chromosomes directory (see Chromosomes) is by default removed. To restart the simulation process from a given generation *g*, with *g* > 1, using the data simulated from generations 1 to *g-1*, **KeepChromosomesFolderYesNo** (Box 14) must be set to *Yes*.

EXAMPLE 1: CONSTRUCTING A PEDIGREE INTEGRATING BOTH EXTERNAL GENERATIONS AND INTERNALLY SIMULATED GENERATIONS

Consider the construction of a pedigree including 4 burn-in generations that comprise 10, 20, 100 and 200 individuals, respectively. Generations 2 to 4 are derived from specific crosses and therefore will be imported under the form of an external pedigree. This is achieved in two steps:

1. Run AlphaSim using the following parameters to simulate generation 1:

```
###############################################################
BOX 1 - PEDIGREE
PedigreeStatus                                ,Internal
NumberGenerationsBurnIn                       ,2
NumberGenerationsSelection                    ,0
NumberOfIndividualsInEachGeneration           ,Variable,10,320
###############################################################
BOX 3 - PLANT BREEDING OPTIONS
PlantBreedingOnOff                            ,On
PerformCrossesConsideringTheGenderYesNo       ,Yes
NumberOfMaleParentsInEachGeneration           ,Constant,0
NumberOfFemaleParentsInEachGeneration         ,Constant,0
NumberOfParentsRegardlessTheGenderInEachGen   ,
PerformSelfingInAtLeastOneGenerationYesNo     ,No
NumberOfSelfedParentsInEachGeneration         ,Constant,0
NumberOfGenerationsIncludingDoubledHaploids   ,0
CreateDoubledHaploidsInTheseGenerations       ,
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                    ,On
StartStopGeneration                           ,1,1
SelectionFlexibilityOnOff                     ,Off
ImportAnExternalPedigreeForThisGeneration     ,
ImportStagePut1ToComputeBV2ToContinue         ,
###############################################################
```

Note that:

- AlphaSim reads the external pedigree as one unique generation. Therefore, **NumberGenerationsBurnIn** is set to 2 instead of 4, the first generation corresponding to the generation that is being simulated, and the second one corresponding to the ones that are being imported. Similarly, **NumberOfIndividualsInEachGeneration** is set to *Variable,10,320*, with 320=20+100+200.

- The numbers of male and female parents to be selected in the first generation do not matter, since the selection of parents is performed according to the external pedigree. Here, we set the numbers of parents to 0.

- The Box 3 remains unchanged across the simulating process and is therefore not shown in the next step.

2. Import the pedigree of generations 2 to 4, including 320 (=20+100+200) individuals in total, using the following parameters:

```
###############################################################
BOX 1 - PEDIGREE
PedigreeStatus                                ,PedigreeGen2to4.txt
NumberGenerationsBurnIn                       ,2
NumberGenerationsSelection                    ,0
NumberOfIndividualsInEachGeneration           ,Variable,10,320
###############################################################
BOX 4 - FLEXIBILITY
```

```
GenerationFlexibilityOnOff                    ,On
StartStopGeneration                           ,2,2
SelectionFlexibilityOnOff                     ,Off
ImportAnExternalPedigreeForThisGeneration     ,
ImportStagePut1ToComputeBV2ToContinue         ,
###############################################################
```

EXAMPLE 2: ADDING SELECTION GENERATIONS TO AN EXTERNAL PEDIGREE

Suppose that we want to add two selection generations to the pedigree constructed in Example 1. Each selection generation includes 50 individuals derived from crossing 10 male parents and 10 female parents. The first selection generation derives from parents selected in the last generation of the external pedigree (generation 4).

In order to perform selection in the last generation of the external pedigree, the external pedigree is imported in two distinct files, one including generations 2 to 3 (120=20+100 individuals), and one including generation 4 (200 individuals), in which selection is performed. This is achieved in four steps:

1. Run AlphaSim using the following parameters to simulate generation 1:

```
###############################################################
BOX 1 - PEDIGREE
PedigreeStatus                                ,Internal
NumberGenerationsBurnIn                       ,3
NumberGenerationsSelection                    ,2
NumberOfIndividualsInEachGeneration           ,Variable,10,120,200,50,50
###############################################################
BOX 3 - PLANT BREEDING OPTIONS
PlantBreedingOnOff                            ,On
PerformCrossesConsideringTheGenderYesNo       ,Yes
NumberOfMaleParentsInEachGeneration           ,Variable,0,0,10,10
NumberOfFemaleParentsInEachGeneration         ,Variable,0,0,10,10
NumberOfParentsRegardlessTheGenderInEachGen   ,
PerformSelfingInAtLeastOneGenerationYesNo     ,No
NumberOfSelfedParentsInEachGeneration         ,Constant,0
NumberOfGenerationsIncludingDoubledHaploids   ,0
CreateDoubledHaploidsInTheseGenerations       ,
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                    ,On
StartStopGeneration                           ,1,1
SelectionFlexibilityOnOff                     ,Off
ImportAnExternalPedigreeForThisGeneration     ,
ImportStagePut1ToComputeBV2ToContinue         ,
###############################################################
```

Note that the Box 3 remains unchanged across the simulating process and is therefore not shown in the next steps.

2. Import the pedigree of generations 2 to 3, including 20+120 individuals in total:

```
###############################################################
BOX 1 - PEDIGREE
PedigreeStatus                                ,PedigreeGen2to3.txt
NumberGenerationsBurnIn                       ,3
NumberGenerationsSelection                    ,2
NumberOfIndividualsInEachGeneration           ,Variable,10,120,200,50,50
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                    ,On
StartStopGeneration                           ,2,2
SelectionFlexibilityOnOff                     ,Off
```

```
ImportAnExternalPedigreeForThisGeneration     ,
ImportStagePut1ToComputeBV2ToContinue         ,
###############################################################
```

3. Import the pedigree of generation 4, including 200 individuals:

```
###############################################################
BOX 1 - PEDIGREE
PedigreeStatus                                ,PedigreeGen4.txt
NumberGenerationsBurnIn                       ,3
NumberGenerationsSelection                    ,2
NumberOfIndividualsInEachGeneration           ,Variable,10,120,200,50,50
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                    ,On
StartStopGeneration                           ,3,3
SelectionFlexibilityOnOff                     ,Off
ImportAnExternalPedigreeForThisGeneration     ,
ImportStagePut1ToComputeBV2ToContinue         ,
###############################################################
```

4. Simulate two additional selection generations including each 50 individuals:

```
###############################################################
BOX 1 - PEDIGREE
PedigreeStatus                                ,Internal
NumberGenerationsBurnIn                       ,3
NumberGenerationsSelection                    ,2
NumberOfIndividualsInEachGeneration           ,Variable,10,220,100,50,50
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                    ,On
StartStopGeneration                           ,4,5
SelectionFlexibilityOnOff                     ,Off
ImportAnExternalPedigreeForThisGeneration     ,
ImportStagePut1ToComputeBV2ToContinue         ,
###############################################################
```

(2) Second flexibility option: SelectionFlexibility

The second flexibility option – **SelectionFlexibility** – allows importing an external pedigree for a given selection generation while working with a general internal pedigree (i.e., when **PedigreeStatus** is *Internal*). This flexibility option allows the user to provide a particular user-defined pedigree structure to one or several selection generations based on simulated or inferred values prior to this point (TBV, gEBV, phenotypes or genotypes). In practice, the simulating process is temporally stopped, then the user makes the selection and mating decisions outside the program, and the externally created pedigree is finally imported before continuing the simulating process.

SelectionFlexibilityOnOff is *On* or *Off*. If *On*, values must be provided for the following two parameters.

ImportAnExternalPedigreeForThisGeneration is the identifier of the selection generation for which an external pedigree needs to be imported. The value of this parameter can range from nGen+1 to nGen+nGenSel.

ImportationStagePut1ToComputeBV2ToContinue specifies the import stage of the external pedigree. This parameter can be either 1 or 2. The value 1 makes AlphaSim run estimation of breeding values for the individuals among which selection and mating have to be performed by the user, by providing an external pedigree. The value 2 allows importing such a pedigree and AlphaSim continues with the simulation.

EXAMPLE 3: IMPORT OF AN EXTERNALLY CREATED SELECTION GENERATION

Consider a pedigree including two burn-in generations and four selection generations, so that the total number of generations is 6. We want to import external pedigrees for generations 3 and 5, i.e., the first and third selection cycles.

This is achieved in four steps:

1. Run AlphaSim using the following parameters file to simulate generations 1 and 2 and compute TBV and gEBV for generation 3:

   ```
   ################################################################
   BOX 4 - FLEXIBILITY
   GenerationFlexibilityOnOff                         ,On
   StartStopGeneration                                ,1,3
   SelectionFlexibilityOnOff                          ,On
   ImportAnExternalPedigreeForThisGeneration          ,3
   ImportationStagePut1ToComputeBV2ToContinue         ,1
   ################################################################
   ```

2. Create the external pedigree for generation 3, based on the TBV, gEBV, phenotypes or genotypes of the individuals from generation 2. Name the file *ExternalPedigree.txt* and copy it in Selection/SelectionFolder1/, which corresponds to generation 3, and then restart AlphaSim using the flexibility box filled in as follows to import the provided pedigree:

   ```
   ################################################################
   BOX 4 - FLEXIBILITY
   GenerationFlexibilityOnOff                         ,On
   StartStopGeneration                                ,3,3
   SelectionFlexibilityOnOff                          ,On
   ImportAnExternalPedigreeForThisGeneration          ,3
   ImportationStagePut1ToComputeBV2ToContinue         ,2
   ################################################################
   ```

3. Now we have to proceed with simulation up to the generation 5. For that purpose, run AlphaSim using the following parameter file:

   ```
   ################################################################
   BOX 4 - FLEXIBILITY
   GenerationFlexibilityOnOff                         ,On
   StartStopGeneration                                ,4,5
   SelectionFlexibilityOnOff                          ,On
   ImportAnExternalPedigreeForThisGeneration          ,5
   ImportationStagePut1ToComputeBV2ToContinue         ,1
   ################################################################
   ```

4. Copy the external pedigree for generation 5 in SelectionFolder3, which corresponds to the third selection cycle or generation 5, and then restart AlphaSim using the Flexibility box filled in as follows:

   ```
   ################################################################
   BOX 4 - FLEXIBILITY
   GenerationFlexibilityOnOff                         ,On
   StartStopGeneration                                ,5,6
   SelectionFlexibilityOnOff                          ,On
   ImportAnExternalPedigreeForThisGeneration          ,5
   ImportationStagePut1ToComputeBV2ToContinue         ,2
   ################################################################
   ```

Details about the format of the external pedigree to be imported are given in section PedigreeFiles.

(3) Third flexibility option: Merge

The third flexibility option - **MergeTypes** - enables the use to combine information across different AlphaSim runs. The merge parameters are specified in Box 4 as the last three parameters. When running AlphaSim normally, the merge parameters would be switched off as below:

```
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                      ,On
StartStopGeneration                             ,5,6
SelectionFlexibilityOnOff                       ,On
ImportAnExternalPedigreeForThisGeneration       ,5
ImportationStagePut1ToComputeBV2ToContinue      ,2
MergeTypes                                      ,none
NumberOfMergeRuns                               ,0
MergeFiles                                      ,/path/
###############################################################
```

In order to utilise the merge function, the selection generations must all derive from a common BurnIn generation, in order to ensure consistency in the parameters simulated in the BurnIn generations (such as position of segregating sites, QTN effect sizes in the base generation, etc.). Simulations should be run in the following format:

First, the user must generate a number of BurnIn generations, with parameters set as shown below. The output of this run must be saved in a BurnIn directory. In the example below, only a single BurnIn generation was simulated:

```
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                      ,On
StartStopGeneration                             ,1,1
SelectionFlexibilityOnOff                       ,Off
ImportAnExternalPedigreeForThisGeneration       ,5
ImportationStagePut1ToComputeBV2ToContinue      ,2
MergeTypes                                      ,none
NumberOfMergeRuns                               ,0
MergeFiles                                      ,/path/
###############################################################
```

The user can then generate a number of AlphaSim selection runs derived from this BurnIn generation. Each individual run must be placed in a directory of its own. In the example below, five Selection generations were simulated:

```
   ###############################################################
   BOX 4 - FLEXIBILITY
   GenerationFlexibilityOnOff                      ,On
   StartStopGeneration                             ,2,6
   SelectionFlexibilityOnOff                       ,Off
   ImportAnExternalPedigreeForThisGeneration       ,5
   ImportationStagePut1ToComputeBV2ToContinue      ,2
   MergeTypes                                      ,none
   NumberOfMergeRuns                               ,0
   MergeFiles                                      ,/path/
   ###############################################################
```

```
Once a number of selection runs have been generated, the user can merge information across runs.
This can be done in two ways::

*(a)    Merge all individuals across all runs*
```

To implement this option, the user must specify the following merge parameters:

```
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                      ,On
StartStopGeneration                             ,2,6
SelectionFlexibilityOnOff                       ,Off
ImportAnExternalPedigreeForThisGeneration       ,5
```

```
ImportationStagePut1ToComputeBV2ToContinue      ,2
MergeTypes                                      ,Directories
NumberOfMergeRuns                               ,4
MergeFiles                                      ,/absolute/path/to/file
###############################################################
```

The **MergeTypes** parameter must be set to **Directories**, with the **NumberOfMergeRuns** parameter reflecting the number of derived runs (not including the BurnIn). The **MergeFiles** parameter is the absolute path to a file containing a list of absolute paths of runs to merge, with one path per line. The first path is always the BurnIn directory; subsequent path orders are not important, however, AlphaSim will merge across directories in the order specified in the file.

Following on from this, AlphaSim can be re-started with merge parameters disabled from the last generation of the merge across runs. For example, if all runs were simulated until generation 6, then AlphaSim would be restarted from generation 7, as shown below:

```
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                      ,On
StartStopGeneration                             ,7,10
SelectionFlexibilityOnOff                       ,Off
ImportAnExternalPedigreeForThisGeneration       ,5
ImportationStagePut1ToComputeBV2ToContinue      ,2
MergeTypes                                      ,none
NumberOfMergeRuns                               ,0
MergeFiles                                      ,/path/
###############################################################
```

If Selection is to be performed with the **SelectionMethod** set to **GBLUP** and the **SelectionPhenotypingStrategy** set to **RandomPhenotypes**, then the user must specify the **FirstAndLastTrainingGenForRandomPhenotypesStrat** as the generation prior to merge. In the example above, this would be generation 6. In addtion, the **FixTheTrainingDataAcrossGenerationsOnOff** must be switched off, and the **UseTrainingDataFromGeneration** option must also be set to 6:

```
###############################################################
BOX 10 - SELECTION
SelectionBasedOnRestrictedOrUnrestrictedQtn      ,Unrestricted
DistributionOfQtnOfSelectionTraits               ,Normal
SelectionMethod                                  ,GBLUP
AddQtnToTheSnpChipYesNo                          ,No
UseExternalSnpSolutionsYesNo                     ,No
SelectionPhenotypingStrategy                     ,RandomPhenotypes
FirstAndLastTrainingGenForRandomPhenotypesStrat  ,6,6
TrainingSetSizeForRandomPhenotypesStrat          ,5
FixTheTrainingDataAcrossGenerationsOnOff         ,Off
UseTrainingDataFromGeneration                    ,6
ComputeAverageCoancestryYesNo                    ,No
###############################################################
```

```
*(b)    User specified IDs within a run*
```

This option allows the user to perform a-priori selection of individuals across runs, and to generate a new instance of AlphaSim with these selected individuals. These user-selected individuals are set as the base population for subsequent AlphaSim runs. To use this option, the merge parameters must be specfied as follows:

```
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                      ,On
StartStopGeneration                             ,7,10
SelectionFlexibilityOnOff                       ,Off
```

```
ImportAnExternalPedigreeForThisGeneration       ,5
ImportationStagePut1ToComputeBV2ToContinue      ,2
MergeTypes                                      ,Ids
NumberOfMergeRuns                               ,4
MergeFiles                                      ,/absolute/path/to/file
###############################################################
```

In this case, the **MergeTypes** parameter must be set to **Ids**, with the **NumberOfMergeRuns** parameter again reflecting the number of derived runs (not including the BurnIn). The **MergeFiles** parameter is the absolute path to a file containing information for Ids to merge. All individuals within the same directory must appear consecutively within the file. An example file is given below:

```
/path/to/BurnInFolder
ID_1                        ,/absolute/path/to/run/directory/1
ID_2                        ,/absolute/path/to/run/directory/1
ID_3                        ,/absolute/path/to/run/directory/2
ID_4                        ,/absolute/path/to/run/directory/2
etc...
```

The user can then restart AlphaSim from generation 2 using the **GenerationFlexibilityOnOff** and **StartStopGeneration** parameters, assuming these new individuals are the base generation. All merge parameters must be switched off in consecutive runs post-merge, as shown below:

```
###############################################################
BOX 4 - FLEXIBILITY
GenerationFlexibilityOnOff                      ,On
StartStopGeneration                             ,2,10
SelectionFlexibilityOnOff                       ,Off
ImportAnExternalPedigreeForThisGeneration       ,5
ImportationStagePut1ToComputeBV2ToContinue      ,2
MergeTypes                                      ,none
NumberOfMergeRuns                               ,0
MergeFiles                                      ,/path/
###############################################################
```

Box 5: Chromosomes

NumberOfChromosomes determines the number of chromosomes (*nChrom*) that must be simulated.

ChromosomeLengthMorgans determines the genetic length of chromosomes in Morgans. A chromosome of length 1 has a probability of 100% of having a recombination per chromosome, and thus, on average, one recombination per chromosome.

During the initialisation of the simulation, AlphaSim makes a call to the software MaCS, which generates a set of founder sequences from which a set of base haplotypes are sampled. The number of haplotypes to sample to generate the base population are specified by the **NumberOfHaplotypes*** parameter.

In addition, AlphaSim enables the user to specify the simulation of sequence data using MaCS for a number of domesticated species. These include: cattle, rabbit, maize, maize landrace and wheat. These are specified using the **ChromosomeSource** parameter as: InternalCattle, InternalRabbit, InternalMaize, InternalMaizeLandrace and InternalWheat, respectively.

Alternatively, the user can provide an absolute path to a file containing parameters to run MaCs. In this case, the **ChromosomeSource** parameter is an absolute path to a file containing parameters for simulating sequence data in MaCS. For further description of the required fields in this file, see section macsParams.txt.

Box 6: SNP chips

UseSnpChipOnOff can be *On* or *Off*. If **UseSnpChipOnOff** is *Off* then all of the segregating sites in base haplotypes are put onto one single SNP chip. If **UseSnpChipOnOff** is *On*, values must be provided for the parameters included in Box 6.

NumberOfSnpChips determines the number of SNP chips (*nChip*) that are created.

nSnpPerChipPerChromosome determines, for each SNP chip, the number of SNP that will be put onto a chip from each chromosome. *nChip* values must be provided for *nSnpPerChipPerChromosome*, so that each chip e includes a total of $nSnpPerChipPerChromosome \times nChrom$ SNP.

MinMaxAlleleFreqForSnp controls the minor allele frequency (MAF) of the SNP included in the chips. Two numbers ranging between 0 and 0.5 must be provided, accounting for the minimum and maximum MAF of the SNP included in the chips. This restriction is applied the same way on all SNP chips.

SnpChipIncludesQtnYesNoRandom can be *Yes*, *No* or *Random*. If *Yes*, the QTN are included in the SNP chips. On the opposite, if **SnpChipIncludesQtnYesNoRandom** is *No*, QTN are not on the SNP chips, i.e., all QTN and SNP markers are different from each other. If *Random*, QTN might be on the SNP chip(s), or not, i.e., the user does not control the inclusion of QTN on the SNP chip(s).

ChipsAreNestedYesNo and **IdOfChipUsedForSelection** must be provided only if there is more than one chip.

ChipsAreNestedYesNo can be either *Yes* or *No*. Setting **ChipsAreNestedYesNo** to *Yes* allows nesting the different chips within each other, i.e., largest SNP chips include all of the SNP present on smaller chips. If **ChipsAreNestedYesNo** is *No*, the SNP are independently selected for each chip.

Note: SNP chips must be nested to allow controlling the inclusion, or exclusion, of QTN in the SNP chips. Therefore, **ChipsAreNestedYesNo** must be *Yes* if **SnpChipIncludesQtnYesNoRandom** is *Yes* or *No*, while **ChipsAreNestedYesNo** can be either *Yes* or *No* if **SnpChipIncludesQtnYesNoRandom** is *Random*.

IdOfChipUsedForSelection determines the SNP chip that will be used to provide the genomic information required to compute genomic-estimated breeding values and perform selection.

Box 7: quantitative trait loci (QTN)

AlphaSim basically samples QTN from the available segregating sites and simulates QTN effects assuming an additive genetic model or a genetic model including both dominance and additive effects.

There are two sets of QTN that are sampled, both sets including a same user-specified number of QTN. The first set, referred to as *unrestricted*, is comprised of candidate QTN selected at random from across the genome. The second set, referred to as *restricted*, is comprised of candidate QTN selected at random from across the genome with the restriction that the minor allele frequency must be comprised in a specified range. The restrictions in allele frequency of both SNP markers and QTN allow the user to manage the possibility that QTN have different average minor allele frequencies than SNP.

In addition, AlphaSim simulates QTN effects that can display one of two types of distribution, *normal* or *Gamma*.

In total, AlphaSim distinguishes four QTN models:

Model 1: Unrestricted QTN with normally-distributed effects

Model 2: Restricted QTN with normally-distributed effects

Model 3: Unrestricted QTN with Gamma-distributed effects

Model 4: Restricted QTN with Gamma-distributed effects

NumberOfQTNperChromosomeNormalDist determines the number of QTN with normally-distributed effects that will be sampled from each chromosome.

Similarly, **NumberOfQTNperChromosomeGammaDist** determines the number of QTN with Gamma-distributed effects that will be sampled from each chromosome. If **NumberOfQTNperChromosomeGammaDist** is greater than 0, values must be provided for the parameter **ShapeAndScaleOfGammaDistForModel3And4**.

Note: **NumberOfQTNperChromosomeNormalDist** or **NumberOfQTNperChromosomeGammaDist** must be greater than 0.

NumberOfMutationsPerChr indicates the number of de-novo causal mutations (*nMut*) that will be inserted in each chromosome. Each mutation corresponds to a single site at which all ancestors and contemporaries having the wild type allele, while one individual will have a mutated allele on one of its haplotypes. This mutation will be transmitted to the next generation according to the Mendelian sampling. All of the mutation sites are considered to be QTN, thereby increasing the total number of QTN by $nMut \times nChrom$.

If **NumberOfMutationsPerChr** is greater than zero, the parameter **MutationalHeritability** must be provided.

MutationalHeritability determines the genetic variance due to mutations.

MinMaxAlleleFreqForModel2And4 controls the minor allele frequency (MAF) of the “restricted” QTN (QTN models 2 and 4). Two numbers ranging between 0 and 0.5 must be provided, accounting for the minimum and maximum MAF of the QTN.

ShapeAndScaleOfGammaDistForModel3And4 determines the shape and scale of the Gamma distribution of QTN effects, which is used in models 3 and 4.

IncludeDominanceEffectsOnOff can be either *On* or *Off*. If *On*, AlphaSim simulates QTN effects assuming dominance genetic model. In that case, values must be provided for the parameter **DominanceDegreeMeanAndVariance**.

DominanceDegreeMeanAndVariance must include two real numbers which determine the mean and variance of the normal distribution from which dominance degrees are sampled.

Note: HOW ALPHASIM SIMULATES ADDITIVE, DOMINANCE AND AVERAGE ALLELE SUBSITUTION EFFECTS

- **Additive effects:** Let *k* indicate a QTN from one of both sets of QTN, unrestricted or frequency-restricted. For each QTN *k*, AlphaSim randomly samples normally-distributed deviates from a normal distribution with zero mean and standard deviation of one unit, and Gamma-distributed deviates from a Gamma distribution with user-specified shape and scale parameters and a 50% chance of being positive or negative. Additive effects are obtained as follows:

  $$a_k = RanDev\sqrt{\frac{var_A}{n_{QTN}}}$$

  where *RandDev* is a random deviate sampled from a normal or Gamma distribution, and var_A is the a priori genetic variance specified using the parameter **TraitVariances** (Box 12).

- **Dominance effects:** According to Wellmann and Bennewitz (2012, 2011), dominance degrees, δ_k, are sampled from a normal distribution with mean m_δ and variance var_δ specified using the parameter **DominanceDegreeMeanAndVariance**:

  $$\delta_k = m_\delta + RandDev\sqrt{var_\delta}$$

  where *RandDev* is a random deviate sampled from a normal distribution with zero mean and standard deviation of one unit. The dominance effects are then:

  $$d_k = \delta_k \cdot |a_k|.$$

- **Computation of average allele substitution effects:** Average allele substitution effects α_k are finally computed for each QTN according to Falconer and Mackay (1996):

  $$\alpha_k = a_k + d_k(q_{k_0} - p_{k_0})$$

where p_{k_0} and q_{k_0} are the frequencies of the non-zero and zero alleles, respectively, at QTN *k* in the base generation of the pedigree.

Box 8: Recombination

AlphaSim allows simulating QTN that affect the recombination rate. This is performed assuming an additive genetic model.

RecombinationOnOff can be *On* or *Off*. Note that **RecombinationSelOnOff** must be *On* from the first burn-in generation when using the flexibility option while selecting for recombination. If **RecombinationOnOff** is *On*, then Box 8 must be filled in as follows.

HeritabilityOfRecombination determines the heritability of recombination rate.

NumberOfQTNperChromForRecombination determines the number of QTN that will be sampled in each chromosome for the "recombination" trait.

When selecting for the recombination rate, selection is performed using a selection index that integrates the breeding values computed for both the recombination trait and the common trait:

$$SelectionIndex_i = \frac{BV_i}{\sqrt{var_A}}(1-\omega) + \frac{BVrecomb_i}{\sqrt{var_{A_{recomb}}}}\omega$$

where BV_i and $BVrecomb_i$ are the breeding values of individual i, var_A and $var_{A_{recomb}}$ are the additive variances, computed using the breeding values of the individuals included in the generation of individual i for the common and recombination traits, respectively, and ω is the weight of the recombination trait. The breeding values are gEBV or TBV according to the selection method, GBLUP or TBV, respectively.

Two distinct weights can be provided for the recombination trait. The first one, called **InitialRecombinationIndexWeight**, is used to compute the index during the *n* first generations of selection, where *n* is determined by **NumberGenerationsSelectionOnRecombination**. The second weight, called **LaterGenRecombinationIndexWeight**, is used to compute the index in the following generations, i.e., from generation *nGen+n+1* to generation *nGen+nGenSel*.

DistributionOfQtnForRecombination determines the distribution of the QTN effects for the recombination trait. The distribution can be either *Normal*, or *Gamma*.

ShapeAndScaleOfGammaDist determines the shape and scale of the Gamma distribution.

Box 9: Gene editing

AlphaSim allows editing genome of a specified number of selected male parents. This occurs as follows. Firstly, the male individuals are ranked in descending order of breeding values. Secondly, *n* individuals are selected from the first ranked individual, where *n* has been specified by the parameter **NumberOfSiresInEachGeneration**. Gene editing can be performed at this stage by selecting n_{edited} individuals among the *n* selected individuals in the generation. There are two options to select the individuals to be edited: either editing the best-performing ones, or editing the least-performing ones. The QTN to be edited are by default selected in descending order of magnitude of their effect, i.e., the QTN with large effect in absolute value are preferentially edited. Alternatively, the user can provide its own selection of QTN to be edited in a file that will be read in. Finally, gene editing is performed so that each of the n_{edited} individuals bears the favourable allele in a homozygous state at that QTN, i.e., 1/1 if the effect of the non-zero allele is positive, or 0/0 if the effect of the non-zero allele is negative.

GeneEditingOnOff can be *On* or *Off*. Note that **GeneEditingOnOff** must be *On* from the first burn-in generation when using the flexibility option while performing gene editing in at least one selection generation. Gene editing is not allowed when selecting by optimal contribution (see below), i.e., **OptimalContributionSelectionOnOff** must be *Off* to perform gene editing.

If **GeneEditingOnOff** is On, Box 9 must be filled in as follows:

EditTopOrBottom is *Top* or *Bottom*. This parameter allows editing the best-performing (*Top*) or least-performing (*Bottom*) individuals among the selected individuals.

NumberOfEditedIndividuals determines the number of individuals to be edited. **NumberOfEditedIndividuals** must be smaller or equal to the number of individuals that is selected in the generation(s) in which gene editing is performed (the number of selected sires in each generation being specified by NumberOfSiresInEachGeneration).

NumberOfGeneEditsPerIndividual determines the number of QTN to be edited, or edits, to be performed in each individual.

PerformGEOnUserQtnSelectionYesNo allows the user to provide his own selection of QTN to be edited. The value of **PerformGEOnUserQtnSelectionYesNo** must be *Yes* or *No*. If *Yes*, a file including the QTN to be edited must be provided in the main directory, i.e., the directory from which the AlphaSim program is run. Details about the format are given in section UserSelectedQtnToBeEdited.txt.

Note that if **PlantBreedingOnOff** is *On*, gene editing can also be performed in selfed individuals when performing selfing, or in the selected individuals regardless their gender when performing crosses without considering the gender.

Box 10: Selection

AlphaSim enables performing selection by truncation or by optimal contribution (OC). By default, selection is performed by truncation, i.e., the best performing individuals are selected.

Truncation selection proceeds as follows. For a given selection generation *g*, the best performing individuals in generation *g-1* are selected based on their breeding values, TBV or gEBV, or on their phenotypic values. The numbers of individuals to be selected are determined by *nSiresInThisGeneration(j-1)* and *nDamInThisGeneration(j-1)* in the case of animal breeding (Box 2), and by *NumberOfMaleParentsInEachGeneration(j-1)* and *NumberOfFemaleParentsInEachGeneration(j-1)*, *NumberOfParentsRegardlessTheGenderInEachGen(j-1)*, or *NumberOfSelfedParentsInEachGeneration(j-1)* in the case of plant breeding (Box 3). The number of progenies per parent is on average equal to the number of individuals in generation *g*, as specified by *NumberOfIndividualsInEachGeneration* in Box 1, divided by the numbers of parents in generation *g-1*.

OC selection is activated by setting **OptimalContributionSelectionOnOff** to *On* in Box 11. The method used to perform OC selection is described in Box 11.

Note: The parameters detailed in Box 10 are general parameters relative to selection; values must be provided regardless of the type of selection that is performed, truncation or optimal contribution.

SelectionBasedOnRestrictedOrUnrestrictedQtn and **DistributionOfQtnOfSelectionTraits** determine the QTN model, and thus the QTN effects and TBV, which will be used to compute the additive variances and phenotypic values during the selection process, and to sort the QTN according to their effect in the scope of gene editing. **SelectionBasedOnRestrictedOrUnrestrictedQtn** can be either *Unrestricted*, or *Restricted*. **DistributionOfQtnOfSelectionTraits** can be either *Normal* or *Gamma*.

SelectionMethod determines the method used to provide the breeding values that are used for selection. It can be *TBV*, *Pheno* or *GBLUP*. If **SelectionMethod** is *TBV*, selection is performed using the true breeding values; if **SelectionMethod** is *Pheno*, selection is performed using the phenotypic values; and if **SelectionMethod** is *GBLUP*, selection is performed using genomic-estimated breeding values (gEBV).

Note: HOW ALPHASIM GENERATES TBV, PHENOTYPES AND gEBV

- **TBV:** TBV are computed for each individual *i* using the allele substitution effects α_k simulated in each set of

QTN, unrestricted and frequency-restricted, according to a normal or Gamma trait distribution:

$$TBV_i = \sum_{k=1}^{n_{QTN}} (2q_{k_0}\alpha_k, (q_{k_0} - p_{k_0})\alpha_k, -2p_{k_0}\alpha_k)[x_{i,k} = 0, x_{i,k} = 1, x_{i,k} = 2]$$

where p_{k_0} and q_{k_0} are the frequencies of the non-zero and zero alleles at QTN *k* in the base generation of the pedigree, and $x_{i,k}$ is the genotype of individual *i* at QTN *k*, which is coded as 0, 1 or 2 according to the number of copies of the non-zero allele (Falconer and Mackay, 1996).

- **Phenotypes:** Phenotypes (or TPV, true phenotypic value) are obtained as follows. First, true dominance values (TDV) are computed for each individual using the dominance effects d_k:

  $$TDV_i = \sum_{k=1}^{n_{QTN}} (-2q_{k_0}^2 d_k, 2q_{k_0}p_{k_0})d_k, -2p_{k_0}^2 d_k)[x_{i,k} = 0, x_{i,k} = 1, x_{i,k} = 2]$$

  If **IncludeDominanceEffectsOnOff** is set to *Off* (i.e., dominance effects are ignored), TDV are equal to 0. Second, residual effects are computed according to the trait heritability provided by the user. To ensure that the trait heritability remains at the specified value, the residual variance var_E is scaled relative to the variance of the TBV and TDV in the base generation of the pedigree. Residual effects are then sampled from a normal distribution with zero mean and variance var_E. Finally, TPV are generated by adding a residual effect e_i to the TBV and TDV:

  $$TPV_i = TBV_i + TDV_i + e_i.$$

  Four distinct TPV are computed for each individual, each of them corresponding to a QTN model (unrestricted or frequency-restricted set of QTN, and normal or Gamma trait distribution).

- **gEBV:** The computation of gEBV involves a "training" dataset and a "test" dataset in which a particular SNP chip is used.

  - SNP chip: If **UseSnpChipOnOff** is *On*, the SNP chip used for selection is specified by the parameter **IdOfChipUsedForSelection** in Box 6. Otherwise, all segregating sites are used.
  - Training dataset: the training dataset includes individuals that are characterized by both phenotypes and SNP genotypes. It can take different forms, which are specified using the parameter **SelectionPhenotypingStrategy** (see below).
  - Test dataset: for a given generation *g*, the test dataset includes all the individuals of generation *g-1*, among which parents are selected. In contrast to the training individuals, the testing individuals are characterized by their SNP genotypes only.

  The computation of gEBV is as follows. The phenotypes of the training individuals are used to train a Ridge regression model (Hoerl and Kennard, 1976; Meuwissen et al., 2001; Whittaker et al., 2000):

  $$y_i \sim N(\mu_i, var_E)$$

  $$\mu_i = \mu_0 + \sum_{j=1}^{n_{SNP}} m_j x_{i,j}$$

  $$m_j \sim N(0, var_m)$$

  where y_i is the phenotype of individual *i*, μ_0 is the intercept, m_j is the effect of SNP *j*, $x_{i,j}$ is the genotype of individual *i* at SNP *j*, and var_m is the variance of SNP effects. The Ridge regression is solved through a call to the software AlphaBayes assuming that the additive and residual variances in the training population are known. Finally, AlphaSim computes gEBV for the selection candidates using their SNP genotypes and the estimated SNP effects $\hat{m}_j$:

$$gEBV_i = \sum_{j=1}^{n_{SNP}} \hat{m}_j x_{i,j}$$

The accuracy of the gEBV is determined by the Pearson correlation coefficient between the gEBV and TBV in the test dataset.

AddQtnToTheSnpChipYesNo must be specified if the **SelectionMethod** is *GBLUP*. The value of this parameter must be either *Yes* or *No*, and has to be provided in agreement with the value of **SnpChipIncludesQtnYesNoRandom**. One can distinguish different cases:

- **If SnpChipIncludesQtnYesNoRandom is No, AddQtnToTheSnpChipYesNo can be either *Yes* or *No*.**
  - If *Yes*, QTN are added to the SNP chip in addition to SNP. The QTN added to the SNP chip will be the unrestricted or restricted QTN according to the value specified for **SelectionBasedOnRestrictedOrUnrestrictedQtn**.
  - If **AddQtnToTheSnpChipYesNo** is *No*, the SNP chip includes SNP markers only.
- If **SnpChipIncludesQtnYesNoRandom** is *Yes*, **AddQtnToTheSnpChipYesNo** must be *No* to avoid that the chip includes the QTN twice.
- If **SnpChipIncludesQtnYesNoRandom** is *Random*, **AddQtnToTheSnpChipYesNo** can be either *Yes* or *No*. It must be noted that if **AddQtnToTheSnpChipYesNo** is *Yes*, the selection chip might include some QTN twice.

Note: Selecting using the phenotypic values is not allowed when selecting for the recombination rate (i.e., when **RecombinationOnOff** in Box 8 is *On*). Indeed, the recombination rates in the selection candidates are still unknown when performing selection since their values are calculated by counting the recombinations that occurred in their progenies.

UseExternalSnpSolutionsYesNo must be specified if the **SelectionMethod** is *GBLUP*. The value of this parameter is either *Yes* or *No*. If *Yes*, four files must be provided: *SnpSolutions.txt*, *AlleleFreqTrain.txt* and *FixedSnp.txt* and *SelectionPhenTrain.txt*, which include the SNP solutions *per se*, the allele frequencies, fixed SNP and phenotypic values in the training dataset. These files must be stored in the main directory, i.e., the directory from where the AlphaSim program is run. The format of these files is described in section SnpSolutionsFiles.

Note: Using external SNP solutions is not supported when simulating selection on multiple traits.

SelectionPhenotypingStrategy must be specified if **SelectionMethod** is *GBLUP* and **UseExternalSnpSolutionsYesNo** is *No*. **SelectionPhenotypingStrategy** can be *AllPhenotypes*, *AllHistoricalPhenotypes*, *PreviousGeneration*, *AllMales* and *RandomPhenotypes*. The individuals included in the corresponding training datasets are determined as follows. For a given selection generation *g*, the training dataset includes:

- If the strategy is *AllPhenotypes*, all individuals from generations 1 to *g-1*;
- If the strategy is *AllHistoricalPhenotypes*, all individuals from generations 1 to *g-2*;
- If the strategy is *PreviousGeneration*, all individuals of generation *g-2* only;
- If the strategy is *AllMales*, all male individuals from generations 1 to *g-1*;
- If the strategy is *RandomPhenotypes*, *n* individuals randomly sampled in generations 1 to *k*, where *n* and *k* are specified by the parameter **TrainingSetSizeAndLastTrainingGenForRandPhenoStrat**.

Note:

- When using the strategies *AllHistoricalPhenotypes* or *PreviousGeneration*, the number of burn-in generations must be at least 2 to enable the selection of the adequate phenotypes for the training dataset.
- When selecting for the recombination rate, phenotypic values are not available for generation *g-1* since the recombination rates in that generation, which are calculated by counting the recombinations that occurred in the progenies, are still unknown. As a result, the strategy *AllPhenotypes* is not compatible with selection for recombination, while the strategy *AllMales* has been modified so that the training dataset includes all males from generations 1 to *g-2* instead of *g-1*. Thus, when selecting for recombination using the strategy *AllMales*, the number of burn-in generations must be at least 2 to enable the selection of the adequate phenotypes for the training dataset.

FirstAndLastTrainingGenForRandomPhenotypesStrat and **TrainingSetSizeForRandomPhenotypesStrat** must be provided only if **SelectionPhenotypingStrategy** is *RandomPhenotypes*.

FirstAndLastTrainingGenForRandomPhenotypesStrat defines the first and last generations in which training individuals will be sampled. Two integers must be provided.

Note: Attention must be paid to set the first and last training generations such that data are available for all of the selection generations to be run. For example, when **GenerationFlexibilityOnOff** is *Off*, *LastTrainGen* must be a burn-in generation, i.e., equal to or smaller than *nGen*, and, when selecting for recombination, equal to or smaller than *nGen-1*, since the recombination rates in the nGenthgeneration are unknown when determining the haplotypes of the first selection generation.

FixTheTrainingDataAcrossGenerationsOnOff must be specified if **SelectionMethod** is *GBLUP* and **UseExternalSnpSolutionsYesNo** is *No*. **FixTheTrainingDataAcrossGenerationsOnOff** must be *On* or *Off*. This parameter allows fixing the training data across all the generations of selection. Thus, by setting **FixTheTrainingDataAcrossGenerationsOnOff** to *On*, the SNP solutions, computed using the training dataset of a given selection generation, are used to compute the gEBV of the test individuals of other selection generations. The generation from which the training data are fixed is specified by the parameter **UseTrainingDataFromGeneration**.

UseTrainingDataFromGeneration must be provided if **FixTheTrainingDataAcrossGenerationsOnOff** is *On*. The value of this parameter is an integer comprised between *nGen+1* and *nGen+nGenSel*.

Note: When the flexibility option is NOT used, the value of **UseTrainingDataFromGeneration** can only be the first selection generation, i.e., *nGen+1*. Fixing the training data to a selection generation that is posterior to generation *nGen+1* requires running AlphaSim sequentially using the flexibility option. For example, to use the training dataset from the third selection generation in posterior selection generations, AlphaSim mut be run from generation 1 to *nGen+3* with **FixTheTrainingDataAcrossGenerationsOnOff** set to *Off*, and then from generation *nGen+4* to *nGen+nGenSel* with **FixTheTrainingDataAcrossGenerationsOnOff** set to *On* and **UseTrainingDataFromGeneration** set to *nGen+3*.

ComputeAverageCoancestryYesNo can be *Yes* or *No*. If *Yes*, information about the co-ancestry in generations *nGen* to *nGen+nGenSel* is computed. The average co-ancestry is computed using the genomic relationship matrix, which is itself computed by calling AlphaAGH. Two distinct values are computed (see section SimulatedData for details). For a given generation, the first value of co-ancestry is the average relationship among the individuals included in the generation considering a same weight for each individual, while the second value is the average relationship weighted according to the contribution of each individual to the next generation.

Box 11: Optimal contribution selection

Broadly, OC selection works by optimising the contribution of each individual to the next generation by accounting for the genetic potential of the selected individuals and for the genetic variation between them. The selection criterion to optimise is:

$$\mathbf{x}'\hat{b} + \lambda\mathbf{x}'\mathbf{G}\mathbf{x}$$

where $\mathbf{x}$ represents the vector of contributions of each individual to the next generation, $\hat{b}$ represents the vector of breeding values of each individual, $\mathbf{G}$ represents the genetic or genomic relationships between all individuals, and λ is a penalty. Thus $\mathbf{x}'\hat{b}$ represents the genetic mean of the selected individuals, and $\mathbf{x}'\mathbf{G}\mathbf{x}$ represents the average co-ancestry of the selection candidates, which is directly related to the level of inbreeding, $\frac{1}{2}\mathbf{x}'\mathbf{G}\mathbf{x}$ (Meuwissen, 1997; Wray and Goddard, 1994).

Note: HOW ALPHASIM PERFORMS OC SELECTION

Two softwares, AlphaAGH and AlphaMate, are successively called. AlphaAGH computes the genomic relationship matrix among the selection candidates. To this purpose, it uses the SNP genotypes or QTN genotypes depending on the selection method, GBLUP or TBV, respectively. Then, AlphaMate performs OC selection *per se*.

Firstly, AlphaMate performs a search for the $\mathbf{x}$ vector that minimises the average co-ancestry among the selection candidates without taking breeding values into account. This is done by setting λ to -1 and limiting the criterion only to the 'inbreeding' component, i.e., to $\lambda\mathbf{x}'\mathbf{G}\mathbf{x}$. A population of randomly generated $\mathbf{x}$ vectors is evaluated using an evolutionary algorithm, which evolves the $\mathbf{x}$ vectors via mutation, recombination, and selection for a specified number of generations. The $\mathbf{x}$ vector that maximises $\lambda\mathbf{x}'\mathbf{G}\mathbf{x}$ (and minimises $\mathbf{x}'\mathbf{G}\mathbf{x}$ given that λ =-1) is selected. This $\mathbf{x}$ vector is referred to as $\mathbf{x_0}$, and $\mathbf{x_0}'\mathbf{G}\mathbf{x_0}$ is assumed to be the minimum average co-ancestry that can be achieved among the selection candidates, or minimum level of inbreeding of their progeny.

Then, AlphaMate tests a range of λ (varying from -10^{-8} to -10^{292} by a multiplicative factor of 10^3) using the selection criterion $\mathbf{x}'\hat{b} + \lambda\mathbf{x}'\mathbf{G}\mathbf{x}$. For each λ, the $\mathbf{x}$ vector that maximises the selection criterion is evolved using the evolutionary algorithm and the corresponding average co-ancestry is computed. This procedure continues the same way until the difference between the initial minimum average co-ancestry, $\mathbf{x_0}'\mathbf{G}\mathbf{x_0}$, and the average co-ancestry obtained with the λ under consideration reaches a user-specified value. The $\mathbf{x}$ vector obtained with the final λ is used to determine the number of matings to which each selection candidate has to contribute.

OptimalContributionSelectionOnOff can be *On* or *Off*. If **OptimalContributionSelectionOnOff** is *On* and **GenerationFlexibilityOnOff** is set to *Off*, then selection is performed by OC in all selection generations. To restrict OC selection to some specific selection generations, **GenerationFlexibilityOnOff** must be *On*. In that case, OC selection is performed in the generations specified by the **StartStopGeneration** option only.

SizeOfXvectPopulationAndMaxNbOfGenerations are parameters of the evolutionary algorithm that determine the size of the population of $\mathbf{x}$ vectors and the number of generations during which the $\mathbf{x}$ vectors are evolved.

PercentageDifferenceInbreeding allows the user to control the increase of the inbreeding level across the selection generations. More specifically, this parameter is the increase in level of inbreeding that must be achieved when performing OC selection. Its value must range between 0 and 1.

Note:

- When performing OC selection, check the *OutAM* file, which contains the printing statements produced by AlphaMate (see section SelectionFolder for details).
- When **PlantBreedingOnOff** is *On* and **PerformCrossesConsideringTheGenderYesNo** is set to *No*, optimal contribution selection allows the occurrence of selfing when this results in an optimal balance of genetic gain vs. co-ancestry.
- OC selection is not allowed when performing gene editing.

Box 12: Traits

AlphaSim allows the simulation of one single trait or multiple traits, with the restriction that all traits are characterized by the same set of QTN, unrestricted or frequency-restricted, and have the same distribution, normal or Gamma.

NumberOfTraits is an integer specifying the number of traits, n_{Traits}, that will be simulated by AlphaSim.

TraitHeritability is an array of dimensions (1, n_{Traits}) that specifies the heritability of each of the *n* traits. If **IncludeDominanceEffectsOnOff** is set to *On*, **TraitHeritability** must include the broad-sense heritability of each trait. Otherwise, it must include the narrow-sense heritability of each trait.

TraitVariances is an array of dimensions (1, n_{Traits}) that specifies the *a priori* genetic variance of each of the *n* traits. The values provided for **TraitVariances** are used to sample additive effects for the QTN (see Box 7 for details).

IndexWeights is an array of dimensions (1, n_{Traits}) that specifies the weight of each of the *n* traits. The index weights must be such that their sum is equal to 1. When n_{Traits}>1, the index weights are used to compute a selection index:

$$SelectionIndex_i = \sum_{r=1}^{n_{Traits}} \frac{BV_{i,r}}{\sqrt{var_{A_r}}} \omega_r$$

where $BV_{i,r}$ is the breeding value of individual *i* for trait *r*, var_{A_r} is the additive variance of trait *r*, computed using the breeding values of the individuals included in the generation of individual *i*, and ω_j is the weight of trait *r*. The breeding values can be TBV, phenotypes or gEBV, according to the selection method.

Note: Multiple traits simulation is not allowed when selecting for recombination, performing gene editing, or using external SNP solutions. In these cases, the number of simulated traits must be equal to 1.

Box 13: Trait correlations

AlphaSim allows specifying the genetic and residual correlations between the trait(s). The correlations must be provided in Box 13 in the form of two matrices called **GeneticCorrelationMatrix** and **ResidualCorrelationMatrix**.

Both matrices are of dimensions $n_{Traits} \times n_{Traits}$, with only lower triangle specified, and have values of 1 on the diagonal. If only one trait is simulated, then both matrices should be reduced to the single scalar “1.0”.

Note: When simulating different traits, AlphaSim uses the parameters **TraitHeritability**, **TraitVariances**, **GeneticCorrelationMatrix** and **ResidualCorrelationMatrix** to simulate additive QTN effects and residual effects for each trait as follows.

- **Additive effects:** The *a priori* genetic variances and the genetic correlation matrix are used to compute the genetic variance-covariance matrix, $\mathbf{V_A}$. After Cholesky decomposition, $\mathbf{V_A} = \mathbf{L_A} \cdot \mathbf{L}'_A$, and the Cholesky factor $\mathbf{L_A}$ is used to compute additive effects for each QTN *k* and trait *r*:

  $$a_{k,r} = \sum_{r=1}^{n_{Traits}} \sum_{s=1}^{r} RandDev \frac{L_{A_{r,s}}}{\sqrt{n_{QTN}}}$$

  where *r* and *s* are trait indicators, n_{Traits} is the number of traits, and *RandDev* is a random deviate sampled from a normal or Gamma distribution as for the simulation of one single trait (see Box 7).

  If **IncludeDominanceEffectsOnOff** is *On*, dominance degrees, $\delta_{k,r}$, are sampled for each trait from a normal distribution with mean m_δ and variance var_δ, as specified by the user in Box 7, and dominance effects $d_{k,r}$ and allelic substitution effects $\alpha_{k,r}$ are computed for each QTN *k* and trait *r* (see Box 7 for details).

- **Residual effects:** Residual variances var_{E_r} are computed for each trait *r* relative to the specified trait heritability and the variance of the TBV and TDV of base individuals obtained with the corresponding trait. The variances var_{E_r} are then used jointly with the specified residual correlation matrix to derive the residual variance-covariance matrice $\mathbf{V_E}$. After Cholesky decomposition, $\mathbf{V_E} = \mathbf{L_E} \cdot \mathbf{L}'_E$, and the Cholesky factors $\mathbf{L_E}$ are used to compute residual effects $e_{i,r}$ for each individual *i* and trait *r*:

$$e_{i,r} = \sum_{r=1}^{n_{Traits}} \sum_{s=1}^{r} RandDevL_{E_{r,s}}$$

where *RandDev* is a random deviate sampled from a Gaussian distribution with zero mean and standard deviation of one unit.

Box 14: Outputs

KeepChromosomesFolderYesNo can be *Yes* or *No*. If *Yes*, the folder *./Chromosomes/*, which includes itself a subfolder for each chromosome (see details in section Chromosomes), is kept after the end of the program (i.e., when the last generation has been run). Otherwise, the folder is deleted.

WriteFullSequenceOutOnOff can be *On* or *Off*. If *On* then distinct files including full sequences are written out, and **ZipSequenceOutputFilesYesNo** must be specified.

The files that are created by setting WriteFullSequenceOutOnOff to *On* are:

- **for each chromosome h and each generation *g*:**
  - *./Chromosomes/Chromosome"h"/SequenceGenotypeGeneration"g".bin*
  - *./Chromosomes/Chromosome"h"/SequencePhaseGeneration"g".bin*
- **for all chromosomes and generations together:**
  - *./SimulatedData/FullSequencePhase.txt*
  - *./SimulatedData/FullSequenceGenotype.txt*
  - *./SimulatedData/FullSequencePhysicalMapIncluded.txt*

ZipSequenceOutputFiles can be *Yes* or *No*. If *Yes*, then the aforementioned files are compressed to ".gz" files. To decompress these files, you can use the command '`gunzip -f filename`'.

Box 15: Genome compression

Genome compression option allows reducing the number of segregating sites generated by MaCS that will be dropped from one generation to another and from which SNP and QTN will be selected.

For example, this option makes dropping highly variable haplotypes with large numbers of segregating sites (due to large effective population size or other factors) feasible in a given amount of computation time and memory.

GenomeCompressionOnOff can be *On* or *Off*. If *On*, then the parameter **GenomeCompression** must be provided.

GenomeCompression specifies, in %, the proportion of segregating sites generated by MaCs for a given chromosome that has to be kept in AlphaSim. For instance, setting GenomeCompression to 75.00 means that 75% of the segregating sites generated for each chromosome will be kept in the simulation.

Note: When compressing the genome, attention must be paid to keep a number of segregating sites high enough to provide the required numbers of SNP and QTN per chromosome.

macsParams.txt (optional)

The default system for generating the founder haplotypes is a system call to an external software package, MaCS (Chen et al., 2009). This component is controlled either internally by AlphaSim, or by reading in an external file of user-specified input MaCS parameters. Users may download a Microsoft Excel sheet (*MaCS_parameters.xlsx*) available with the AlphaSim executables which provides assistance on the generation of MaCS input parameters. A brief description of the parameters for MaCS files is given below. For a detailed explanation of the parameters refer to the MaCS documentation (Chen et al., 2009; Hudson, 2002). The initial set of parameters describing the genome "architecture" is:

- the length of the haplotypes (in base pairs): a common value might be 10^8 b, which corresponds approximately to 1 Morgan.
- `-t` *theta*: parameter theta equals $4N_e \cdot \mu$ with μ being the per-base mutation rate; common values for μ are in the range of 1×10^{-8} to 2×10^{-8}, which corresponds to about one or two mutations per haplotype of 10^8 b.
- `-r` *rho*: parameter rho equals $4N_e \cdot \ rho$ with ρ being the per-base recombination rate; a common value for ρ is 10^{-8}, which would give on average 1 recombination per haplotype of 10^8 b and defines the genetic length of such haplotypes to be 1 Morgan.

Effective population size (N_e) in the above specifications is the "final/present" N_e of a population from which the haplotypes are sampled at the end of coalescent simulation. Depending on the type of simulation this N_e might vary from $\sim$ 20 when we want to simulate data for a population of 40 inbred plant lines, over $\sim$ 100 when we want to simulate data for an intensively selected population, up to 1,000 or more when we want to simulate more diverse settings. Specifying small N_e leads to a small number of segregating sites, as the amount of variation defined through N_e is low. However, most populations went through various processes, for example domestication, which changed effective population size over time to today's values. When this is accommodated, the amount of variation simulated (number of segregating sites and distribution of allele frequencies) is more in line with observed variation in real populations. MaCS has options to accommodates particular types of changes in N_e (exponential expansion and decay). When this is not sufficient, we can approximate trajectory of changes by steps via repeated use of `-eN` options:

- `-eN` *time scale*: parameter that describes the time (in number of generations back in time divided by the $4N_e$) when N_e is changed by a scaling factor relative to the final N_e. Usually several changes in N_e are defined, for which the supplied Excel sheet might come handy. For cattle, `-eN` values were set using the final N_e equal to either 100 or 1000 (see the two distinct files). For rabbit, wheat, and maize, the values were obtained using $N_e = 100, 50$ and 100, respectively. Note that the grid of `-eN` values in examples depend on the final and historical values of N_e, and are thus example specific.

Additional parameters allow splitting the population of haplotypes in two distinct sub-populations, which might be useful when simulating several breeds or heterotic groups. For example, the maize population in AlphaSim is simulated as:

- `-I` $N_p, N_{h_1}, N_{h_2}, \ldots, N_{h_n}$ with N_p being the number of sub-populations, and $N_{h_1}, N_{h_2}, \ldots, N_{h_n}$ the numbers of haplotypes to sample in the 1^{st}, 2^{nd}, ..., n^{th} sub-population, respectively.
- `-ej` t, i, j indicate the time of divergence in the $4N_e$ units between subpopulations i and j.

Seed.txt (mandatory)

Running AlphaSim requires provided a file including a seed. This file must be named *Seed.txt* and stored in the main directory, i.e., the directory from which the AlphaSim program is run, and includes an negative integer (e.g., -139446321).

Pedigree files (optional)

When working with an external pedigree or an internal pedigree but external selection pedigrees, AlphaSim requires providing pedigree files in the following format. Pedigree files should have three columns, individual, male parent, and female parent. It should be space or comma separated with missing/uknown parents coded as 0. No header line should be included in the pedigree file. Both numeric and alphanumeric formats are acceptable. The pedigree does not have to be sorted in any way as the program automatically does this.

When working with an external pedigree, the pedigree file must be stored in the main directory, i.e., the directory from which the AlphaSim program was run. The name of the pedigree file is provided in the **PedigreeStatus** option of *AlphaSimSpec.txt*.

When importing an external pedigree for a given selection generation, the pedigree file must be named *ExternalPedigree.txt* and stored in the selection folder corresponding to the selection generation under consideration (see Box 4 for details). This pedigree file must have as many rows as there are progeny in the generation that is about to be simulated.

User selection of QTN to be edited (optional)

When performing gene editing on his own selection of QTN (**PerformGEOnUserQtnSelectionYesNo** is *Yes*), the user must provide a file including the QTN to be edited in the main directory, i.e., the directory from which the AlphaSim program is run. This file must be named *UserSelectedQtnToBeEdited.txt* and must have one single column, including 0's (QTN cannot be edited) and 1's (QTN can be edited), with the values sorted from the first QTN of the first chromosome to the last QTN from the last chromosome. For instance, if there are 2 chromosomes with 10 QTN each, QTN 1 to 10 from the first chromosome must be provided first, followed by QTN 1 to 10 from the second chromosome. AlphaSim will perform gene editing on the first n_{edited} QTN having a value of 1 as specified in the file.

SNP solutions files (optional)

When using SNP solutions derived from some external data (**UseExternalSnpSolutionsYesNo** is *Yes*), AlphaSim requires four files to be provided: *SnpSolutions.txt*, *AlleleFreqTrain.txt*, *FixedSnp.txt* and *SelectionPhenTrain.txt*. These files should have columns space or comma separated and without a header line.

- *SnpSolutions.txt* must include two columns, the first including a line identifier and the second the estimated SNP effect, and as many rows as there are SNP on the selection SNP chip.
- *FixedSnp.txt* must include one column with as *nSNP* rows if the selection chip does not include *QTN*, *nSNP+nQTN* otherwise, with *nSNP* being the number of SNP included in the selection chip and *nQTN* the total number of QTN with normal distribution. The value is 0 if the SNP is fixed among the individuals included in the training population, 1 otherwise (polymorphic SNP).
- *AlleleFreqTrain.txt* must include two columns, the first including a line identifier and the second, the frequency of the non-zero allele for each of the not-fixed SNP. The file must have as many rows as there are polymorphic SNP on the selection SNP chip, in contrast to *SnpSolutions.txt* which must include as many rows as there SNP on the selection SNP chip.
- *SelectionPhenTrain.txt* must include two columns, the first including an line identifier and the second, the phenotypic values of the individuals included in the training dataset. The file must have as many rows as there are individuals in the external training dataset.

These four files must be stored in the main directory, i.e., the directory from which the AlphaSim program was run.

1.10.2 Additional programs

Several additional programs are required when running AlphaSim with different options. A short comment on what each of these programs does is given here.

MaCS is called to simulate the founder haplotypes (see section **‘macsParams.txt‘_** for details).

The following programs are called when using specific options:

- **AlphaBayes** is called when **SelectionMethod** is *GBLUP* to provide estimates of SNP effects.
- **AlphaAGH** is is called when **OptimalContributionSelectionOnOff** is *On* as well as when **ComputeAverageCoancestryYesNo** is *Yes* to compute the genomic relationship matrix among a given set of individuals.
- **AlphaMate** is called when **OptimalContributionSelectionOnOff** is *On* to perform optimal contribution selection.

All of these programs must be stored in the main directory, i.e., the directory from which the AlphaSim program is run. According to the parameters specified in *AlphaSimSpec.txt*, AlphaSim copies and then runs the required programs in the appropriate folders.

1.10.3 Output files

The output of AlphaSim is organised in three directories (*Chromosomes*, *Selection* and *SimulatedData*). A description of what is contained within each of these directories is given below.

In addition to output files, AlphaSim prints statements in the terminal. This output contains information about the process, some synthetic results, error statements and notes. Error statements stop the simulating process. In contrast, notes do not stop the simulating process; however, they can be useful to track an error that would not have been indicated by the program.

Chromosomes

The chromosome directory includes a subdirectory for each of the simulated chromosomes. Each of these subdirectories contains:

Files created by MaCS

- *FinishedMacs.txt* includes the number of segregating sites generated by MaCS, *nSegreg*.
- *MacsHaplotypes.txt* includes the sequences of each of the haplotypes generated by MaCS, *nHaplos*. The file has *nHaplos* rows and *nSegreg* columns.
- *Output.txt* includes, for each segregating site, the physical position of the site on the map and the value (0 or 1) in each of the *nHaplos* haplotypes generated by MaCS.
- *PhysicalMapInput.txt* contains the positions of each of the segregating sites generated by MaCS.

Files created by AlphaSim

- *BaseIndividualsGameteTracking.txt* includes five columns: identifier of the individual, identifier of each of the two paternal haplotypes, and identifier of each of the two maternal haplotypes. The file is filled in for the individuals of the first generation only (i.e., the founders).
- *Chip”e”GenotypeGeneration”g”.txt* includes the genotypes of each individual of generation *g* on chip *e*. The files have as many rows as there are individuals in the g^{th} generation, and as many columns as there are SNP on the eth chip plus one, for the individual ids (first column).
- *Chip”e”PhaseGeneration”g”.txt* includes the sequence phases of each individual of generation *g* on chip *e*. The files have twice as many rows as there are individuals in the g^{th} generation (two lines per individual, one for

each haplotype), and as many columns as there are SNP on the eth chip plus one, for the individual identifiers (first column).

- *Chip"e"SnpInformation.txt* includes information about the SNP present on chip *e*. There are four columns: line identifier, SNP identifier, its minor allele frequency, and its physical position on the chromosome.
- *Chip"e"Summary.txt* includes only one number, which is the mean heterozygosity of the SNP found on chip *e*. The hererozygosity is obtained as follows:

$$Heterozygosity = \frac{\sum_{j=1}^{n_{SNP}} \left[1 - \left(p_j^2 + (1-p_j)^2\right)\right]}{n_{SNP}}$$

  where *p* is the frequency of the non-zero allele at SNP *j*.

- *GenomeGeneration"g".txt* stores the genomic data of generation g. The data are under the form of a three-dimensional array, the three dimensions corresponding to the individual, the segregating site and the haplotype haplotype, respectively. The file is unformatted, it is readable by AlphaSim only.
- *MacsHaplotypesInBlocks.txt* stores the data of each haplotype expressed in binary numbers (see details in Description of the method for block definition). This file is unformatted, it is readable by AlphaSim only.
- *MacsHaplotypesIncluded.txt* includes the sequences of each of the haplotypes generated by MaCS. The file has as many rows as there are haplotypes and as many columns as there are segregating sites that are included (see details in Description of the method for "included" segregating sites).
- *MinorAlleleFrequency.txt* includes two columns: the identifier of the segregating site and the allele frequency.

  The minor allele frequency (MAF) is computed for each segregating site based on the sequence phases in the first generation (i.e., the founder's haplotypes) as follows:

$$MAF = \frac{\sum_{i=1}^{n_{indiv}} Phase_1 + Phase_2}{2n_{indiv}}$$

  where $Phase_1$ and $Phase_2$ are the alleles on the first and second haplotypes, and n_Indiv is the number of founders. If MAF is smaller than 0.5, then MAF is substituted by $1 - MAF$.

- *MutAnimIndicator.txt* includes two columns and as many rows as there are individuals in the pedigree. The first column is the individual identifier, and the second one is 1 or 0 whether the individual is mutated or not.
- *MutDeNovoInfo.txt* includes the information about the mutation sites: mutation site, mutated individual, mutated haplotype (1 or 2), block including the mutation site, and position of the mutation site (see details in Description of the method for block definition).
- *NbOfQtnMutationsAndSnp.txt* summarizes the number of QTN, mutations and SNP that are included on the chromosome (see details in Description of the method for included segregating sites).
- *PhysicalMapIncluded.txt* contains five columns and as many rows as there are included segregating sites sites after having applied the genome compression percentage if **GenomeCompressionOnOff** is *On* (see Box 15 for details). The first column is the line identifier, the second is the segregating site, the third is the physical position of the site on the map, the fourth is a mutation indicator being either 1, if the site is a mutation site, or 0 otherwise, and the fifth column includes the identifier of the mutated individual when the mutation indicator is 1.
- *PhysicalMapCompressed.txt* contains three columns and as many rows as there are included segregating sites after having applied the genome compression percentage (see Box 15 for details).
- *PhysicalMapUncompressed.txt* contains three columns and as many rows as there are segregating sites generated by MaCS: identifier of the segregating site, physical position of the site on the map and mutation indicator being 1 if the site is a mutation site, or 0 on the opposite.
- *RecombinationTracking.txt* includes two rows per individual, and three first columns followed by as many columns as there were recombinations in an individual. The first column is the individual identifier, the second column is the identifier of the gamete received from the father (gamete 1 or gamete 2), and the third one is

the identifier of the gamete received from the mother (again, 1 or 2). The fourth column includes the number of recombinations that occurred in the male or female parent, respectively, and the following columns provide the positions of the recombinations.

- *RecombinationQtnAlleleFreqGeneration"g".txt*, *RestrictedQtnAlleleFreqGeneration"g".txt*, and *UnrestrictedQtnAlleleFreqGeneration"g".txt* include three columns and one row per QTN for each set of QTN, recombination, restricted and unrestricted, respectively. The first column is a line identifier, the second one is the QTN identifier, and the third one is the frequency of the non-zero QTN allele in generation *g*.
- *RecombinationQtnGenotype"g".txt*, *RestrictedQtnGenotype"g".txt*, and *UnrestrictedQtnGenotypeGeneration"g".txt* provide the same information than *Chip"e"GenotypeGeneration"g".txt* for the recombination, restricted and unrestricted sets of QTN, respectively.
- *RecombinationQtnInformationGeneration"g".txt*, *RestrictedQtnInformationGeneration"g".txt*, and *UnrestrictedQtnInformationGeneration"g".txt* provide the same information than *Chip"e"SnpInfomration"g".txt* for the recombination, restricted and unrestricted sets of QTN, respectively.
- *RecombinationQtnPhaseGeneration"g".txt*, *RestrictedQtnPhaseGeneration"g".txt*, and *UnrestrictedQtnPhaseGeneration"g".txt* provide the same information than *Chip"e"PhaseGeneration"g".txt* for the recombination, restricted and unrestricted sets of QTN, respectively.
- *RecombinationQtnSummary.txt*, *RestrictedQtnSummary.txt*, and *UnrestrictedQtnSummary.txt* provide the same information than *Chip"e"Summary.txt* for the recombination, restricted and unrestricted sets of QTN, respectively.
- *SegSites.txt* stores the number of segregating sites generated by MaCS after having applied the genome compression percentage if **GenomeCompressionOnOff** is *On* (see Box 15 for details).
- *SegSitesIncluded.txt* stores the number of included segregating sites, which is the largest integer that is smaller than the number of segregating sites divided by the number of loci per block, which is set to 60.b

When **WriteFullSequenceOutOnOff** is *On*, AlphaSim creates two additional files per generation:

- *SequenceGenotypeGeneration"g".bin*
- *SequencePhaseGeneration"g".bin*

Setting **ZipSequenceOutputFilesYesNo** to *Yes* transforms these files into ".txt.gz" files.

SelectionFolder

The SelectionFolder directory includes a subdirectory for each simulated selection generation. Each of these subdirectories contains basically five files. These files are created regardless of the selection method, *TBV*, *GBLUP* or *Pheno*. In the file names below, the letter "e" is the identifier of the SNP chip used for selection.

- *SelectionTbvTest.txt* provides the TBV of each of the individuals included in the test dataset. The first column is the individual identifier, and the second one is its TBV.
- *TmpChip"e"GenotypeTest.txt* provides the SNP genotypes of the individuals included in the test dataset. The first column is the individual identifier, and the additional columns contain the genotype at each of the SNP included on the selection SNP chip. The number of SNP used for selection is determined by the number of SNP per chip, increased by the number of QTN if the latter are added to the selection SNP chip (see **SnpChipIncludesQtnYesNoRandom** in Box 7 and **AddQtnToTheSnpChipYesNo** in Box 10). The SNP are ordered as follows: SNP from 1st chromosome, SNP from 2nd chromosome, ..., SNP from nth chromosome, and then, if the selection SNP chip includes the QTN, QTN from 1st chromosome, QTN from 2nd chromosome, ..., QTN from nth chromosome.

SELECTION METHOD:

When **SelectionMethod** is *GBLUP*, AlphaSim creates distinct files relative to the training dataset, and then calls the software AlphaBayes to solve the Ridge regression.

The following files are input files for AlphaBayes, created by AlphaSim:

- *AlphaBayesSpec.txt* is created by AlphaSim and read by AlphaBayes. It includes the parameters required to run AlphaBayes.
- *FixedSnp.txt* is 0, if the SNP is fixed among the individuals included in the training population, 1 otherwise (polymorphic SNP). The file includes one column with as *nSNP* rows if the selection chip does not include QTN, *nSNP+nQTN* otherwise, with *nSNP* being the number of SNP included in the selection chip and *nQTN* the total number of QTN with normal distribution.
- *Seed.txt* includes the seed required to run AlphaBayes.
- *SelectionPhenTrain.txt* provides the phenotypic value of each of the individuals included in the training dataset. The first column is the individual identifier, and the second, the phenotypic value (or TPV) computed using the QTN model used for selection. Details about the selection method, type of training dataset, and QTN model used for selection, are given in Box 10.
- *TmpChip"e"GenotypeTrain.txt* provides the genotypes at each of the SNP used for selection for the individuals included in the training dataset. The format of the file is the same as TmpChip"e"GenotypeTest.txt.
- *TmpChip"e"PhaseTrain.txt* provides the sequence phases at each of the SNP used for selection for the individuals included in the training dataset. The format of the file is the same as *TmpChip"e"GenotypeTest.txt*, with the exception that *TmpChip"e"PhaseTrain.txt* has twice as many rows as there are individuals in the training dataset (two lines per individual, one for each haplotype).

In turn, AlphaBayes generates several output files:

- *outAB* is the output produced by AlphaBayes (printing statements).
- *AlleleFreqTrain.txt* includes the frequency of the non-zero allele across the training population. The files has two columns, the first including a line identifier and the second, the frequency of the non-zero allele for each of the not-fixed SNP, and as many rows as there are polymorphic SNP on the selection SNP chip.
- *Ebv.txt* includes two columns and as many rows as there are individuals in the test dataset. The first column is the individual identifier, and the second one the estimated breeding values.
- *SnpSolutions.txt* provides the solution of the Ridge regression for each of the SNP (and QTN if the selection chip includes QTN) included in the selection chip. The file has two columns, the first including a line identifier and the second the estimated SNP effect, and as many rows as there are SNP on the selection SNP chip.
- *TbvEbvCorrelation.txt* includes the correlation between the true breeding values (TBV) and the estimated ones computed among the individuals included in the test dataset.

When the parameter **FixTheTrainingDataAcrossGenerationsOnOff** is *On*, then the file *TmpChip"e"GenotypeTestScaled.txt* is created for the generations that are posterior to the generation specified by **UseTrainingDataFromGeneration**. This file includes the genotypes of the test dataset scaled according to the allele frequencies and phenotypic data in the training dataset. The scaled genotypic values are used together with the SNP solutions from the fixed training generation, which is specified by **UseTrainingDataFromGeneration**, to compute EBV.

MULTIPLE TRAITS:

When simulating more than one trait, the names of the files included trait-specific information are specified by the trait identifier.

The file *SelectionIndex.txt* is created, which includes the selection index that integrates the breeding values (*TBV*, *gEBV*, or phenotypic values according to the **SelectionMethod**) of each of the simulated traits according to the specified weights (see Box 12 for details).

RECOMBINATION:

When selecting for recombination in addition to the trait of interest (Box 8 is On), two additional input files are created:

- *SelectionPhenRecombTrain.txt* is empty if **SelectionMethod** is TBV or Pheno. If **SelectionMethod** is GBLUP, this file provides the phenotypic value of each of the individuals included in the training dataset. The first column is the individual identifier, and the second one the phenotypic value (or TPV) computed using the QTN model used for selection for the selected trait. The selection method, type of training dataset and QTN model used for selection are specified in Box 10, "Selection".
- *SelectionTbvRecombZTesting.txt* provides the recombination TBV of each of the individuals included in the test dataset. The first column is the individual identifier, and the second one is its recombination TBV. The recombination TBV is normally distributed with mean zero and standard deviation one.

AlphaBayes generates additional output files: *EbvRecomb.txt*, *SnpSolutionsRecomb.txt*, *TbvEbvCorrelRecomb.txt*, and when the parameter **FixTheTrainingDataAcrossGenerationsOnOff** is *On*, *TmpChip"e"GenotypeTestScaledRecomb.txt*. These files include the same information as *Ebv.txt*, *SnpSolutions.txt*, *TbvEbvCorrel.txt*, and *TmpChip"e"GenotypeTestScaled.txt*, respectively, for the recombination trait.

The file *SelectionIndex.txt* includes the selection index that integrates the breeding values (*TBV* or *gEBV* according to the **SelectionMethod**) of both the recombination trait and the common trait according to the specified weights (see Box 8 for details).

GENE EDITING:

When performing gene editing, the file *GeneEditingInformation.txt* is created. This file provides information about the gene editing: the first column includes the identifier of each edited sire in the generation, the second is the global identifier of the edited QTN, and the third and fourth provide the observed and aimed QTN genotypes, respectively. The aimed QTN genotype is either zero (when the non-zero allele of the QTN has a negative effect on the trait), or two (when the non-zero allele of the QTN has a positive effect on the trait). The number of rows in the file is equal to the number of edited sires multiplied by the number of edits per sire.

OPTIMAL CONTRIBUTION SELECTION:

When performing optimal contributions selection, two applications, AlphaAGH and AlphaMate, are called successively; AlphaAGH computes the genomic relationship matrix, while AlphaMate performs OC selection *per se* (see Box 11 for details).

The files *AlphaAGHSpec.txt* and *AlphaMateSpec.txt* are created by AlphaSim. These files include the parameters required to run AlphaAGH and AlphaMate, respectively.

The following files are created by AlphaAGH:

- *outAGH* includes the output produced by the printing statements in AlphaAGH.
- *AlleleFreqTest.txt* includes the frequency of the non-zero allele across the test population.
- *GFullMatrix1-1.txt* includes the genomic relationship matrix. This matrix is a $n \times n$ matrix, where *n* is the number of individuals included in the test population.

When **SelectionMethod** is *GBLUP*, AlphaAGH uses the file *TmpChip1GenotypeTest.txt* to compute the genomic relationship matrix. When **SelectionMethod** is *TBV*, the file *TmpQtnGenotypeTest.txt* is created to compute the genomic relationship matrix. This file has the same structure as *TmpChip1GenotypeTest.txt*, but it includes the genotypes of the test individuals at the QTN, restricted or unrestricted, according to the QTN model used for selection (see Box 7 for details), rather than their genotypes on the SNP chip used for selection.

AlphaMate creates seven files. All of them are stored in the folder AlphaMateResults, with the exception of outAM which is stored in the SelectionFolder of the generation under consideration.

- *outAM* includes the output produced by the printing statements in AlphaMate.
- *ContribAndMatingNbPerIndivFinal.txt* has as many rows as there are individuals in the test dataset, and four columns describing the results of the optimal contribution using the final λ, i.e., the λ that allows achieving the specified difference in level of inbreeding between generations (see Box 11 for details). The four columns include a row identifier, and, for each individual, its identifier, contribution and the number of matings in which the individual is involved. The contributions are such that $\sum_{i=1}^{nIndiv_{Test}} x_i = 1$. If the gender is considered

when performing optimal contribution, a "gender column" is added. In that case, the contributions are such that $\sum_{i=1}^{nMaleIndiv_{Test}} x_i = \sum_{i=1}^{nFemaleIndiv_{Test}} x_i = 0.5$.

- *ContribAndMatingNbPerIndivZero.txt* provides the same information than *ContribAndMatingNbPerIndivFinal.txt* using λ equal to -1 and without looking to maximising the genetic gain, i.e., when seeking to minimize the average level of inbreeding in the next generation (see Box 11 for details).
- *Frontier.txt* provides information about the optimization process. The first column is a row identifier. The second column includes the λ value. The third column provides the achieved average co-ancestry among the selection candidates ($\mathbf{x'Gx}$), the fourth one provides the achieved genetic gain ($\mathbf{x'}\hat{b}$), and the last one provides the value of the selection criterion ($\mathbf{x'}\hat{b} + \lambda\mathbf{x'Gx}$) (see Box 11 for details).
- *LambdaValueFinal.txt* provides the latest tested λ value, which allows achieving the specified difference of average inbreeding level between the two generations.
- *MatingListFinal.txt* contains the mating list obtained using the final λ value, which allows achieving the specified difference in level of inbreeding between generations. The file has two columns, including for each mating the identifiers of the individuals used as male and female parents, respectively. This mating list will be inserted in the pedigree of the generation that is being simulated by AlphaSim.
- *MatingListZero.txt* contains the mating list obtained using λ = -1 and without looking to maximising the genetic gain. This mating list is thus assumed to provide the minimum average level of inbreeding in the next generation. The file has two columns, including for each mating the identifiers of the individuals used as male and female parents, respectively.

CO-ANCESTRY:

When **ComputeAverageCoancestryYesNo** is *Yes*, AlphaAGH is called to compute the genomic relationship matrix. Additional files for AlphaAGH are created. *AlphaAGHSpec.txt* includes the parameters required to run AlphaAGH. When **SelectionMethod** is *TBV*, the file *TmpQtnGenotypeTest.txt* is created to compute the genomic relationship matrix. This file has the same structure as *TmpChip1GenotypeTest.txt*, but it includes the genotypes of the test individuals at the QTN, restricted or unrestricted, according to the QTN model used for selection (see Box 10), rather than their genotypes on the SNP chip used for selection.

Note: In the selection folder corresponding to generation *g*, AlphaSim computes the co-ancestry among the individuals of generation *g-1* using the genomic data of the test generation. The computation of the average co-ancestry in the last generation, in which no selection is performed, is done as follows. When running the last generation, AlphaAGH is called twice, a first time to compute the co-ancestry in the penultimate generation and a second time to compute the co-ancestry in the last generation. An additional file is created, which includes the genotypes of the last generation. This file is *TmpChip1GenotypeLastGeneration.txt* or *TmpQtnGenotypeLastGeneration.txt* depending on the **SelectionMethod**, *GBLUP* and *TBV*, respectively.

The following files are created by AlphaAGH:

- *outAGH* includes the output produced by the printing statements in AlphaAGH.
- *AlleleFreqTest.txt* includes the frequency of the non-zero allele across the test population.
- *GFullMatrix1-1.txt* includes the genomic relationship matrix. This matrix is a $n \times n$ matrix, where *n* is the number of individuals included in the test population.

SimulatedData

The SimulatedData directory includes a folder and several files:

- *AllIndividualsSnpChips* is a folder that includes two files per SNP chip *e*, *Chip"e"Genotype.txt* and *Chip"e"Phase.txt*, which gathers the genotypes and sequence phases of all individuals across all generations and chromosomes.

- *Chip"e"SnpInformation.txt* gathers the information about the SNP present on chip e from all of the simulated chromosomes. The file includes five columns: line identifier,the chromosome ID, SNP ID, its minor allele frequency, and its physical position on the chromosome.
- *Gender.txt* has three columns and as many rows as there are individuals in the pedigree. The first column includes the generation, the second one the individual ID, and the third one its gender, with 1 accounting for male and 2 for female.
- *GeneticMeansPerGeneration.txt* provides the average TBV achieved for each generation, QTN model and trait.
- *GlobalRecombinationMax.txt* provides the maximum position at which a recombination occurred on each chromosome.
- *NbOfIndivInEachGen.txt* provides, for each generation, the number of individuals, the number of male individuals, and the number of female individuals.
- *NbOfSegSitesPerChrom.txt* provides, for each chromosome, the total number of segregating sites generated by MaCS and the total number of segregating sites that have been included by AlphaSim.
- *Pedigree.txt* includes the complete pedigree that has been simulated. It has four columns: generation, ID of the individual, its father and its mother.
- *PedigreeTbvTdvTpv.txt* includes the pedigree as well as the TBV, TDV and TPV (true breeding values, true dominance values and true phenotypic values) that have been simulated for each QTN model and trait. Column identifiers are "TBV", "TDV" or "TPV", followed by the name of the QTN model and the trait ID. For the name of the QTN model, *Norm* and *Gamm* stand for normal and Gamma distribution, and *Unres* and *Restr* stand for unrestricted QTN and frequency-restricted QTN, respectively (see Box 7 for details).
- *ProportionOfVariationExplainedByNumberOfQtn.txt* provides, for each trait and each QTN model, the number of QTN that are required to explain five distinct percentages of variation, 1, 5 10, 25 and 50%.
- *RecombinationCount.txt* includes three columns and as many rows as there individuals in the pedigree. The first column includes the ID of the individual, and the second and third ones provide the number of recombinations that occurred across all of the simulated chromosomes in its paternal and maternal gametes, respectively.
- *ResidualVarCovMatrixCholDcQtnModel"k".txt* provides, for each of the four QTN models, the Cholesky decomposition of the residual variance-covariance matrix.
- *RestrictedQtnAdditEffects.txt* and *UnrestrictedQtnAdditEffects.txt* provide the additive effects of the frequency-restricted and unrestricted QTN, respectively. The first three columns include the global QTN identifier, the chromosome identifier and the QTN identifier as it is on each chromosome. The next columns include, for each trait successively, the additive effect of the QTN assuming a normal trait distribution, and, if the number of QTN with Gamma trait distribution is not null, the additive effect of the QTN assuming a Gamma trait distribution.
- *RestrictedQtnIndivGenotypes.txt* and *UnrestrictedQtnIndivGenotypes.txt* provide the genotypes of all of the individuals included in the pedigree on the frequency-restricted and unrestricted QTN, respectively. The first column includes the individual ID, and the additional columns contain the genotype at each of the QTN with restricted frequency.
- *RestrictedQtnInformation.txt* and *UnrestrictedQtnInformation.txt* gather the information about the frequency-restricted and unrestricted QTN, respectively, from all the simulated chromosomes. The files include eight columns: the global QTN identifier, the chromosome identifier, the QTN identifier as it is on each chromosome, its minor allele frequency and the frequency of its non-zero allele in the base generation of the pedigree, its physical position on the chromosome, the mutation indicator being 1 for the mutation sites and 0 otherwise, and the identifier of the mutated individual.
- *TotalGenicVariancesPerGeneration.txt* provides, for each trait and QTN model, the total additive and dominance genic variances in each generation. According to Falconer and Mackay (1996), the total additive ($\sigma^2_{A_g}$) and

dominance ($\sigma^2_{D_g}$) genic variances in generation *g* are as follows:

$$\sigma^2_{A_g} = \sum_{k=1}^{n_{QTN}} 2p_{k_g} q_{k_g} \alpha_k^2$$

and

$$\sigma^2_{D_g} = \sum_{k=1}^{n_{QTN}} (2p_{k_g} q_{k_g} d_k)^2$$

where p_{k_g} and q_{k_g} are the frequencies of the non-zero and zero alleles at QTN *k* in generation *g*, α_k is the allele substitution effect at QTN *k*, and d_k is the dominance effect at QTN *k*.

- *TraitVarianceComponents.txt* provides, for each trait and each QTN model, the additive (or genetic) variance, the dominance variance, the residual variance, the narrow-sense heritability, and the broad-sense heritability.

DOMINANCE:

When simulating dominance effects (**IncludeDominanceEffectsOnOff** in *On* in Box 7), four additional files are created:

- *RestrictedQtnAverAlleleSubstEffects.txt* and UnrestrictedQtnAverAlleleSubstEffects.txt* provide, for each trait and each QTN, frequency-restricted or unrestricted, the allele substitution effect computed assuming a normal trait distribution and, if the number of QTN with Gamma trait distribution is not null, the allele substitution effect computed assuming a Gamma trait distribution.
- *RestrictedQtnDominEffects.txt* and *UnrestrictedQtnDominEffects.txt* provide, for each trait and each QTN, frequency-restricted or unrestricted, the dominance effect and dominance degree assuming a normal trait distribution and, if the number of QTN with Gamma trait distribution is not null, the dominance effect and dominance degree assuming a Gamma trait distribution.

RECOMBINATION:

When selecting for recombination, three additional files are created. These are:

- *RecombinationQtnIndivGenotypes.txt* provides the genotypes of all of the individuals included in the pedigree at the QTN for recombination. The first column includes the individual identifier, and the additional columns contain the genotype at each of the QTN for recombination.
- *RecombinationQtnInformationEffects.txt* gathers the information about the QTN for recombination from all of the simulated chromosomes. The file includes seven columns: the global QTN identifier, the chromosome identifier, the QTN identifier as it is on each chromosome, its minor allele frequency and the frequency of its non-zero allele in the base generation of the pedigree, its physical position on the chromosome, and its additive effect on recombination.
- *RecombinationVarianceComponents.txt* provides the additive variance, the residual variance, and the simulated heritability computed using the QTN for the recombination rate.
- *ProportionOfVariationExplainedByNumberOfQtnRecombination.txt* provides the same information as *ProportionOfVariationExplainedByNumberOfQtn.txt*, for the QTN for the recombination rate.

GENE EDITING:

When performing gene editing, one additional file is created: *UnrestrictedQtnSortedForGeneEditing.txt* if selection is performed using the unrestricted QTN (QTN model 1 or 3), or *RestrictedQtnSortedForGeneEditing.txt* if selection is performed using the restricted QTN (QTN model 2 or 4). This file includes six columns and as many rows as there are edited QTN per individual. The QTN are sorted by descending effect size in absolute value. The first column is a line identifier. The additional columns include, successively, the global QTN identifier, the chromosome identifier, the QTN identifier as it is on each chromosome, an indicator of the direction of the QTN effect (1 if the effect is positive, 0 otherwise), and the identifier of the mutated individual for QTN corresponding to de novo mutations.

CO-ANCESTRY:

When **ComputeAverageCoancestryYesNo** is *Yes*, the file *AverageCoancestry.txt* is created. This file includes, for each generation ranging from *nGen* to *nGen+nGenSel* and the corresponding selection cycle, the average co-ancestry computed before selection (average co-ancestry computed by considering same weight to each individual) and the one computed after selection (weighted average computed by considering the contribution of each individual to the next generation).

FULL SEQUENCE OUTPUT:

When **WriteFullSequenceOutOnOff** is *On*, AlphaSim creates the following files, for all chromosomes and generations together:

- *FullSequencePhase.txt*
- *FullSequenceGenotype.txt*
- *FullSequencePhysicalMapIncluded.txt*

Setting **ZipSequenceOutputFilesYesNo** to *Yes* transforms these files into ".txt.gz" files.

1.10.4 Simulation example in AlphaSim

The provided parameters file can be used as an example of a simulation in AlphaSim. In that example, we simulate a pedigree including 6 generations, 3 burn-in and 3 derived from selection. The first generation includes 500 individuals. The second generation includes 50 DH generated from 50 male individuals randomly taken in generation 1. The third generation includes 25 individuals derived from 25 male individuals and 25 female individuals randomly taken in generation 2 and crossed with each other. Generations 4 to 6 include 20 individuals each derived from 5 individuals that are selected in generations 3 to 5, respectively, and selfed. AlphaSim creates the pedigree so that each parent has an equal number of progenies; therefore, each selfed individual will have 4 progenies (=20/5).

The genome has 2 chromosomes of length 1M. Two SNP chips are simulated, including 200 and 100 SNP, respectively, with minor allele frequencies comprised between 0.1 and 0.5. The SNP chips do not include QTN. Both SNP chips are nested, i.e., the largest chip includes all SNP present on the smallest one. The largest chip (chip 1) is used to perform gBLUP selection. Each chromosome has 100 QTN with normally distributed additive effects, and 50 QTN with Gamma-distributed additive effects. Dominance effects are simulated using dominance degrees sampled from a normal distribution with mean 0.5 and variance 0.1. QTN have minor allele frequencies comprised between 0.0 and 0.3 for the frequency-restricted QTN model (see Box 7). Gamma effects were simulated with shape and scale of 0.30 and 0.40, respectively.

Selection is conducted assuming that the traits are determined by the unrestricted set of QTN and have normally-distributed effects. By default, AlphaSim performs truncation selection, i.e., the best-performing individuals are selected. In the present example, truncation selection is carried out using genomic data through the gBLUP method. QTN are added to the selection SNP chip, which includes therefore 400 SNP and 200 QTN [2 chromosomes * (200 SNP/chromosome + 100 QTN/chromosome)]. The training population used to perform gBLUP selection contains 400 phenotypes randomly taken in the first generation of the pedigree. The training data are fixed across the selection generations to the training data used in generation 4. This means that the SNP effects estimated when running generation 4 will be used to compute the gEBV of the following selection generations, i.e., generations 5 and 6 in the present example. We asked to provide the data of average co-ancestry in each generation (this is possible for the selection generations only).

Two traits are simulated, with index weights equal to 0.7 and 0.3. These weights are used to integrate the gEBV computed for each of both traits in a selection index (see Box 12 for details). Because both additive and dominance effects are simulated, the broad-sense heritability is provided. The traits have a broad-sense heritability of 0.8 and 0.6, respectively, and an a priori genetic variance of 1.0 each. The 'true' genetic variance is computed as the variance of the simulated TBV in the base generation of the pedigree. The between-trait genetic and residual correlations are provided as triangular 2 x 2 matrices.

Regarding the output options, the "Chromosomes" folder is kept at the end of the simulating process, while the full sequence of all individuals included in the pedigree is written out, and files are zipped.

No genome compression is applied.

To run this parameters file, copy all the files from the software package in a given directory.

1.11 Background reading

1. Chen, G.K., Marjoram, P., Wall, J.D., 2009. Fast and flexible simulation of DNA sequence data. Genome Res. 19, 136–142. doi:10.1101/gr.083634.108
2. Falconer, D.S., Mackay, T.F.C., 1996. Introduction to quantitative genetics. London: Longman.
3. Hickey, J.M., Gorjanc, G., 2012. Simulated data for genomic selection and genome-wide association studies using a combination of coalescent and gene drop methods. G3 2, 425–427. doi:10.1534/g3.111.001297
4. Hudson, R.R., 2004. ms a program for generating samples under neutral models. Bioinformatics 18: 337-338.
5. Wray, N.R., Goddard, M.E., 1994. Increasing long-term response to selection. Genet. Sel. Evol. 26, 431. doi:10.1186/1297-9686-26-5-431

CHAPTER

TWO

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*