SNaP

— Program Documentation —

Version 3.0

Michael Nothnagel[1]

Christian-Albrechts-Universität zu Kiel
Intitut für Medizinische Informatik und Statistik (IMIS)
Universitätsklinikum Schleswig-Holstein
Kiel, Germany

e-mail: nothnagel@medinfo.uni-kiel.de

September 21, 2005

[1]Parts of this work were done at the Rockefeller University, Laboratory of Statistical Genetics, New York, NY, U.S.A., and at the Max Delbrück Center for Molecular Medicine, Department of Bioinformatics, Berlin, Germany

Contents

1 Software overview

1.1 Objective

The SNaP software generates data sets of sequences of single nucleotide polymorphisms (SNPs) and corresponding phenotypic expressions. It is assumed that the SNPs occur in independent blocks of linkage disequilibrium (LD).

Features The program allows for

- an unlimited number of unrelated individuals and nuclear families
- a random or fixed number of children in the families
- a random assignment of a phenotype conditional on the genotype at one or more SNP loci
- a quantitative trait (QT) or a disease affection status as phenotypic expression
- global, sex-specific, age-specific, and sex-age-specific penetrance and phenotypic mean values
- two sampling schemes
- a simple sequence design via LD blocks
- SNPs that are potentially mono-allelic or stringently bi-allelic
- several genotypic and haplotypic data output formats and other format options
- some post-processing goodies like the introduction of genotyping errors and the removal of causative SNPs

SNaP is fast, can easily be used with job-settings files, is portable to most computer platforms, and is adaptable to memory consumption. Several example files are provided for different study and sampling designs and phenotypes.

Rationale The software rests on two biological observations on the molecular level (see, for example, [1–6, 10, 12]):

- Given a SNP sequence, this sequence tends to fall apart into blocks with high degrees of linkage disequilibrium (LD) between SNPs from the same block and low or no degrees of LD between SNPs in different blocks.

- The blocks exhibit a limited haplotypic diversity, i.e. only a small fraction of block haplotypes is observed with notable frequency. An example would be a 5-SNP sequence where only 3 or 5 haplotypes out of possible $5^2 = 32$ occur.

So far, the algorithm makes no further assumptions on the LD structure within the blocks. LD between SNPs is only caused by the specification of the possible block haplotypes and their frequencies (see section 2.2 for details).

Data Examples Possible data sets that can be generated are:

- *Sib-pairs* of a certain age range and their parents with an affection status conditional on an additive single locus with low penetrance, but high susceptibility allele frequency, that acts differently in males and females.

- *Case-control data* with males of age 55 years and older under a recessive-multiplicative two-loci disease model using separate case-control sampling

- *Nuclear families of random size* with one up to ten children between 1 and 15 years old; a genomic sequence with several extended regions of LD, and a quantitative trait depending on three non-observable SNP in these regions

- *Genotypes only* without a phenotypic expression but with known underlying haplotypes, e.g. to be used with the evaluation of haplotype reconstruction methods

- *Haplotypes* containing various blocks of different sizes and with different number of block haplotypes, e.g. to be used in the investigation and development of LD measures

The software is projected to further improvements in the future.

1.2 Algorithm summary

SNaP generates data sets by going through the following steps:

1. A random haplotype is sampled from the pre-specified haplotype set for each block of the SNP sequence. Block haplotypes are concatenated to create the haplotype of the full sequence. This procedure is repeated to generate the *individual's genotype* (dual-haplotype).[1]

[1]So far, SNP alleles are coded 1 and 2. An extension to ACGT coding is projected.

If family data are generated, father and mother are sampled as independent individuals. Subsequently one haplotype is randomly chosen from the father and one from the mother to form a child's genotype.[2]

2. Each individual is randomly assigned *sex* and *age* according to user-specified distributions and limits.

3. If requested, a *phenotypic expression* is randomly assigned to the individual conditional on the genotypic state at one or more causative loci, according to the specified phenotypic model.[3] This assignment can optionally be made dependent on the sex and/or age of the individual.

4. Optionally, *noise* such as genotyping errors can be added to the genotypic observations.

5. Optionally, the causative SNPs can be *removed* from the generated data set. This is to model the frequent non-observability of causative SNPs in real-world samples which then have to be inferred by SNPs that in LD with these causative SNPs.

1.3 Job-settings file and data file: an example

The following job file, `example.set`, generates a sample of unrelated 200 individuals with genotypes of 8 SNPs in two blocks of sizes 5 and 3. The seventh SNP in the sequence, i.e. the second SNP in the second block, is a causative SNP with susceptibility allele frequency of 0.2. Three block haplotypes carry the susceptibility allele 2, but possibly more haplotypes of the full 8-SNP sequence. The causative SNP's genotype is used to define an individuals's affection status using a single-locus additive model with full penetrance with no influence of sex or age. Individuals who are homozygous for the susceptibility allele (2/2) will be affected with certainty whereas heterozygous individuals (1/2) are affected only with 50 % chance. The ratio of males to females in the sample will be approximately 1.07. The subjects will be between 1 and 100 years old, but predominantly around 25 years; the age distribution in the sample will be same for males and females and follow a Normal distribution. The sexes will also have identical disease patterns.

```
[General]                                  |
Datafile            = 'example.dat'        | [Design]
SettingsLogfile     = 'example.set'        | Phenotype          = 'qualitative'
OutputType          = 'haplotypes'         | StudyDesign        = 'individuals'
OutputFormat        = 'Standard2'          | SamplingDesign     = 'joint'
```

[2]Recombination at this stage is not yet implemented.

[3]A software extension for a haplotypic phenotype definiton is projected.

```
NumberOfIndividuals    = 200

[SexAge]
SexRatio               = 1.07000
AgeMinimum             = 1
AgeMaximum             = 100
AgeDistribution        = 'normal'
AgeMean                = 25.00000
AgeStd                 = 10.00000

[Model]
NumberOfLoci           = 1
NumberOfStates         = 3
BiallelicCheck         = 'y'
HideCausalSNPs         = 'n'
GenotypingErrorRate    = 0.00000
GenotypingErrorVisible = 'y'
RandomSeed             = 2150
ModelType              = 'homogeneous'
Penetrances            =
    1.00000     0.50000     0.00000

[Separators]
BehindPhenotype        = '\t'
BetweenHaplotypes      = '\t'
BetweenBlocks          = ' '
BetweenSNPs            = '.'

[Blocks]
NumberOfBlocks         = 2

(Block)
Number                 = 1
NumberOfMarkers        = 5
SuscLocusPosition      = 0

{NoSuscHaplotypes}
HtNumber               = 3
HtBlock                = 11222
HtBlock                = 21121
HtBlock                = 22112
HtFrequ                = 0.70000
HtFrequ                = 0.10000

(Block)
Number                 = 2
NumberOfMarkers        = 3
SuscLocusPosition      = 2
SuscAlleleFrequ        = 0.20000

{NoSuscHaplotypes}
HtNumber               = 2
HtBlock                = 111
HtBlock                = 211
HtFrequ                = 0.60000

{SuscHaplotypes}
HtNumber               = 3
HtBlock                = 122
HtBlock                = 221
HtBlock                = 222
HtFrequ                = 0.30000
HtFrequ                = 0.50000
```

The data file `example.dat` is created by typing `snap example.set` at the command prompt and might look as follows:

```
1       1       0       0       1    5 1   1.1.2.2.2 2.1.1   1.1.2.2.2 2.2.1
2       1       0       0       2   27 1   1.1.2.2.2 1.1.1   2.2.1.1.2 2.1.1
3       1       0       0       1   43 2   1.1.2.2.2 2.1.1   2.2.1.1.2 2.2.1
4       1       0       0       1   34 2   1.1.2.2.2 1.2.2   1.1.2.2.2 2.2.1
5       1       0       0       2   20 1   1.1.2.2.2 1.1.1   1.1.2.2.2 1.2.2
6       1       0       0       2   23 1   1.1.2.2.2 2.1.1   1.1.2.2.2 1.1.1
7       1       0       0       1   22 2   1.1.2.2.2 2.2.2   1.1.2.2.2 2.1.1
8       1       0       0       1   41 1   1.1.2.2.2 1.1.1   1.1.2.2.2 2.2.2
9       1       0       0       2   19 2   1.1.2.2.2 2.1.1   1.1.2.2.2 2.2.1
10      1       0       0       1   49 1   1.1.2.2.2 1.1.1   1.1.2.2.2 2.1.1
11      1       0       0       1   29 2   1.1.2.2.2 1.1.1   2.1.1.2.1 2.2.2
:       :       :       :       :    : :     :       :       :       :
```

1.4 Where to go from here

Sections 3 and 4 give details on the software's usage. To compile the source code or adapt the program to special settings, see section 5. Section 2 gives a description of the underlying algorithm.

1.5 Citation & Feedback

When using the software for publication, please cite the following reference:

> Nothnagel M (2002). Simulation of LD block-structured SNP haplotype data and its use for the analysis of case-control data by supervised learning methods. *Am J Hum Genet* **71**(4) Suppl.: A2363.

Comments and error feedback are appreciated. Please send an e-mail to:

> `nothnagel@medinfo.uni-kiel.de`, subject: `SNaP`.

1.6 License

The software is distributed under the **GNU General Public License** (see `http://www.gnu.org/` for details):

2 Description of the algorithm

2.1 Summary

Basic assumptions The implemented algorithm makes the following assumptions:

- The SNPs of a sequence occur in *independent disjoint blocks*:
    - There are potentially high degrees of linkage disequilibrium (LD) between SNPs within a block.
    - There is no LD between SNPs in different blocks, i.e. blocks are independent.

  This is appropriate to model genomic regions on different chromosomes or in distant areas on the same chromosome.

- The phenotypic expression is defined by one or more *causative SNPs* (*susceptibility loci*) . Every causative locus therefore possesses exactly two alleles.

- Each block can contain at most one causative locus. Thus the software cannot used to model causative SNPs that are in LD with each other, e.g. two causative SNPs within a small gene.

Data design options Several choices have to be made for the data set generation:

- *Sequence layout*: The user has to specify how the SNP sequence is composed of blocks and where the causative SNPs reside. Everything between a number of single-SNP "blocks" and a single block containing various SNPs is possible. The specification includes the number of blocks, their size, their number of block haplotypes and optionally their alleles and frequencies.

- *Study design*: SNaP can generate unrelated individuals, e.g. cases and controls, or nuclear families with a fixed or random number of children.

- *Sampling design*: For a binary phenotype such as affection status, cases and controls can, but do not have to, be sampled separately. Otherwise a specified number of individuals or families is generated (joint sampling).

- *Sex and age*: Both study designs require a specification of the distributions of sex and age in the sample.

- *Type of phenotypic expression*: Individuals can either be assigned a qualitative phenotype (affection status), a quantitative trait (QT) value, or no phenotype at all. The phenotype is defined conditional the genotypes of the causative SNPs.

- *Disease model*: Each genotypic state corresponds to an affection probability (penetrance) or a mean QT value, respectively, that have to be specified by the user. The specification can be homogeneous, sex-specific, age-specific, or both.

- *Genotypic output*: Finally, one can decide on the output format of the genotypes, if typing errors are to be introduced, and if the genotype of the causative SNPs should be withheld from the output file.

Data generation algorithm Before the data set is generated, SNaP looks for incomplete specifications in the job settings. Non-specified or only partially specified blocks haplotypes are randomly completed by drawing from the set of the remaining possible haplotypes according to the uniform distribution. This can result in sets whose haplotypes are more distantly related on average than they would under some coalescence model. In this course, the optional requirement of SNPs to be bi-allelic further narrows the set of possible block haplotypes to those where each SNPs allele occurs in at least one haplotype. Block haplotypes with non-specified frequencies are assumed to equally frequent.

The data generation algorithm for a single individual is as follows:

1. A *genotype* (dual-haplotype) for an individual is randomly generated. This is done by repeating the following steps two times: First sampling haplotypes from the specified set of possible haplotypes according to their probabilities for each block and second concatenating the sampled block haplotypes to create the full sequence haplotype.
   Genotypes for father and mother in nuclear families are independently generated as described above. For each child, one haplotype is randomly chosen from the father and one from the mother to form the offspring genotype.

2. Each individual is randomly assigned *sex* and *age* according to the specified distributions and limits.
   In nuclear families, these specifications describe the children's distribution; the parental ages are randomly assigned according to additional parameters.

3. If requested, each individual is randomly assigned a *phenotypic expression* of either

   - an affection status (affected or not affected) according to the specified penetrance (affection probability) of his or her genotype,
   - a quantitative trait (QT) value that is drawn from a Gaussian distribution with the specified QT mean for his or her genotype and a homoscedastic variance, or
   - no phenotype.

   Penetrances, QT means and QT variances can be specified homogeneously for the whole population, sex-specific, age-specific, and sex-age-specific.

4. If requested, *genotyping errors* or *missing genotypes* are introduced into the genotypic "observations" as a representation of noise in the data. SNPs are then randomly flipped to the other allele or marked unknown according to some specified probability. Noise is added after a potential phenotype definition.

5. If requested, the genotypes of *causative SNPs* used in the definition of the phenotype can be *withheld* from the SNP sequence output.

The SNP allele frequencies are in *Hardy-Weinberg equilibrium* by definition when joint sampling is specified (see section 3.2.3), because the haplotype frequencies completely determine the genotype frequencies. Furthermore, there is no or only random association (LD) between SNPs within different blocks. This includes all causative SNPs.

The algorithm will proceed until the requested totals of individuals, families, or cases/controls are generated. In the latter case, additional potential members for a group will be discarded if the size of this group already equals the requested total.

2.2 Genotypes: structure & generation

The basic data object to be generated is haplotypes of single nucleotide polymorphisms (SNPs). They can be thought of as being extracted from the general chromosomal nucleotide sequence of the usually two human chromosomes (see figure 1).[4] The SNP sequence falls apart into several independent blocks (no or low degrees of LD between SNPs from different blocks) which are treated and sampled separately. To generate a random haplotype of the SNP sequence, block haplotypes are randomly drawn from a specified set according to their specified probabilities and concatenated to yield the complete sequence haplotype.

[4] The number of homologous chromosomes (i.e. the number of haplotypes per person) is assumed to be 2. Other numbers can be specified (see section 3.2.5) although the phenotype definition algorithm in the program needs to be adapted subsequently. This could be useful for trisomic disorders of the genome such as the Down syndrome.

2.2.1 Parametrization

Block parametrization Suppose the SNP sequence consists of b independent blocks. Every block is characterized by:

- a block position, i, within the blocks sequence (block number $i \in \{1, \ldots, b\}$)
- a block size, b_i, i.e. the number of SNPs within the block
- the position of a possible susceptibility locus within the block $l_i \in \{0\} \cup \{1, \ldots, b_i\}$ and the frequency of its susceptibility allele $p_i^s \in [0, 1]$
- the set of haplotypes that the block can assume (see below).

The numbering of blocks along the sequence and of SNPs within the block goes from left to right.

The block size can assume values between 1 and some technically limited number (see section 3.2.8).[5] If the block does not contain a susceptibility locus, the locus position and the susceptibility allele frequency equal zero: $l_i = 0$, $p_i^s = 0$.

A block can only assume a limited number of pre-set block haplotypes, the so-called "observed" haplotypes which occur with frequencies greater than zero. The observed block haplotypes have to be specified as parameters, whether set by the user or randomly generated by the program. If a block contains a susceptibility locus, two groups of haplotypes must be specified that contain the pre-set haplotypes with and without the susceptibility allele at the locus. Otherwise, only the second group is needed. See Figure 1 for an illustration of the parametrization.

Block haplotypes parametrization Suppose, the i-th block under consideration contains a susceptibility locus (causative SNP). In this case, the set of m_i haplotypes is divided into two groups: The *susceptibility group* (marked by the superscript "s") includes the m_i^s haplotypes that have the susceptibility, or mutated, allele at the locus ($\{H_{i1}^s, \ldots, H_{im_i^s}^s\}$), while the *non-susceptibility group* (marked by the superscript "n") consists of the m_i^n haplotypes with the non-susceptibility, or wild-type, allele ($\{H_{i1}^n, \ldots, H_{im_i^n}^n\}$), with $m_i = m_i^s + m_i^n$. The probabilities to observe a haplotype from these two groups are given by the susceptibility allele frequency: p_i^s and $p_i^n = 1 - p_i^s$, respectively.

Within both groups, each haplotype has some probability to occur:

$$h_{ij}^k = \mathrm{P}(H_{ij}^k) \quad (j = 1, \ldots, m_i^k,\ k = "s", "n")\,.$$

[5]So far, the program does not model any form of linkage disequilibrium (LD) between SNPs within a block explicitly. Especially for larger block sizes the program is projected to incorporate some form of sophisticated LD modelling in the future.

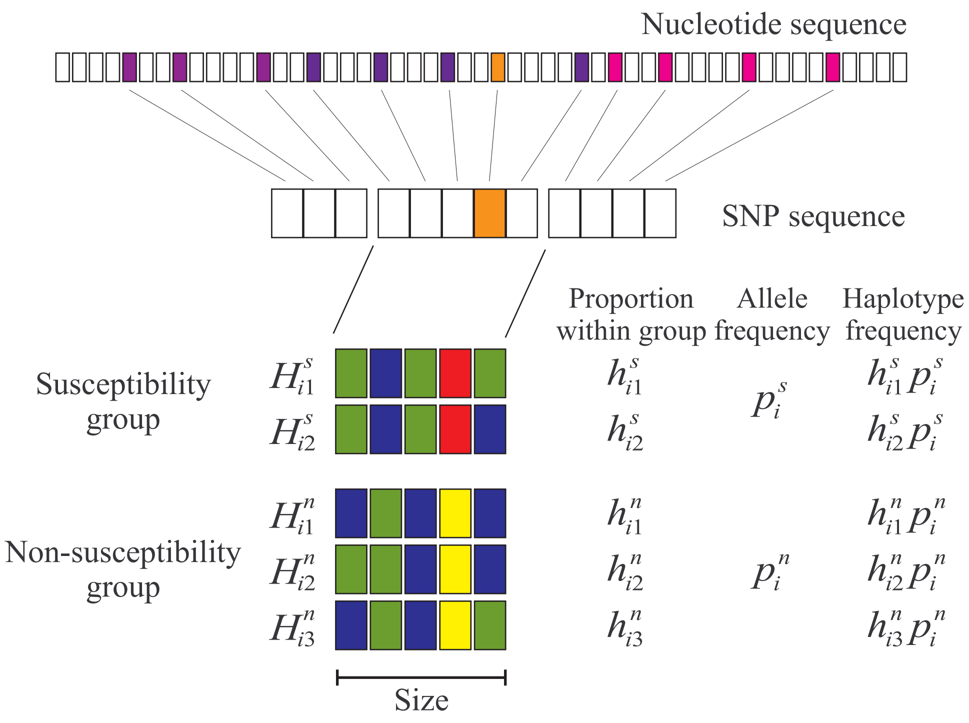

Figure 1: Scheme of block parametrization.

Since these haplotype frequencies $h_{ij}^{\cdot}$ refer to the group, they sum to 1 in each of the two groups:

$$\sum_{j=1}^{m_i^k} h_{ij}^{\cdot} = 1 \quad (j = 1, \ldots, m_i^k,\, k = "s", "n")\,.$$

The frequency of a particular haplotype in the population is given by the product of the corresponding allele frequency times the within-group probability: $p_i^{\cdot}\, h_{ij}^{\cdot}$. Haplotypes that are not included in the description by the two groups have a population frequency of zero.

If there is no susceptibility locus in a particular block, then the susceptibility group is simply discarded $(m_i^s = 0)$ and, thus, $p_i^n = 1$ and $m_i = m_i^n$. The group-specific haplotype probabilities then equal the population frequencies.

2.2.2 Completion of missing information

Block haplotype alleles and probabilities do not need to be specified. Before data are generated, SNaP looks for incomplete specifications in the job settings and automatically completes information on them.

Generation of missing SNP alleles and haplotypes Partially or completely missing block haplotypes are randomly generated by SNaP according to the uniform distribution. To this aim it is assumed that every SNP allele has the same probability of occurrence in the pre-set haplotype alleles and that the SNPs are independent from each another. Thus, every block haplotype is selected for the pre-set with equal probability (conditional on the portion of SNP alleles already specified). This can result in sets whose haplotypes are more distantly related on average than they would under some coalescence model.

Check of SNPs to be bi-allelic SNPs often differ in frequency and even occurrence between populations and between samples. This means that SNPs are potentially mono-allelic in a sample. The program can check for mono-allelic SNPs in the block pre-set haplotypes. When single SNPs or whole pre-set haplotypes are left open to be randomly generated by SNaP, these SNPs can be allowed to be mono-allelic or can be forced to be bi-allelic. Requiring the SNPs to be bi-allelic in the pre-set block haplotypes narrows the set of possible block haplotypes to those where each SNPs allele occurs in at least one haplotype.

Missing haplotype frequencies Block haplotypes with non-specified probabilities are assumed to equally frequent within the (non-)susceptibility group. The completing haplotype probabilities are chosen in such a way that the total sum of haplotype probabilities within a group equals one.

2.3 Assignment of sex and age

Sex and age are randomly and separately assigned to an individual. Thus both sexes follow the same age distribution, except for parents in nuclear families.

2.3.1 Sex in the sample

The distribution of males and females in a sample of unrelated individuals is specified by the sex ratio males to females:

$$r_{sex} = \frac{m}{f} \quad .$$

The proportions of males and females in the sample are thus:

$$m = \frac{r_{sex}}{1 + r_{sex}} \quad \text{and} \quad f = \frac{1}{1 + r_{sex}} \quad .$$

If nuclear families are generated, the sex ratio relates to the children; the sex of the parents is obviously assigned. To create samples of only a single

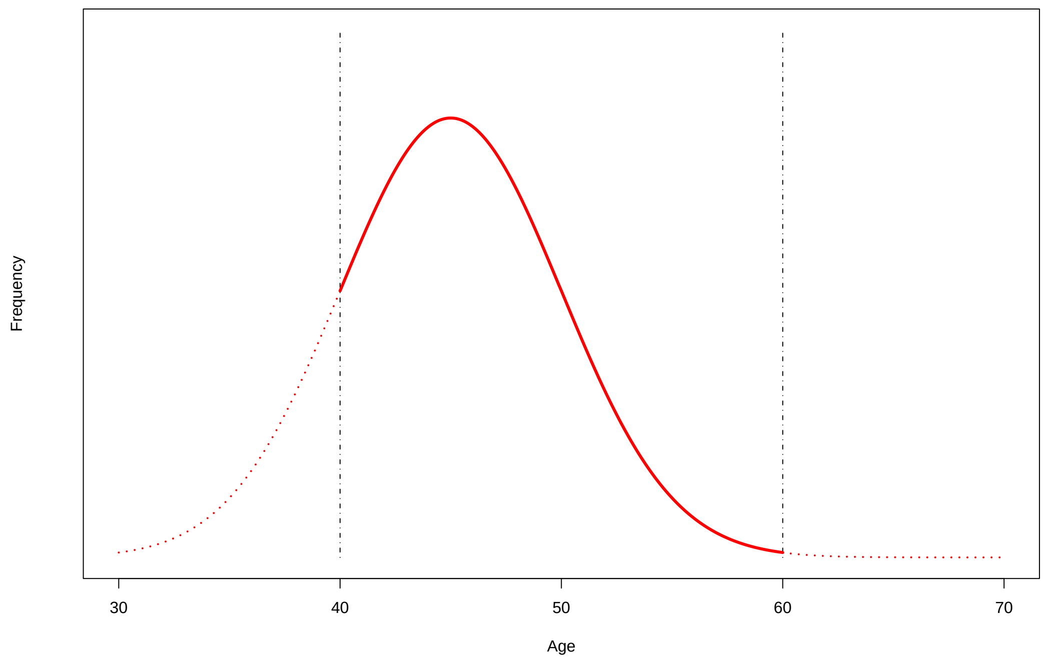

Figure 2: **Scheme of a sample age distribution.** The graph illustrates the frequency of the valid values (solid line) for a Normal age distribution with mean 45 years and a standard deviation of 5 years and an age range from 40 through 60 years.

sex, extreme values for r_{sex} should be used; however a subsequent control of the generated data is mandatory.

2.3.2 Age assignment

If unrelated individuals are sampled, the random age assignment to an individual is determined by the upper and lower limit for the age (age range) and by the age distribution. SNaP offers currently two distributions: With the uniform distribution, each value within the age range has equal probability of being assigned. With the Normal distribution, the mean value and the variance have additionally to be specified. Still only values within the age range are valid, see Figure 2.

If nuclear families are sampled, the specifications made above apply for the children. A minimum age difference between the children has now also to be specified, usually equalling one year. Furthermore, the mean and the standard deviation for the mother's age at birth of her first child and the mean and the standard deviation for the parental age difference father–

mother have to be specified. The age of the mother and of the father are randomly assigned using a Normal distribution with the specified parameters.

2.4 Phenotype assignment conditional on the genotype

Individuals can be assigned a phenotype. The phenotype can either be qualitative, e.g. an affection status, or quantitative, i.e. some quantitative trait (QT). The assignment is made conditional on the genotypes of the causative SNPs, the *genotypic state.* It is *not* dependent on some allelic combination at a particular *haplotype.* SNaP can also be used without the assignment of a phenotype.

2.4.1 Determination of the genotypic state

The algorithm requires the genotype to consist of exactly two haplotypes and the loci to have exactly two alleles. Thus, every locus is in one of three allelic states:

Genotype at a single locus	Copies of susc. allele	*Index*
Homozygous susc./mutated	2	1
Heterozygous	1	2
Homozygous non-susc./wild-type	0	3

Now let there be $L \in \{1, \ldots\}$ loci. The genotype on these loci combined can then assume 3^L different allelic combinations, or *genotypic states.* The number of states evolves exponentially with an increasing number of loci:

Number of loci:	1	2	3	4	5	6	7	...
Number of states:	3	9	27	81	243	729	2187	...

2.4.2 Phenotype assignment

Affection status assignment The probability of affection given the genotype is defined as the penetrance

$$a_g = \mathrm{P}(y \mid g) \ .$$

where y denotes the affection status and g the genotypic state. For a single-locus model ($L = 1$), there are exactly three penetrances in the model: the affection probability for a homozygous genotype with the susceptibility allele, a_1, the probability with the heterozygous genotype, a_2, and the homozygous wild-type genotype's probability, a_3. Thus the penetrance vector

$$(a_1, a_2, a_3)$$

describes the affection probabilities when 2, 1, or 0 susceptibility alleles are present in an individual.

For example, a *dominant* disease model is described by $a_1 = a_2 > a_3$, e.g. $(a_1, a_2, a_3) = (1, 1, 0)$ for full penetrance. A *recessive* model is described by $a_1 > a_2 = a_3$, e.g. $(a_1, a_2, a_3) = (0.9, 0, 0)$ for reduced penetrance. $a_1 > a_2 > a_3$ specifies an *intermediate* model. A general formulation for an *additive* model is $(a_1, a_2, a_3) = (2\alpha + \beta, \alpha + \beta, \beta)$, while a *multiplicative* model is described by $(a_1, a_2, a_3) = (\gamma^2 + \beta, \gamma + \beta, \beta)$.

Now consider two loci ($L = 2$) and accordingly nine penetrances a_{11}, a_{12}, a_{13}, a_{21}, ..., a_{33}. Causative SNPs are numbers from left to right along the sequence. The penetrances can handily be arranged in matrix form:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

The first line gives the penetrances for two copies of the susceptibility allele at the first causative SNP (further left) and 2, 1, and 0 copies, respectively, of the susceptibility allele at the second causative SNP (further right), and so on.

Consider these three examples:

$$\begin{pmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \end{pmatrix} \begin{pmatrix} 0.7 & 0.7 & 0.7 \\ 0.7 & 0.0 & 0.0 \\ 0.7 & 0.0 & 0.0 \end{pmatrix} \begin{pmatrix} 0.70 & 0.50 & 0.40 \\ 0.35 & 0.25 & 0.20 \\ 0.05 & 0.05 & 0.05 \end{pmatrix}$$

The left example specifies a completely recessive model where the disease only occurs if and only if the individual is homozygous for the susceptibility allele at both causative SNPs. The middle example describes a model in which two loci independently and with the same probability cause the disease. Each locus could separately be considered as following a recessive model. The right example models a major effect at the first locus, amplified by an additional effect at the second locus, and some hidden sources like environmental variance or locus heterogeneity.

The general order of penetrances is as follows:

$$\begin{array}{lll} a_{11...111} & a_{11...112} & a_{11...113} \\ a_{11...121} & a_{11...122} & a_{11...123} \\ \cdots & & \\ a_{13...331} & a_{13...332} & a_{13...333} \\ a_{21...111} & a_{21...112} & a_{21...113} \\ \cdots & & \\ a_{33...331} & a_{33...332} & a_{33...333} \end{array}$$

The penetrance values for every genotypic state, $(a_1, \ldots, a_{3^L})$, have to be specified by the user.

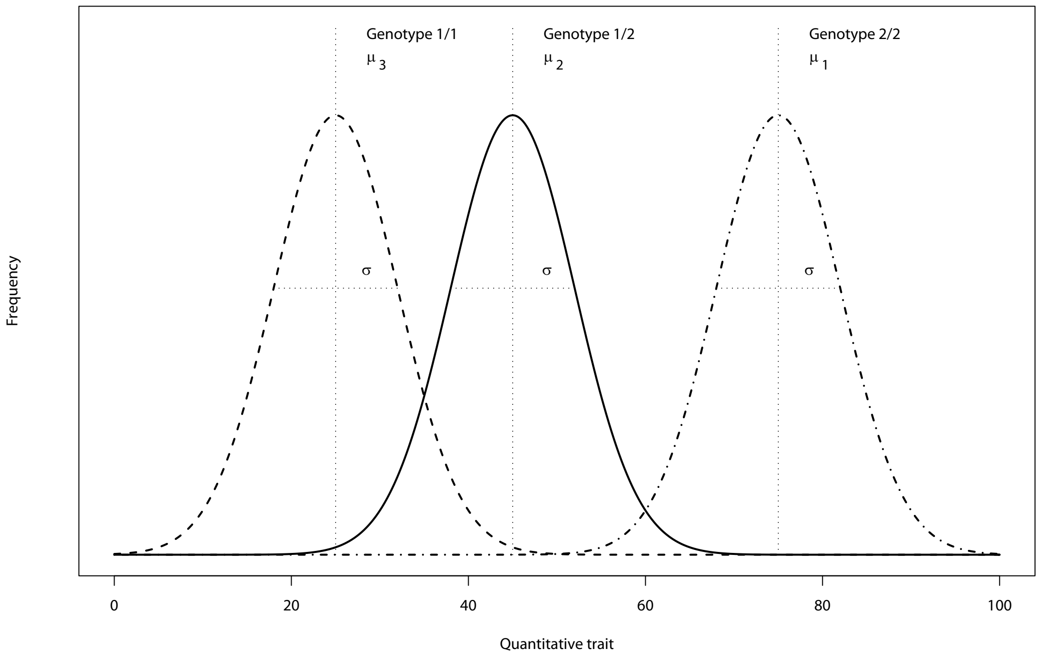

Figure 3: **Scheme of QT value assignment.** Illustration of the QT value distribution for a single-locus model. For each genotype the QT values follow a Normal distribution with homoscedastic variance. The mean value for two copies of the susceptibility allele is μ_1 whereas the mean for the wildtype genotype is μ_3!

Quantitative trait assignment The assignment of a quantitative trait (QT) to an individual is quite similar to assigning an affection status. Each genotypic state corresponds to a mean value of the QT. The actual value for the individual is drawn from a Gaussian distribution[6]:

$$y = \mathrm{N}(\mu_l, \sigma)$$

where μ_l designates the mean value of the l-th genotypic state and σ the homoscedastic phenotypic variance. See Figure 3 for an illustration.

The mean values for every genotypic state, $(\mu_1, \ldots, \mu_{3^L})$, and σ have to be specified by the user in the same order as the penetrances. The QT values assume exactly the specified means by setting $\sigma = 0$.

Sex and age SNaP allows for different genotypic effects between males and females and between different age groups. Penetrances, mean values, and variance can be specified homogeneously, i.e. applying for all individuals regardless of sex and age, specific for both sexes, dependent on an age interval, and sex-and-age specific.

[6]The program uses the function `gasdev()` from [9, ch. 7.2.].

2.5 Postprocessing

2.5.1 Genotyping errors

Genotyping and others errors do occur in reality and can affect the power of gene mapping studies [7]. To account for this phenomenon, SNaP offers the feature of introducing errors into the genotypes *after* a potential phenotype was assigned.

If requested, every SNP in the sequence has the user-specified probability $e > 0$ of being altered to another allele[7] or being marked unknown, depending on the choice of the user. More sophisticated methods of error simulation may be included into future versions of the program.

2.5.2 Removal of causative SNPs

Causative SNPs are frequently unobservable or not included in the sample and can only be inferred by sample SNPs that are in LD with them. To model this situation, SNaP offers the option to withhold the causative SNPs from the data output. In this way, the direct source for the phenotype definition is hidden from further data analyses.

[7]For the usual case of exactly two alleles per SNP, the SNP is simply altered to the other allele. The replacement algorithm cycles through the set of possible alleles of the SNP excluding the one actually present to find the new allele. If, for example, 3 alleles are present at the SNP, 1 is altered to 2, 2 to 3, and 3 to 1. If the maximum number of SNP alleles is set to 2 (see section 3.2.8) this effectively switches the locus to the other allele. Randomly drawing from the remaining alleles might be included in future versions of the program.

3 Software usage

3.1 Program call & options

The program can be called with several options:

	Option	*Result*
`snap`		List options
`snap`	`[-s] <jobfile(s)>`	Process job-settings file(s), generate data
`snap`	`-e`	Generate example job-settings files
`snap`	`-p`	List limits and job-setting phrases

More than one job-settings file can be passed on to the program. For example, to process all job-settings files in the current directory with suffix `.job`, simply enter '`snap *.job`' at the command line.

In case the program has to be **compiled** before its use or if you experience memory problems, please see section 5.1 for assistance.

The silence switch -s Use the `-s` switch to suppress feedback from the program except for error messages. Usually, the programs gives information about its current operations.

The example option -e Use the `-e` option to create several example job files. The produced files contain the correct designators for the parameter settings and give you an impression how a job file will look like. For your convenience, prepare your own job files by modifying these example files.

The following example files are generated:

Option	*Job file for*
`example1.set`	Case-control separate sampling under a two-loci model, equal numbers of males and females aged 40-75 years homogeneous penetrances, haplotypes in Standard 1 format
`example2.set`	Unrelated-individuals sampling under a single-locus affection model, higher percentage of males, ages normally distributed around 25 years, homogeneous penetrances, haplotypes in Standard 2 format
`example3.set`	Unrelated-individuals sampling with a quantitative trait (QT) dependent on two loci, no errors, compact genotypic output
`example4.set`	Nuclear families with 4 children each, aged 1-30 years, a single-locus QT, non-detected genotyping errors, haplotypes in Linkage format
`example5.set`	Nuclear families with 1-6 children each, aged 5-20 years, a sex-specific affection status definition dependent on a single locus, non-detected genotyping errors, haplotypes in Linkage format
`example6.set`	Nuclear families with 1-6 children each, aged 5-20 years, a age-specific affection status definition dependent on a single locus, non-detected genotyping errors, haplotypes in Linkage format
`example7.set`	Nuclear families with 1-6 children each, aged 5-20 years, a sex-and-age-specific QT in a single QTL, non-detected genotyping errors, haplotypes in Linkage format

You can use these example files to generate data file examples. The generated data and the parameter settings used will be written to the files specified in the job-settings file. See section 3.2 for the parameter specification.

The parameter listing option -p The purpose of the `-p` option is twofold. If you want to know what the actual phrases are to be used to specify the job settings, e.g. a particular study design, the type of phenotypic expression, or how the susceptibility allele and affection status are coded, use this option.

If you have experienced problems, e.g. regarding memory consumption or "strange behavior" by the software, e.g. because not all lines in the job file were read into the program, you might want to look for internal parameter limits of the compiled version, e.g. how many lines are allowed in a job-settings file or in a parameter group. Some modification of these settings might be necessary to obey the limitations of your computer or to enable larger block numbers and haplotype numbers. See section 5 for details. For an output example produced by using the `-p` option, see section 3.3.2.

3.2 Parameter specification

All parameters for generating a data set have to be specified in a text file, the *job settings file*. Its name is passed to the program as an argument. The parameters in the file are bundled into several groups; each group starts with a group designator. There are also subgroups with their own designators. Each parameter is set by its designator, followed by the assignment character ("`=`" by default) and the value.

Please observe:

- Designators, assignment character, and values have to be separated by *spaces*.
- The names of the designators are *case-sensitive*.
- Character and string specifications have to be *enclosed* into `''`.

You can use the example option when calling the program (see section 3.1) to produce example settings files. These files contain the parameter and group designators and some value settings in the correct syntax. For an example of a job file see section 3.3.1.

The specified parameters are checked by the program. If possible, missing parameters are generated (e.g. haplotype frequencies, block haplotype pre-settings). Otherwise the program will be terminated. See section 2.2 for the data parameterization. The program will give warnings about insufficient or modified parameters.

Phrases and parameter designators can be changed by re-compilation (see section 5.2.1). The designators mentioned in this documentation refer to the release version of the program.

3.2.1 Job file structure

The job-settings file consists of the following parameter groups:

- General settings
- Design settings
- Sex and age specifications
- Model specification
- Separator settings

- Blocks specification

The order of the groups in the file is irrelevant.

3.2.2 General job parameters

The general parameter group contains general specifications like the number of cases and controls to be generated and the names of the files to where the data should be written. The group might look like this:

```
[General]
Datafile          = 'example.dat'
SettingsLogfile   = 'example.set'
OutputType        = 'haplotypes'
OutputFormat      = 'Linkage'
```

DataFilename & SettingsFilename `DataFilename` specifies the file to where the generated data should be written. Since parameters might be changed during checking, `SettingsFilename` specifies the file where the parameter settings actually used for the data generation are saved.

OutputType The `OutputType` option specifies, whether genotypic data (`'genotypes'`) or haplotypic data (`'haplotypes'`) should be printed to the data file.

OutputFormat This specifies the data format in the output. There are the following options for *haplotypes*:

- `'Standard1'` causes the haplotypes to be printed below each other.
- `'Standard2'` will result in a single line per individual, listing the haplotypes one after another on the same line.
- `'Linkage'` will arrange the data in linkage format ([13]). Alleles of the first haplotype will always appear ahead of the corresponding allele of the second haplotype. Thus the haplotypic structure can eventually be recovered.

For *genotypes*, there are two options:

- `'Compact'` causes the genotypes at the SNP loci to be coded in a compact from as being homozygous or heterozygous for the particular alleles (e.g. '1', '2', or '3').
- `'Linkage'` will arrange the data in linkage format ([13]). In contrast to the haplotypic output, the two alleles of a particular SNP from the haplotypes will be sorted before print-out. Thus the information on the haplotypic phase is irrecoverably lost.

See section 4.1 for example outputs.

Attention: The program does *not* check if the specified target data files already exists. Any existing file will be overridden!

3.2.3 Phenotype, study design, and sampling design

The design parameter group describes the type of the phenotypic expression, the study design, the sampling design, the number of individuals or families to be generated, and the kind of families to be generated. The start of the group will look like this:

```
[Design]
TypeOfPhenoExpression  = 'qualitative'
StudyDesign            = 'individuals'
...
```

Depending on the values of these parameters, further specifications have to be made (see below).

Phenotype Setting `Phenotype` to `'qualitative'` causes SNaP to produce a qualitative phenotype, e.g. an affection status, as output. Setting it to `'quantitative'` produces quantitative trait (QT) values as output. By setting it to `'none'`, the software generates genotypic data without a phenotype.

StudyDesign The parameter `StudyDesign` can be set to `'individuals'`, resulting in unrelated individuals as data units, or to `'family'`, generating families that can be used with family-based analysis methods, e.g. the TDT-based test family (Spielman et al. [11] and others).

SamplingDesign If an unrelated-individuals study design was chosen and the phenotype is a qualitative one, cases and controls can be sampled either separately or jointly. The statement for this looks like this:

```
[Design]
...
SamplingDesign  = 'separate'
...
```

Using the `'joint'` argument instead causes SNaP to sample both groups jointly. For low susceptibility allele frequencies and low penetrances, this might result in only a few cases in a bulk of controls.

Joint sampling is always assumed for a quantitative phenotype or when no phenotypic expression was required.

NumberOfIndividuals, NumberOfCases & NumberOfControls For separate or joint sampling, the required numbers of individuals or cases & controls to be generated must be specified. This is done with the following statements:

```
[Design]
...
NumberOfIndividuals  = 1000
...
```

```
[Design]
...
NumberOfCases        = 400
NumberOfControls     = 600
...
```

NumberOfFamilies, NumberOfChildren & TypeOfFamily If the study design was chosen to be family-based, the required number of families to be generated, the number of children per family, and the type of family must be specified. The statement for this looks like this:

```
[Design]
...
NumberOfFamilies  = 500
NumberOfChildren  = 2
TypeOfFamily      = 'fixed'
...
```

If `TypeOfFamily` was set to `'fixed'`, every generated family will have exactly as many children as specified. If it is set to `'random'` instead, each family will have the specified number at most, while the actual number is drawn from an uniform distribution[8] over {1,...,`NumberOfChildren`}.

3.2.4 Sex & age parameters

The sex & age group specifies the distribution of males and females in the sample and the distribution of the age.

SexRatio This ratio specifies the distribution males and females, either among the unrelated individuals or among the children under a family design. The statement looks like this:

```
[SexAge]
SexRatio                  = 1.07000
...
```

A ratio of 2 will result in as twice as many male individuals or children, respectively, than female ones. For samples of only one sex one should specify extreme values for the ratio. However, even then individuals of

[8]The software is projected to offer other distributions in the future.

the other sexes could occur and, thus, a control of the generated sample is necessary.

The age specification requires a number of statements. The statements `AgeMinimum`, `AgeMinimum`, and `AgeDistribution` are always required. The first two specify the limits for the age of individuals under unrelated-individuals design or of the children under the family design in the sample. `AgeDistribution` can either be `'uniform'` or `'normal'`. In the latter case, `'AgeMean'` and `'AgeStd'` have additionally to be specified. These statements might look like these:

```
[SexAge]
...
AgeMinimum           = 5
AgeMaximum           = 20
AgeDistribution      = 'normal'
AgeMean              = 12.00000
AgeStd               = 4.00000
...
```

Under the family design, additional parameters need to be specified. The age of the mother at birth of her first child is randomly drawn from a Normal distribution with mean `MotherAgeFirstChild` and standard deviation `MotherAgeStd`. The age difference between the father and the mother is randomly drawn as well according to a Normal distribution with parameters `ParentalAgeDiff` and `ParentalAgeDiffStd`. A example for first parenthood with mothers aged 27 years on average and on average 28-years old fathers is as follows:

```
[SexAge]
...
MotherAgeFirstChild  = 27.00000
MotherAgeStd         = 3.00000
ParentalAgeDiff      = 1.00000
ParentalAgeDiffStd   = 3.00000
...
```

Note that the age of the parents in the sample is their parental age *plus* the age of their oldest child.

The following statement declares the minimum interval in age between two children in a family:

```
[SexAge]
...
MinimumChildAgeDist  = 1.00000
```

Attention: Large values for this parameters, together with a large number of children per family in a small age range can lead to contradicting conditions during sampling.

3.2.5 Model parameters

The model parameter group specifies the parameters of the underlying genetic model for the phenotypic expression, like the number of haplotypes, the number of susceptibility loci, and the penetrances or mean genotypic effects. For an affection status dependent on two loci and the sex of the individual, the group might look like this:

```
[Model]
NumberOfLoci               = 2
NumberOfStates             = 9

ModelType                  = 'sex-specific'

(MalePhenotype)
Penetrances                =
  1.000   0.800   0.100
  0.500   0.400   0.100
  0.100   0.100   0.100

(FemalePhenotype)
Penetrances                =
  0.800   0.500   0.100
  0.400   0.250   0.100
  0.100   0.100   0.100

BiallelicCheck             = 'y'
HideCausalSNPs             = 'n'
GenotypingErrorRate        = 0.05000
GenotypingErrorVisible     = 'y'
RandomSeed                 = 500
```

NumberOfLoci The value of `NumberOfLoci` specifies the number of causative SNP (loci) in the SNP sequence. It is cross-checked with the sum of specified loci in the blocks settings (see section 3.2.7). The value for `NumberOfStates` is calculated by the program and gives the number of possible allelic combinations at all loci (genotypic states, see section 2.4.2); its specification is irrelevant.

ModelType The statement `ModelType` can assume the values `'homogeneous'`, `'sex-specific'`, `'age-specific'`, and `'sex-age-specific'` and declares that genotypic-specific penetrances or QT means, respectively, are either homogeneous (independent of sex and age), sex-specific, age-specific, or both.
`NumberOfStates`

Penetrances The *state* of the genotype at a particular locus is coded as 1, 2, or 3 if the locus is homozygous for the susceptibility allele, heterozygous for both alleles, or homozygous for the non-susceptibility allele, respectively (see section 2.4.2). According to the number of states the genotype can assume, an equal number of penetrances must be set. Penetrance as the probability of a particular genotype to cause an affection status ranges between the values 0 and 1. Penetrances have to be specified in the following order:

- If there is only one susceptibility locus, the sequence is simply:
  a_1 a_2 a_3
- For two loci, the first (left) locus is fixed at first and the second (right) locus assumes all possible allelic states subsequently. Then the first locus assumes its second allelic state while the second locus assumes all its states et cetera:
  a_{11} a_{12} a_{13} a_{21} a_{22} a_{23} a_{31} a_{32} a_{33}
- The general order to specify the penetrances is:
  $a_{1...111}$ $\cdots$ $a_{1...113}$ $a_{1...121}$ $\cdots$ $a_{1...133}$ $a_{1...211}$ $\cdots$ $a_{3...323}$ $a_{3...331}$ $a_{3...332}$ $a_{3...333}$

For better readability of the job file, the penetrances can be grouped, e.g. to three values per line (but don't have to). In the case of two loci, this yields the convenient matrix form:

$$\begin{array}{ccc} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{array}$$

Attention: a_1, a_{11}, ... are the coefficients for an individual who is homozygous for the *susceptibility* allele(s), not for allele `'1'`!! Please refer to section 2.4.2.

If the penetrances are independent of sex and age, their need to be specified only once. If there is a sex difference, each group sex contains a header and a the subsequent penetrance specification:

```
(MalePhenotype)
Penetrances                 =
  ...

(FemalePhenotype)
Penetrances                 =
  ...
```

If different ages have different probabilities of affection, the user has to specify two or more age groups with their starting age and their specific penetrances:

```
{AgeGroup}
LowerAgeLimit         = 0
Penetrances           =
 ...

{AgeGroup}
LowerAgeLimit         = 40
Penetrances           =
  ...
```

The first group has to start at zero. The last group states the penetrances for all individuals older than the starting age, even if they happen to be 150 years old.

If penetrances are sex and age specific, the age groups have to be specified within each sex group. Age groups can differ between the sexes, e.g.:

```
(MalePhenotype)

{AgeGroup}
LowerAgeLimit         = 0
Penetrances           =
  ...

{AgeGroup}
LowerAgeLimit         = 40
Penetrances           =
  ...
```

```
(FemalePhenotype)

{AgeGroup}
LowerAgeLimit         = 0
Penetrances           =
  ...

{AgeGroup}
LowerAgeLimit         = 50
Penetrances           =
  ...
```

PhenotypicValues & PhenotypicVariance The definition of quantitative traits (QT) is quite similar to the definition of an affection status. Instead of the penetrance values, here genotype-dependent phenotypic means and a homoscedastic standard deviation need to be specified (see section 2.4). They can assume any rational values. The order of specification is the same as described in the `Penetrances` statement paragraph. The non-negative standard deviation is the same for all genotypic states and must be specified, too. An example structure for sex-and-age specific QTs is as follows:

```
(MalePhenotype)

{AgeGroup}
LowerAgeLimit       = 0
PhenotypicMeans     =
   10.0    15.0    20.0
PhenotypicVariance  = 1.00000

{AgeGroup}
```

```
LowerAgeLimit       = 40
PhenotypicMeans     =
  110.0   150.0   200.0
PhenotypicVariance  = 2.00000


(FemalePhenotype)

{AgeGroup}
LowerAgeLimit       = 0
```

```
PhenotypicMeans     =
   30.0    40.0    50.0
PhenotypicVariance  = 3.00000

{AgeGroup}
```

```
LowerAgeLimit       = 50
PhenotypicMeans     =
  130.0   140.0   150.0
PhenotypicVariance  = 4.00000
  ...
```

BiallelicCheck If this option is set to `'y'`, then the SNPs in the block pre-set haplotypes are checked for having at least two alleles. If SNP positions in the haplotypes or whole haplotypes were not specified the program-generated haplotypes are forced to result in bi-allelic SNPs. Using the `'n'` option disables allele checking and allows the program-generated haplotypes to contain mono-allelic SNPs. The statement looks like this:

```
[Model]
...
BiallelicCheck             = 'y'
...
```

RemoveCausalSNPs If this option is set to `'y'`, the causative SNPs used for the affection status definition will be withheld from the sequence. Otherwise, they will remain in the data file. The statement looks like this:

```
[Model]
...
RemoveCausalSNPs           = 'n'
...
```

GenotypingError & GenotypingErrorVisible The program allows for the incorporation of genotyping errors and other sources of noise, e.g. missing marker information. If the error probability e is set to some value greater than zero using `GenotypingError`, every SNP has equal chance to be altered to another allele with this probability, thereby representing a misdetection. If `GenotypingErrorVisible` is set to `'y'`, altered SNPs will be marked as unknown and are thereby visible in the data file. This algorithm can be used to simulate missing marker information. Otherwise, the correct allele will simply be replaced by a wrong one. An example statement might look like this:

```
[Model]
...
GenotypingError            = 0.010
GenotypingErrorVisible     = 'y'
...
```

RandomSeed The seed parameter `RandomSeed` is used to control the pseudo-random number generator in `C`. It can assume integer values between 1 and some larger number, at least 32767. An example statement might look like this:

```
[Model]
...
RandomSeed                = 500
...
```

If you plan to run multiple replications, it is mandatory to use varying seeds for each run.

3.2.6 Separator characters

The separator group specifies the characters used to separate the different entities in the data file (see section 4.1). The group might look like this:

```
[Separators]
BehindStatus              = '\t'
BetweenHaplotypes         = ' '
BetweenBlocks             = '.'
BetweenSNPs               = ''
```

`BehindStatus` denotes the separator between the affection status and the genotypic information. `BetweenHaplotypes` separates the haplotypes of the genotype. Finally, `BetweenBlocks` and `BetweenSNPs` denote the separators between the SNP blocks in a haplotype and between SNPs within a block. For demonstration purposes, the latter two might be set to some value; for normal data generation they are usually set to the empty character `''`.

The separator group settings have no influence if `'Linkage'` is specified as the data output format in the general group.

Special characters are specified by using the backslash and the appropriate designator, similar to `C` [8] or `Perl` [14]. For example, tabulators are represented by `'\t'`, newlines by `'\n'`.

Attention: On some platforms, in particular on Windows, empty characters could cause problems such as segmentation faults. If this is the case, please specify a character for each separator and try it solves the problem.

3.2.7 Block parameters and haplotype pre-settings

The blocks group specifies all parameters for the blocks pre-settings, like the number of blocks and their sizes, the set of observed haplotypes and their frequencies, and the position of potential susceptibility loci. The group consists of several subgroups, one for each block. The whole group might look like this:

```
[Blocks]
NumberOfBlocks            = 2

(Block)
...

(Block)
...
```

`NumberOfBlocks` specifies the number of blocks, n, constituting the complete SNP sequence. The block statements do not have to be in the order of their occurrence at the sequence.

Block specification The block subgroups themselves contain parameters specifications for a particular block and one or two subordinated group(s) with specifications for the set of observed haplotypes in this block (pre-set haplotypes, $\{H^s_{i1}, \ldots, H^s_{im^s_i}\} \cup \{H^n_{i1}, \ldots, H^n_{im^n_i}\}$; see section 2.2). An example for a block subgroup containing a susceptibility locus might look like this:

```
(Block)
Number                    = 1
Size                      = 3
SuscLocusPosition         = 2
SuscAlleleFrequ           = 0.200

{NoSuscHaplotypes}
...

{SuscHaplotypes}
...
```

`Number` specifies the block number, i, within the blocks sequence (from left to right); it has to be in the range $1, \ldots,$ `NumberOfBlocks`. `Size` defines the number of SNPs in the block, b_i.

`SuscLocusPosition` specifies the position of the susceptibility locus within the block, l_i. The value must be in the range $1, \ldots,$ `Size`, or equal to 0 if a susceptibility locus is not present in the block. If a locus is present, then `SuscAlleleFrequ` specifies the frequency of the susceptibility allele, p^s_i.

Block haplotypes specification The specification of sets of block haplotypes consists of the type of the set, the number of haplotypes, their frequencies, and perhaps pre-settings of the SNP alleles in the haplotypes. An example might look like this:

```
{NoSuscHaplotypes}
HtNumber                  = 2
HtBlock                   = 111
HtBlock                   = 21
HtFrequ                   = 0.600
HtFrequ                   = 0.400
```

```
{SuscHaplotypes}
HtNumber                  = 3
HtBlock                   = 122
HtBlock                   = 22
HtBlock                   = 22
HtFrequ                   = 0.300
HtFrequ                   = 0.500
```

The type of the set is declared by the headers `SuscHaplotypes` or `NoSuscHaplotypes` depending on whether the haplotype set contains the susceptibility allele or not. The total of the observed haplotypes in both sets, m_i, equals the sum of both set sizes (m_i^s and m_i^n, respectively). Blocks that do not contain a causative SNP lack the `SuscHaplotypes` sub group.

Within a set, `HtNumber` specifies the number of pre-set haplotypes ($m_i^{\cdot}$). `HtBlock` statements can contain (partially) pre-set haplotypes ($\{H_{ij}^{\cdot}\}$). If SNP alleles in a haplotype or whole haplotypes are missing, they are randomly generated with equal chance for each allele when using `'n'` with the `BiallelicCheck` option. When specifying `'y'` instead, only those haplotypes will be considered which result in bi-allelic SNPs (see section 3.2.5). `HtFrequ` statements specify the haplotype frequencies *within* the set ($\{h_{ij}^{\cdot}\}$). Missing frequencies are calculated to equally share the difference between 1 and the sum of the already specified frequencies.

For both `HtBlock` and `HtFrequ` statements the order is important. Thus, the first `HtBlock` statement refers to the first haplotype and the first `HtFrequ` statement specifies its frequency; the second `HtFrequ` statement gives the frequency of the second (stated or generated) haplotype and so forth.

3.2.8 Limits for the parameter settings

Parameter specifications, like the size of a block or the number of blocks in the SNP sequence, are technically limited by settings within the compiled version, mostly for memory consumption reasons. Some of these limits can be changed (see section 5.2.2). Using the parameter option when calling the program (see section 3.1) lists the parameter limits of the compiled version. For an example for the parameter limits listing, see section 3.3.2.

3.3 Examples

3.3.1 Example for a job file

An example for a job-settings file is listed below. Some of the haplotypes are not or not completely specified and will be randomly generated or completed by the program. Frequencies are also missing and will be calculated, too. Notice that in the susceptibility block specification all haplotypes possess the same allele at position 1 within the set, in compliance with the susceptibility allele coding.

```
[General]
Datafile                = 'example5.dat'
SettingsLogfile         = 'example5.set'
OutputType              = 'haplotypes'
OutputFormat            = 'Linkage'

[Design]
Phenotype               = 'qualitative'
StudyDesign             = 'nuclear-families'
NumberOfFamilies        = 500
NumberOfChildren        =   6
TypeOfFamily            = 'random'

[SexAge]
SexRatio                =  1.07
AgeMinimum              =  5
AgeMaximum              = 20
AgeDistribution         = 'normal'
AgeMean                 = 12.0
AgeStd                  =  4.0
MotherAgeFirstChild     = 27.0
MotherAgeStd            =  3.0
ParentalAgeDiff         =  1.0
ParentalAgeDiffStd      =  3.0
MinimumChildAgeDist     =  1.0

[Model]
NumberOfLoci            = 1
NumberOfStates          = 3
BiallelicCheck          = 'y'
HideCausalSNPs          = 'n'
GenotypingErrorRate     = 0.05
GenotypingErrorVisible  = 'y'
RandomSeed              = 3006

ModelType               = 'sex-specific'
(MalePhenotype)
Penetrances             =
    0.90000     0.50000     0.10000
```

```
(FemalePhenotype)
Penetrances             =
    0.80000     0.40000     0.10000

[Blocks]
NumberOfBlocks          = 2

(Block)
Number                  = 1
NumberOfMarkers         = 5
SuscLocusPosition       = 0

{NoSuscHaplotypes}
HtNumber                = 3
HtBlock                 = 11222
HtBlock                 = 21121
HtBlock                 = 221
HtFrequ                 = 0.70000
HtFrequ                 = 0.10000

(Block)
Number                  = 2
NumberOfMarkers         = 2
SuscLocusPosition       = 1
SuscAlleleFrequ         = 0.40000

{NoSuscHaplotypes}
HtNumber                = 2
HtBlock                 = 11
HtBlock                 = 12
HtFrequ                 = 0.70000

{SuscHaplotypes}
HtNumber                = 2
HtBlock                 = 21
HtBlock                 = 22
HtFrequ                 = 0.50000
HtFrequ                 = 0.50000
```

3.3.2 Example for the parameter limits output

An example for the parameter limits and settings phrases output is shown below, produced by using the `-p` parameter option when calling the program. To modify these constrains and codings, see section 5.2.2.

```
-- Limits for job description file --
Maximum line length                          : 200
Maximum number of lines in job description file : 5000
Maximum number of lines in parameter group      : 4000
Maximum file name length                     : 100

-- Limits for SNP, block, haplotype, and family specifications --
```

```
Maximum number of SNP alleles                               : 2
Maximum number of SNPs per block                            : 50
Maximum number of haplotypes per block                      : 50
Maximum number of blocks per snp sequence                   : 20
Number of allele combinations per locus                     : 3
Maximum number of susceptibility loci per person            : 6
Maximum number of allele combinations at susc. loci         : 729
Maximum number of sexes                                     : 2
Maximum number of age groups for penetrances/phenotypic means : 10
Maximum number of children per family                       : 10
Minimum age at first motherhood                             : 12
Maximum age at last motherhood                              : 45

-- Fixed settings --
Coding of susceptibility allele                  : 2
Coding of non-susceptibility allele              : 1
Coding of genotyping errors                      : 0
Coding of homozygous for susceptibility allele   : 3
Coding of homozygous for non-susceptibility allele : 1
Coding of heterozygous SNP                       : 2
Coding of affected persons                       : '2'
Coding of non-affected persons                   : '1'
Coding of unknown affection status               : '0'

-- Data output type phrases --
Name for haplotypes output type : 'haplotypes'
Name for genotype output type   : 'genotypes'

-- Haplotype output format phrases --
Name for output format 1       : 'Standard1'
Name for output format 2       : 'Standard2'
Name for output format linkage : 'Linkage'

-- Genotype output format phrases --
Name for output format compact : 'Compact'
Name for output format linkage : 'Linkage'

-- Study design phrases --
Name for individual design       : 'individuals'
                                   'cc'
                                   'case-control'
Name for nuclear-families design : 'nuclear-families'
                                   'family'
                                   'TDT'

-- Sampling design phrases --
Name for separate sampling of cases and controls : 'separate'
                                                   'case-control'
                                                   'cc'
Name for joint sampling of cases and controls    : 'joint'
                                                   'general'

-- Model phrases --
Name for sex and age independent penetrances/phenotypic means : 'homogeneous'
```

```
                                                                 'global'
                                                                 'independent'
Name for sex specific penetrances/phenotypic means             : 'sex-specific'
                                                                 'sex'
Name for age specific penetrances/phenotypic means             : 'age-specific'
                                                                 'age'
Name for sex and age specific penetrances/phenotypic means     : 'sex-age-specific'
                                                                 'sex-age'
                                                                 'sexage'

-- Age & sex phrases --
Male sex                           : '1'
Female sex                         : '2'
Unknown sex                        : '0'
Uniform age distribution           : 'uniform'
Normal (gaussian) age distribution : 'normal'
                                     'gauss'

-- Phenotypic expression type phrases --
Name for qualitative phenotypic expression  : 'qualitative'
                                              'case-control'
                                              'cc'
                                              'dichotomous'
Name for quantitative phenotypic expression : 'quantitative'
                                              'QT'
                                              'continuous'
Name for no phenotypic expression           : 'none'

-- Family size phrases --
Name for families with a fixed number of children     : 'fixed'
Name for families with randomly 1..number of children : 'random'
```

4 Generated output files

If the parameters were properly specified, the program will produce two files with names as specified in the job file: a data file containing the generated data and a job file containing the parameter settings actually used for the generation.

4.1 The data file

The data file contains the affection status and the genotype/haplotypes of the generated individuals. Depending on the format specified (see section 3.2.2), the data output can take several forms:

- Haplotypes:
  - Standard 1 format
  - Standard 2 format
  - Linkage format
- Genotypes
  - Compact format
  - Linkage format

Haplotypic output

Standard 1 format The Standard 1 format arranges the haplotypes of a genotype below each other in succeeding lines. The family coding, sex and age, and the status/QT values of the person appears at the beginning of the first line:

```
FamID PID FID MID Sex Age Status/QT Haplotype1
                                    Haplotype2
FamID PID FID MID Sex Age Status/QT Haplotype1
                                    Haplotype2
  :    :   :   :   :   :     :          :
```

An sample of unrelated individuals in this data format would look similar to this:

```
1 1 0 0 2 41 1  211 11222 11
                111 11220 21
2 1 0 0 1 56 1  211 11222 11
                111 21121 11
: : : : :  : :        :
```

In this example, two haplotypes with three blocks were generated (see sections 3.2.2 and 3.2.7). The haplotype and the affection status are separated by a tabulator, the blocks by a space, while the SNP alleles within the block are simply pasted together, using no separator (see section 3.2.6). Also note the "`0`" character in the sequence. This marks a SNP whose allele is unknown, e.g. due to a genotyping error. The error was made "visible" setting the `TypingErrorVisible` option to `'y'`. Specifying `'n'` would result in an undetected wrong SNP allele (see section 3.2.5).

Standard 2 format The standard 2 format prints the haplotypes of a genotype on a single line, separated by a user-specified separator. The status of the person appears at the beginning of the line:

```
FamID PID FID MID Sex Age Status/QT Haplotype1 Haplotype2
FamID PID FID MID Sex Age Status/QT Haplotype1 Haplotype2
  :    :   :   :   :   :      :          :          :
```

The example described above now looks like this:

```
1 1 0 0 2 41 1  211 11222 11   111 11220 21
2 1 0 0 1 56 1  211 11222 11   111 21121 11
: : : : :  : :        :              :
```

Linkage format The linkage specification arranges the haplotypes in accordance to the linkage programs file format [13]:

```
FamID PID FID MID Sex Age Status/QT SNP1_A1 SNP1_A2 SNP2_A1 SNP2_A2 ...
  :    :   :   :   :   :      :        :       :       :       :
```

`FamID` is a successive number, starting with 1 and stopping with the total of generated individuals or families or the sum of the totals of affected and unaffected individuals. `PID` denotes the individual's ID within the family; for unrelated-individuals, all individuals will have PID 1. `FID` and `MID` denote the PIDs of the father and the mother of the individual and equal zero if there are no parents in the sample.

The SNP alleles of the two haplotypes are "merged": First, the allele from the first haplotype of the first SNP is printed, then the allele from the other haplotype of the first SNP, then the alleles of the second SNP and so forth. In this way, the underlying haplotypic phase of the genotypes can be recovered.

The example described above now looks like this:

```
1 1 0 0 2 41 1  2 1 1 1 1 1 1 1 1 1 2 2 2 2 2 0 1 2 1 1
2 1 0 0 1 56 1  2 1 1 1 1 1 1 2 1 1 2 1 2 2 2 1 1 1 1 1
```

Genotypic output

The genotype format puts the genotype behind the status column of a person:

```
FamID PID FID MID Sex Age Status/QT Genotype
  :    :   :   :   :   :      :         :
```

Compact format In `Compact` format, each SNP locus is coded as being homozygous for allele 1 (`'1'`), being heterozygous (`'2'`), or being homozygous for allele 2 (`'3'`).

The example from above would now look like this:

```
1 1 0 0 2 41 1  211 11330 21

2 1 0 0 1 56 1  211 21232 11

: : : : :  : :          :
```

Linkage format This is output is similar to the Linkage format for haplotypes with one exception: the alleles for each SNP locus are now ordered. In this way, the haplotypic structure is distorted and cannot simply be inferred from the columns in the data file.

4.2 Settings file

If all parameters were completely specified, the settings file produced by the program will merely be a copy of the user's job file. However, if the specification was incomplete and, thus, the program has altered or generated parameters (e.g. block haplotypes pre-sets), this settings file lists the parameter settings that were actually used for the data generation. The program will warn the user if parameters were altered.

5 Technical notes

5.1 Source code files and compilation

SNaP is written in ANSI-C [8] and should therefore be portable to potentially every computer platform. Table 1 lists the source code files and their contents. To compile the program under UNIX/LINUX, use the `make` command together with the make file `snap.m`:

```
make -f snap.m
```

To change the `C` compiler to be used or the name of the executable file, modify the `COMPILER` and the `PROGRAM` statement in the make file. If the `make` command is not available on your machine, you might want to try `gcc *.c -o snap -lm` instead.

5.2 Adaptations of the source code

5.2.1 Language and parameter names adaptation

To adapt the program to other languages or to change the designators for the parameter settings (perhaps because you find them completely silly in their existing form), only the phrases in the following header files need to be substituted:

- The file `snpmsg.h` contains all message phrases for error, status, and warning messages.
- The files `snpsetc.h` and `snpconst.h` contain all phrases of parameter designators, group names and others phrases for the parameter read-in, check, and write-out.

After re-compilation a program with modified phrases and parameter designators is available.

5.2.2 Adaptations affecting memory usage

If you experience memory problems when running SNaP or if you want to enable more block, larger block, and higher haplotype numbers per block, you will need to change the constants that cause these constrains. These constants are bundled in the file `snpconst.h`. This file also contains the codings for affection status, susceptibility allele, and the character for genotyping errors/unknown genotypes.

Use the parameter option when calling SNaP (see section 3.1) to get a list of the limits, phrases, and constants of the compiled version.

[9]These files were kindly provided by Robert Fürst, Berlin, Germany.

File	Contents
`snap.c`	Main procedure
`snap.m`	Make file for compilation
`set.c/h`	General functions for parameter settings
`snpconst.h`	Constants of program-wide scope (memory-affecting) and some designator phrases
`snpdata.c/h`	Functions for data generation
`snpmsg.c/h`	Messages & phrases (contains message phrases)
`snpout.c/h`	Data output functions
`snpparam.c/h`	Definition and initialization of parameter data structures
`snpset.c/h`	SNaP-specific parameter-set functions (using `set.c/h`)
`snpsetc.h`	Constants for parameter setting (contains parameter phrases)
`snpstruc.c/h`	Definition and initialization of generated data structures
`snpexmpl.c/h`	Functions for generating example settings files
`mymath.c/h`[9]	A few mathematical functions
`rng.c/h`[9]	Gaussian random number generator
`fundamental.h`[9]	Fundamental functions for `rng`
`bool.h`[9]	Definition file for boolean logic
`message.c/h`[9]	Message functions for `rng`

Table 1: SNaP source code files and their contents.

References

[1] G. R. Abecasis, E. Noguchi, A. Heinzmann, J. A. Traherne, S. Bhattacharyya, N. I. Leaves, G. G. Anderson, Y. Zhang, N. J. Lench, A. Carey, L. R. Cardon, M. F. Moffatt, and W. O. Cookson. Extent and distribution of linkage disequilibrium in three genomic regions. *Am J Hum Genet*, 68(1):191–197, Jan 2001. ISSN 0002-9297.

[2] S. Bolk, J. Higgins, J. Moore, H. Nguyen, J. Roy, S. Schaffner, E. Lander, M. Daly, and D. Altshuler. The extent and diversity of common human haplotypes. *Am J Hum Genet*, 69 (Suppl.)(4):176, Oct 2001.

[3] M. Daly, J. Rioux, S. Schaffner, T. Hudson, and E. Lander. Fine-structure haplotype map of 5q31: implications for gene-based association studies and genomic ld mapping. *Am J Hum Genet*, 69 (Suppl.)(4):181, Oct 2001.

[4] M. J. Daly, J. D. Rioux, S. F. Schaffner, T. J. Hudson, and E. S. Lander. High-resolution haplotype structure in the human genome. *Nat Genet*, 29(2):229–232, Oct 2001.

[5] E. Dawson, G. R. Abecasis, S. Bumpstead, Y. Chen, S. Hunt, D. M. Beare, J. Pabial, T. Dibling, E. Tinsley, S. Kirby, D. Carter, M. Papaspyridonos, S. Livingstone, R. Ganske, E. Lohmussaar, J. Zernant, N. Tonisson, M. Remm, R. Magi, T. Puurand, J. Vilo, A. Kurg, K. Rice, P. Deloukas, R. Mott, A. Metspalu, D. R. Bentley, L. R. Cardon, and I. Dunham. A first-generation linkage disequilibrium map of human chromosome 22. *Nature*, 418(6897):544–8, Aug 1 2002. ISSN 0028-0836.

[6] S. B. Gabriel, S. F. Schaffner, H. Nguyen, J. M. Moore, J. Roy, B. Blumenstiel, J. Higgins, M. DeFelice, A. Lochner, M. Faggart, S. N. Liu-Cordero, C. Rotimi,

A. Adeyemo, R. Cooper, R. Ward, E. S. Lander, M. J. Daly, and D. Altshuler. The structure of haplotype blocks in the human genome. *Science*, 296(5576):2225–9, Jun 21 2002. ISSN 1095-9203.

[7] D. Gordon, S. J. Finch, M. Nothnagel, and J. Ott. Power and sample size calculations for case-control genetic association tests when errors are present: application to single nucleotide polymorphisms. *Hum Hered*, 54(1):22–33, 2002. ISSN 0001-5652.

[8] B. W. Kernighan and D. M. Ritchie. *The C Programming Language.* Prentice Hall Software Series. Prentice Hall P T R, Upper Saddle River, second edition, 1988.

[9] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C — The Art of Scientific Computing.* Cambridge University Press, Cambridge, second edition, 1997.

[10] D. Reich, M. Cargill, S. Bolk, J. Ireland, P. Sabeti, D. Richter, T. Lavery, R. Kouyoumjian, S. Farhadian, R. Ward, and E. Lander. Linkage disequilibrium in the human genome. *Nature*, 411(6834):199–204, May 10 2001.

[11] R. Spielman, R. McGinnis, and W. Ewens. Transmission test for linkage disequilibrium: the insulin gene region and insulin-dependent diabetes mellitus (IDDM). *Am J Hum Genet*, 52(3):506–16, Mar 1993.

[12] L. Subrahmanyan, M. Eberle, A. Clark, L. Kruglyak, and D. Nickerson. Sequence variation and linkage disequilibrium in the human t-cell receptor beta (tcrb) locus. *Am J Hum Genet*, 69(2):381–95, Aug 2001.

[13] J. D. Terwilliger and J. Ott. *Handbook of Human Genetic Linkage.* John Hopkins University Press, Baltimore & London, 1994.

[14] L. Wall, T. Christiansen, and R. Schwartz. *Programming Perl.* O'Reilly, Cambridge, second edition, 1996.