GeneEvolve Documentation

Rasool Tahmasbi
Institute for Behavioral Genetics,
University of Colorado, Boulder,
USA
Rasool.Tahmasbi@Colorado.edu

Matthew C. Keller
Institute for Behavioral Genetics,
University of Colorado, Boulder,
USA
matthew.c.keller@colorado.edu

version 1.0.0
March 2016

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 What is *GeneEvolve*?

GeneEvolve is C++ code for simulating sequence-level genetic data over large genomic regions in large populations, using an object-oriented approach. This allows compiling *GeneEvolve* on any computer platform, which supports standard C++ compiler.

Computer simulations are excellent tools for understanding the evolutionary and genetic consequences of complex processes whose interactions cannot be analytically derived. Unlike coalescent [1] based simulators, *GeneEvolve* runs forward-in-time, which allows it to provide a wide range of scenarios for selection, population size and structure, migration, recombination and familial effects.

GeneEvolve is fast and memory efficient simulator which can handle complex life events. User-friendly and easy to work are among its other advantages.

1.2 How to download *GeneEvolve*?

GeneEvolve can be download from https://github.com/rtahmasbi/GeneEvolve. The source codes, examples and this documentation are available freely for downloading.

You also can run the following commands for downloading, compiling and testing *GeneEvolve*:

```
wget https://github.com/rtahmasbi/GeneEvolve/archive/master.zip
unzip master
cd GeneEvolve-master/
unzip GeneEvolve.zip
cd GeneEvolve
make
bin/GeneEvolve --help
```

1.3 How to compile *GeneEvolve*?

GeneEvolve can be run on different platforms. After downloading it, you need to uncompress the zipped file `GeneEvolve.zip` and then go to the root directory and type `make`. After successful compiling, the `GeneEvolve` simulator will be in the `bin` subdirectory.

```
GeneEvolve --file_gen_info [file] \
--file_hap_name [file] \
--file_recom_map [file] \
--file_cv_info [file] \
--file_cvs [file]
```

Figure 1.1: *GeneEvolve* command line (basic parameters)

You also need to load a standard C++ compiler, by the command `module load gcc/gcc-4.9.2`.

1.4 How *GeneEvolve* works?

GeneEvolve is a stand-alone program. Our aim was to make it simple and user friendly for different levels of user's knowledge. Users can create complex life events by adding and combining more parameters. The minimum required parameters are *generation information* (such as population size, spousal correlation for mating, offspring distribution and selection function per generation), *haplotype information* (which is haplotypes for starting generation) *recombination map*, *cv list* (position of CVs and their additive and dominance effects) and the actual *CV's haplotype*. These parameters are listed in Figure 1.1. The detailed explanation for each parameter is illustrated in the following chapters.

For different levels of variance of additive and dominance effects, user can use the following parameters

`--va [number] --vd [number]`

and for the unique, familial, shared sibling (common), and population specific environmental effects, user can also specify the following parameters, respectively:

`--ve [number] --vf [number] --vc [number] --gamma [number]`

There are more parameters for random mating, selection function, migration and so on, available in this documentation file.

1.5 Quick start

For an explanatory example, we will simulate a set of genotype and then will run *GeneEvolve* in the following `R` script. This script will also create all the required parameters. Clearly the structure of real genotypes are more complex than this example, but this simple example is useful for educational purpose. This file can also be downloaded from the *GeneEvolve* webpage.

```
############################################################
# create genotypes in hap, legend, indv format
# creating CVs
############################################################

NCHR <- 3
NSNP <- rep(1000,NCHR) #number SNPs per chromosome
NCV <- rep(100,NCHR) #number CVs per chromosome
NIND <- 1000
```

```
VAR.A <- 1
VAR.D <- .1

# We create this map in order to use the real genomic map distance
map.pos=c(249238555, 242900000, 197900000, 199150000, 180750000, 170955878,
    159167276, 146364984, 141088894, 135526070, 135002856, 133841619, 115148564,
    105826742,  102484095, 90201263, 81100000, 78062535, 59160912, 6.3e+07,
    48157860, 51246300)



NCHR <- length(NSNP)
cv.info=c()

for (ichr in 1:NCHR)
{
    print("-----------------------------------------------")
    nsnp_chr <- NSNP[ichr]
    p <- runif(nsnp_chr,min=.05,max=.95)
    hap <- matrix(0, nrow=NIND, ncol=2*nsnp_chr)
    for (isnp in 1:nsnp_chr)
    {
        hap[,2*isnp-1] <- rbinom(NIND,1,p[isnp])
        hap[,2*isnp] <- rbinom(NIND,1,p[isnp])
    }
    # creating ref.hap file
    write.table(hap, file=paste("ref.chr",ichr,".hap",sep=''), quote=FALSE,row.
    names=FALSE, col.names=FALSE)
    # creating ref.legend file
    legend.id <- paste("rs",1:nsnp_chr,sep='')
    legend.pos <- sort(sample(map.pos[ichr], nsnp_chr, replace = FALSE))
    legend.al0 <- rep("C", nsnp_chr)
    legend.al1 <- rep("T", nsnp_chr)
    write.table(cbind(legend.id,legend.pos,legend.al0,legend.al1), file=paste("
    ref.chr",ichr,".legend",sep=''), quote=FALSE,row.names=FALSE, col.names=FALSE
    )
    # creating ref.indv file
    write.table(1:NIND, file=paste("ref.chr",ichr,".indv",sep=''), quote=FALSE,
    row.names=FALSE, col.names=FALSE)
    ####
    # creating cv.hap file
    ncv_chr <- NCV[ichr]
    cv_chr <- sort(sample(1:nsnp_chr, ncv_chr, replace = FALSE))
    cv_hap_index <- sort(c(2*cv_chr-1,2*cv_chr))
    cvs <- hap[,cv_hap_index]
    write.table(cvs, file=paste("cv.chr",ichr,".hap",sep=''), quote=FALSE,row.
    names=FALSE, col.names=FALSE)
    cv.pos <- legend.pos[cv_chr]
    cv.maf <- apply(matrix(c(cvs),nrow=2*NIND),2,mean)
    cv.maf[which(cv.maf>.5)] <- 1-cv.maf[which(cv.maf>.5)]
    cv.a <- rnorm(ncv_chr,mean=0,sd=sqrt(VAR.A))
    cv.d <- rnorm(ncv_chr,mean=0,sd=sqrt(VAR.D))
    cv.info.chr <- cbind(ichr,cv.pos,cv.maf,cv.a,cv.d)
    cv.info <- rbind(cv.info,cv.info.chr)
```

```
}

colnames(cv.info)=c("chr","pos","maf","a","d")
write.table(cv.info, file=paste("cv.info",sep=''), quote=FALSE,row.names=FALSE, col.names=TRUE)


########### --file_cvs
b=paste("cv.chr",INCLUDE.CHRS,".hap",sep='')
b=cbind(INCLUDE.CHRS,b)
write.table(b,file=paste("par.pop1.cv_hap_files.txt",sep=''),quote=FALSE,row.names=FALSE,col.names=FALSE)


########### --file_hap_name
a1=paste("ref.chr",INCLUDE.CHRS,".hap",sep="")
a2=paste("ref.chr",INCLUDE.CHRS,".legend",sep="")
a3=paste("ref.chr",INCLUDE.CHRS,".indv",sep="")
a=cbind(INCLUDE.CHRS,a1,a2,a3)
write.table(a,file=paste("par.pop1.hap_sample_address.txt",sep=''),quote=FALSE,row.names=FALSE,col.names=c("chr","hap","legend","sample"))



# creating population info

#############################################################
#DEFINE WILDCARDS
#Must run this section. These need to be changed as you see fit
#############################################################

NUM.GENERATIONS <- 20     #number of generations
POP.SIZE <- rep(3000,NUM.GENERATIONS)  #population size over time
PHENO.MATE.COR <- rep(.5,NUM.GENERATIONS)   #phenotypic correlation between mates at each generation
OFFSPRING_DIST <- rep("p",NUM.GENERATIONS) # p or P=Poisson distribution, f or F=fixed distribution
OFFSPRING_DIST[NUM.GENERATIONS] <- "f" # last generation is fixed
SELECTION.FUNCTION <- rep("logit",NUM.GENERATIONS) #Selection function for each generation
SELECTION.FUNCTION.PAR1 <- rep(20,NUM.GENERATIONS) #Selection function for each generation
SELECTION.FUNCTION.PAR2 <- rep(0,NUM.GENERATIONS) #Selection function for each generation

#How fine grained should the recombination map be? This has an important effect on speed & memory. The lower this number (in kb) the more RAM and longer the program takes. Numbers < 10 are too fine-grained to make any difference. Typical choices are between 10-50.

INCLUDE.CHRS=1:NCHR
```

```
99
100 ##########popinfo.txt
101 write.table(cbind(POP.SIZE,PHENO.MATE.COR,OFFSPRING_DIST,SELECTION.FUNCTION,
    SELECTION.FUNCTION.PAR1,SELECTION.FUNCTION.PAR2), file=paste("par.pop1.info.
    txt",sep=''),quote=FALSE,row.names=FALSE, col.names=c("pop_size","mat_cor","
    offspring_dist","selection_func","selection_func_par1","selection_func_par2")
    )
```

An explanatory example

Then run *GeneEvolve* by the following command (you should change the `path` to the appropriate directory)

```
103 /path/bin/GeneEvolve \
104 --file_gen_info par.pop1.info.txt \
105 --file_hap_name par.pop1.hap_sample_address.txt \
106 --file_recom_map Recom.Map.b37.50KbDiff \
107 --file_cv_info cv.info \
108 --file_cvs par.pop1.cv_hap_files.txt \
109 --prefix out1
```

An explanatory example

In order to scale the variance of additive and unique environmental effects, run the following command

```
1 /path/bin/GeneEvolve \
2 --file_gen_info par.pop1.info.txt \
3 --file_hap_name par.pop1.hap_sample_address.txt \
4 --file_recom_map Recom.Map.b37.50KbDiff \
5 --file_cv_info cv.info \
6 --file_cvs par.pop1.cv_hap_files.txt \
7 --va 1 \
8 --ve 1 \
9 --prefix out2
```

Scaling the variance of additive and unique environmental effects

Chapter 2

Population Genetics Models

2.1 Population's basic information

2.1.1 The Wright–Fisher model

The total population size is more often determined by external factors like availability of food or living space, or the action of predators, than by summing independent family sizes (the branching process models). His first approximation to reality is therefore a model in which the total population size is a fixed number dictated by external constraints, and the most popular is that associated with the names of Sewall Wright and R. A. Fisher (for more info see [2]).

This assumes discrete, non-overlapping generations $G_0, G_1, G_2, \ldots$ in which each generation contains a fixed number N of individuals. Each member of G_{i+1} is the child of exactly one member of G_i, but the number of children born to the jth member of G_i is a random variable v_i subject of course to the constraint

$$\sum_{j=1}^{N} v_i = N.$$

2.1.2 Population size in *GeneEvolve*

For inputting the basic information of population, user should use the parameter

```
--file_gen_info [file_generaions_info.txt]
```

where `file_generaions_info.txt` is a text file with 6 columns and $n+1$ lines, where n is the number of generations to be simulated by *GeneEvolve*. This file should have a header and the first column is the population size. The structure of this file is listed in figure 2.1. *GeneEvolve* can simulate different population sizes in each generation, by specifing some numbers in the first column.

2.1.3 Number of offspring's distribution

The third column of file `file_generaions_info.txt` determines the distribution of offsprings per generation. It can be `p` for Poisson or `f` for fixed number of offsprings (see the third column of figure 2.1).

The mean of Poisson distribution in each generation is obtained by

$$\frac{N_i}{C_i},$$

```
pop_size mat_cor offspring_dist selection_func selection_func_par1
   selection_func_par2
3000 0 p logit 20 0
3000 0 p logit 20 0
3000 0 p logit 20 0
3000 0 p logit 20 0
3000 0 p logit 20 0
3000 0 p logit 20 0
3000 0 p logit 20 0
3000 0 p logit 20 0
3000 0 p logit 20 0
3000 0 p logit 20 0
3000 0 p logit 20 0
3000 0 f logit 20 0
```

Figure 2.1: The structure of file_generaions_info.txt file.

where N_i is the user specified population size (first column of `file_generaions_info.txt`) and C_i the number of couples at generation i.

For the fixed number of offsprings, we have

$$k = \text{round}(\frac{N_i}{C_i}).$$

The selection function has a direct effect on C_i. If the selection function allows all individuals to marry, then C_i should be $N_i/2$ in average. So each family should have 2 offsprings in average. More stringent selection function can reduce the C_i and as the results, it increases the average family size, N_i/C_i. In Section 2.6, you can find more information about the selection function.

2.2 Haplotypes of initial population

GeneEvolve works with the haplotypes. For more information about haplotype file format, see the appendix B. Since the size of genome can be large, the haplotype of each chromosome should be in a separate file. For the initial population (generation 0), the user should specify a file containing the *address* of each chromosome. With the parameter `--file_hap_name [file.txt]`, user should address the `.hap`, `.legend`, and `.indv` files, respectively, for each chromosome per line. Figure 2.2 shows a typical example with 22 chromosomes. This file has header and the first column is chromosome number. Note that the file `.indv` has no header.

It is also possible for *GeneEvolve* to work just with few chromosome. See the following example, where the user simulate just with chromosomes 6, 10 and 22.

Example 1 (Working with few chromosomes)**.** If user wants to simulate a population with 3 chromosome (6, 10 and 22), he/she should prepare the following file for the parameter `--file_hap_name [file.txt]`.

```
chr hap legend indv
6 /path/chr6.hap /path/chr6.legend /path/chr6.indv
10 /path/chr10.hap /path/chr10.legend /path/chr10.indv
22 /path/chr22.hap /path/chr22.legend /path/chr22.indv
```

file.txt

```
chr hap legend indv
1 /path/chr1.hap /path/chr1.legend /path/chr1.indv
2 /path/chr2.hap /path/chr2.legend /path/chr2.indv
3 /path/chr3.hap /path/chr3.legend /path/chr3.indv
4 /path/chr4.hap /path/chr4.legend /path/chr4.indv
5 /path/chr5.hap /path/chr5.legend /path/chr5.indv
6 /path/chr6.hap /path/chr6.legend /path/chr6.indv
7 /path/chr7.hap /path/chr7.legend /path/chr7.indv
8 /path/chr8.hap /path/chr8.legend /path/chr8.indv
9 /path/chr9.hap /path/chr9.legend /path/chr9.indv
10 /path/chr10.hap /path/chr10.legend /path/chr10.indv
11 /path/chr11.hap /path/chr11.legend /path/chr11.indv
12 /path/chr12.hap /path/chr12.legend /path/chr12.indv
13 /path/chr13.hap /path/chr13.legend /path/chr13.indv
14 /path/chr14.hap /path/chr14.legend /path/chr14.indv
15 /path/chr15.hap /path/chr15.legend /path/chr15.indv
16 /path/chr16.hap /path/chr16.legend /path/chr16.indv
17 /path/chr17.hap /path/chr17.legend /path/chr17.indv
18 /path/chr18.hap /path/chr18.legend /path/chr18.indv
19 /path/chr19.hap /path/chr19.legend /path/chr19.indv
20 /path/chr20.hap /path/chr20.legend /path/chr20.indv
21 /path/chr21.hap /path/chr21.legend /path/chr21.indv
22 /path/chr22.hap /path/chr22.legend /path/chr22.indv
```

Figure 2.2: The structure of [file.txt] in [–file_hap_name].

2.3 Recombination and linkage

2.3.1 Introduction

Genetic *recombination*, also called *crossing over*, refers to genetic events that can occur during the formation of sperm and egg cells. During the early stages of cell division in meiosis, two chromosomes of a homologous pair may exchange segments, producing genetic variations in germ cells. For example, if one homologous chromosome has a *haplotype* (genetic sequence on the same chromosome) AB, and another homologous chromosome has a haplotype ab, one of the gamete cells, because of recombination, may have a chromosome with genotype Ab. Such gametes are called recombinants. The proportion of recombinants is called the *recombination rate* between these two loci, which is 1/2 if two loci are on two different chromosomes, and thus segregate independently.

The *genetic distance* (also called *map distance*) between two loci is defined as the average number of crossovers between the loci per meiosis. The unit of genetic distance is the *centiMorgan* (cM). Two loci are 1 cM apart if on average there is one crossover occurring between these two loci on a single strand for every 100 meiosis. The distribution of recombination events varies between the sexes: females have on average 1.65-fold more recombination events when they make eggs than males do when they make sperms. Even on a single chromosome, recombination rate is uneven, and there exists recombination hotspots with peak recombination rate hundreds or thousands times that of the surrounding regions.

Given a reference panel of haplotypes $H_N = \{h_1, \ldots, h_{2N}\}$ as input, where each haplotype is typed at L bi-allelic sites, that is $h_i = (h_{i,1}, \ldots, h_{i,L})$ and $h_{i,j} \in \{0, 1\}$, the program chooses mates based on the AM or RM, and for the newly simulated child, each haplotype is a recombination of parent's haplotypes, where

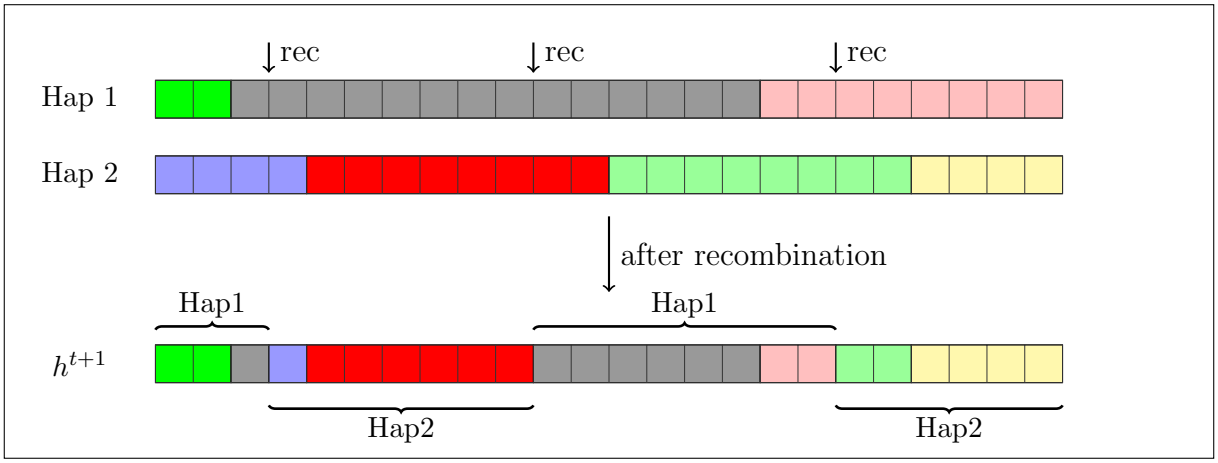


Figure 2.3: Recombination model: three recombination occurred at different positions. Each color code is an ancestral IBD.

the recombination rate is inputed by the user. Precisely, assume that at generation t, the haplotypes of one parent are h_i^t and h_{i+1}^t. These haplotypes are stored as a continues sequence of half open intervals, i.e., $h_i^t = \{[1, r_1) \cup [r_1, r_2) \cdots \cup [r_{v-1}, L]\}$ and $h_{i+1}^t = \{[1, s_1) \cup [s_1, s_2) \cdots \cup [s_{u-1}, L]\}$, for any arbitrary numbers u and v. Now assume that at a genetic phase, for a new child a recombination occurs at position w. If $r2 \leq w < r3$ and $s3 \leq w < s4$, then the new recombined haplotype becomes

$$h_i^{t+1} = \{[1, r_1) \cup [r_1, r_2) \cup [r_2, w) \cup [w, s_4) \cup [s_4, s_5) \cdots \cup [s_{u-1}, L]\}.$$

It is also possible that several recombinations occurs; the idea is the same. For a graphical illustration see the figure 2.3.

Clearly, working with a continues sequence of intervals is much faster than working with real genotypes. Saving them in computer memory is more efficient, too.

Based on the user-specified mutation rate, the position of new mutations will also store for the new generated haplotypes.

2.3.2 Recombination in *GeneEvolve*

different positions of genome have different recombination probabilities. A high-resolution recombination map of the human genome is estimated by Kong et. al. [3] and is available online.

To locate the recombination file, you can use the parameter `--file_recom_map [file_recom.txt]`. The file `file_recom.txt` has a header with 3 columns: chromosome number, base-pair distance and cM distance.

Example 2 (Recombination map file format)**.** In the following listing file, you can see part of a equidistant genetic map (with 50k length). By definition, the probability of recombination in a chunk is the difference between its two cM distances divided by 100. These probabilities will be computed automatically by *GeneEvolve* program.

```
chr  bp     cM
1  1128555  1.13368814337268
1  1138555  1.14905703198189
1  1148555  1.15742981154837
1  1158555  1.16581577645964
```

```
1 1168555 1.17462263654929
1 1178555 1.18341927550241
1 1188555 1.19221591445554
1 1198555 1.20104594872684
1 1208555 1.20989254327934
1 1218555 1.21873913783184
```

file_recom.txt

For example, the probability occurrence of one recombination in the interval $[1148555, 1158555)$ is equal to

$$\frac{1.16581577645964 - 1.15742981154837}{100} = 0.00008385964911.$$

2.4 Simulating complex quantitative traits

Computer programs that can simulate genotypes with phenotypes based on user-specified disease or quantitative trait models are useful in genetic studies. They can be used to evaluate statistical power when planning a study design based on the proposed sample size, the assumed genotypic relative risks (GRR), and allele frequencies. They are also useful for evaluating type I error rates for new statistical association tests and power comparisons between the new tests and other existing tests. *GeneEvolve* can simulate several phenotypes, simultaneously, based on GWAS panels (e.g., Illumina and Affymetrix) or sequence data (UK10K, 1000 genome, and etc.).

For simplicity, we assume there is one phenotype. For several phenotypes, the idea is the same. For each individual i, the phenotypic value P_i is a random variable defined as

$$P_i = A_i + D_i + F_i + E_i + C_f + \Gamma_p \tag{2.1}$$

where A_i and D_i are additive and dominance genetic terms. The terms F_i, E_i, C_f, and Γ_p are familial, unique, shared sibling (common), and population specific environmental effects, respectively.

For familial, shared sibling (common), and population specific environmental effects, see Sections 2.7, 2.8 and 2.9, respectively.

To simulate a phenotypes, user can specify m casual variants, $\{\text{cv}_j\}_{j=1}^{m}$, and their additive (a_j) and dominance (d_j) effects. The value a_j is the deviation of the homozygote genotype value from the midpoint of the two homozygous genotype values and d_j is the deviation of the heterozygote genotype value from this midpoint.

For each individual i, the additive term A_i is a linear function of some *casual variables* (CVs) multiplied by their additive effect size (α_j), i.e.,

$$A_i = \sum_{j=1}^{m} \frac{(x_{ij} - 2p_j)}{\sqrt{2p_j q_j}} \alpha_j, \tag{2.2}$$

where

$$\alpha_j = a_j + d_j(q_j - p_j),$$

is the additive effect size, p_j is frequency of one allele with $q_j = 1 - p_j$ the frequency of the other, $x_{ij} \in \{0, 1, 2\}$ is the genotypic value for cv_j, and m is the number of CVs. See Figure 2.4.

For each individual i, the dominance term D_i is,

$$D_i = \sum_{j=1}^{m} t_{ij}\delta_j, \tag{2.3}$$

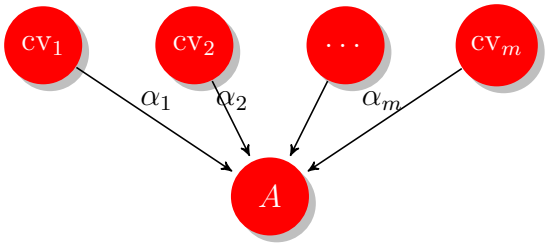


Figure 2.4: Additive term

where

$$\delta_j = \frac{d_j}{2p_jq_j},$$

is the dominance effect size, and

$$t_{ij} = \begin{cases} -2p_j^2 & \text{if } x_{ij} = 0 \\ 2p_jq_j & \text{if } x_{ij} = 1 \\ -2q_j^2 & \text{if } x_{ij} = 2 \end{cases}.$$

For more information, see Reference [4].

Assuming no LD between CVs, for the additive term we have

$$\begin{aligned} \text{var}[A_i] =& \sum_{j=1}^{m} \alpha_j^2 \text{var}[x_{ij} - 2p_j] \\ =& \sum_{j=1}^{m} [a_j + d_j(q_j - p_j)]^2, \end{aligned} \tag{2.4}$$

since $\text{var}[x_{ij}] = 2p_jq_j$.

For the dominance term,

$$\mathbb{E}[t_{ij}] = -2p_j^2 \times q_j^2 + 2p_jq_j \times 2p_jq_j - 2q_j^2 \times p_j^2 = 0,$$

and

$$\begin{aligned} \text{var}[t_{ij}] =& (-2p_j^2)^2 \times q_j^2 + (2p_jq_j)^2 \times 2p_jq_j + (-2q_j^2)^2 \times p_j^2 \\ =& 4p_j^4 \times q_j^2 + 4p_j^2q_j^2 \times 2p_jq_j + 4q_j^4 \times p_j^2 \\ =& 4p_j^2q_j^2[p_j^2 + 2p_jq_j + 2q_j^2] \\ =& 4p_j^2q_j^2. \end{aligned}$$

Therefore,

$$\begin{aligned} \text{var}[D_i] =& \sum_{j=1}^{m} \delta_j^2 \text{var}[t_{ij}] \\ =& \sum_{j=1}^{m} \frac{d_j^2}{4p_j^2q_j^2} 4p_j^2q_j^2 \\ =& \sum_{j=1}^{m} d_j^2. \end{aligned} \tag{2.5}$$

2.4.1 Phenotypes in *GeneEvolve*

To simulate a phenotype, you should specify the parameters

`--file_cv_info [cv_info.txt] --file_cvs [cvs.txt]`

The file `cv_info.txt` contains CV information and has a header with 5 columns: chromosome number, base-pair distance, MAF, additive (a) and dominance (d) effects (see Example 3).

The file `--cvs.txt` has no header and contains the address of haplotype files contain the CVs. This file has just 2 columns: chromosome number and address of CV haplotype file (see Example 4).

Example 3 (CV information file format)**.** The file format for `--file_cv_info [cv_info.txt]` is illustrated in the following listing.

```
chr pos maf a d
1 4328476 0.189021 1.18585978544321 -1.54676435875486
1 4500436 0.450922 -0.128733310867769 1.11559331900794
1 7736097 0.359727 -0.379038685626258 -1.12984403982323
1 14418448 0.35875 0.0959318783552277 0.527555965661557
1 15825195 0.303792 1.89998223576411 -0.724345249882114
1 17889690 0.0623102 -0.368424537129164 -0.120587409943792
...
22 42504679 0.335594 0.990334634266996 -0.698799892553054
22 44338134 0.418218 1.26982516250768 0.990334634266996
22 47450911 0.487309 1.8384670663176 -0.625110331654888
22 49411595 0.413211 1.44121811394195 0.878797016360936
```

cv_info.txt

Example 4 (CVs file format)**.** The file format for `--file_cvs [cvs.txt]` is illustrated in the following listing.

```
1 /path/CVs.chr1.hap
2 /path/CVs.chr2.hap
...
22 /path/CVs.chr22.hap
```

cvs.txt

2.4.2 Scaling VA and VD

In Equations 2.4 and 2.5 we compute the additive and dominance variances. It is possible in *GeneEvolve* that user scale them to any arbitrary positive number using the following parameters.
`--va [number] --vd [number]`

These parameters can be used for each phenotype, if there are more than one.

2.4.3 Simulating multivariate phenotypes

To simulate k phenotypes (multivariate phenotypes) with different CVs, user should specify the parameters `--file_cv_info [cv_info.txt]` and `--file_cvs [cvs.txt]` in the command line, k times.

Example 5 (Simulating 3 phenotypes)**.** In the following listing, *GeneEvolve* will create 3 phenotypes, where their CVs are listed in different files.

```
GeneEvolve --file_gen_info gen.info --file_hap_name hap_add.txt \
--file_recom_map map.txt \
--file_cv_info cv_info_p1.txt --file_cvs cvs_p1.txt  \
--file_cv_info cv_info_p2.txt --file_cvs cvs_p2.txt  \
--file_cv_info cv_info_p3.txt --file_cvs cvs_p3.txt
```

Simulating 3 phenotypes

2.5 Random and non-random mating systems

Mating and reproductive systems affect the way that alleles are combined in individuals in a population. Outcrossing organisms put together new combinations of genes rapidly, leading to many different genotypes within populations (and creating high genotype diversity) and the potential for rapid adaptation in a changing environment.

For random mating, user should use the parameter `--RM`. If it is the case, then the second column in file `file_generaions_info.txt` inputted in parameter `--file_gen_info [file_generaions_info.txt]` will not be used. In random mating, offsprings will choose their parents randomly.

On the other hand, in assortative mating (mating based on some phenotype), we will create a random variable called mating value, MV, by combing all the k phenotypes as

$$\mathrm{MV}_i = \sum_{j=1}^{k} \omega_j P_{ij}, \tag{2.6}$$

where P_{ij} is the jth phenotype for individual i and ω_j's are some coefficients (see Figure 2.5).

Mates will chose their spouses based on the mating values, MV. User can specify the correlation (which can change across time) between mates in the second column of file `file_generaions_info.txt`. Their correlation is

$$\mu = \mathrm{corr}[\mathrm{MV}_{FA}, \mathrm{MV}_{MO}]. \tag{2.7}$$

2.6 Natural selection

Natural selection is the differential survival and reproduction of individuals due to differences in phenotype. It is a key mechanism of evolution, the change in heritable traits of a population over time. Natural variation occurs among the individuals of any population of organisms. Many of these differences do not affect survival or reproduction, but some differences may improve the chances of survival and reproduction of a particular individual.

For each individual i, the selection value can be computed from k phenotypes as

$$\mathrm{SV}_i = \sum_{j=1}^{k} \lambda_j P_{ij}, \tag{2.8}$$

where P_{ij} is the jth phenotype for individual i and λ_j's are some user define coefficients (see Figure 2.5).

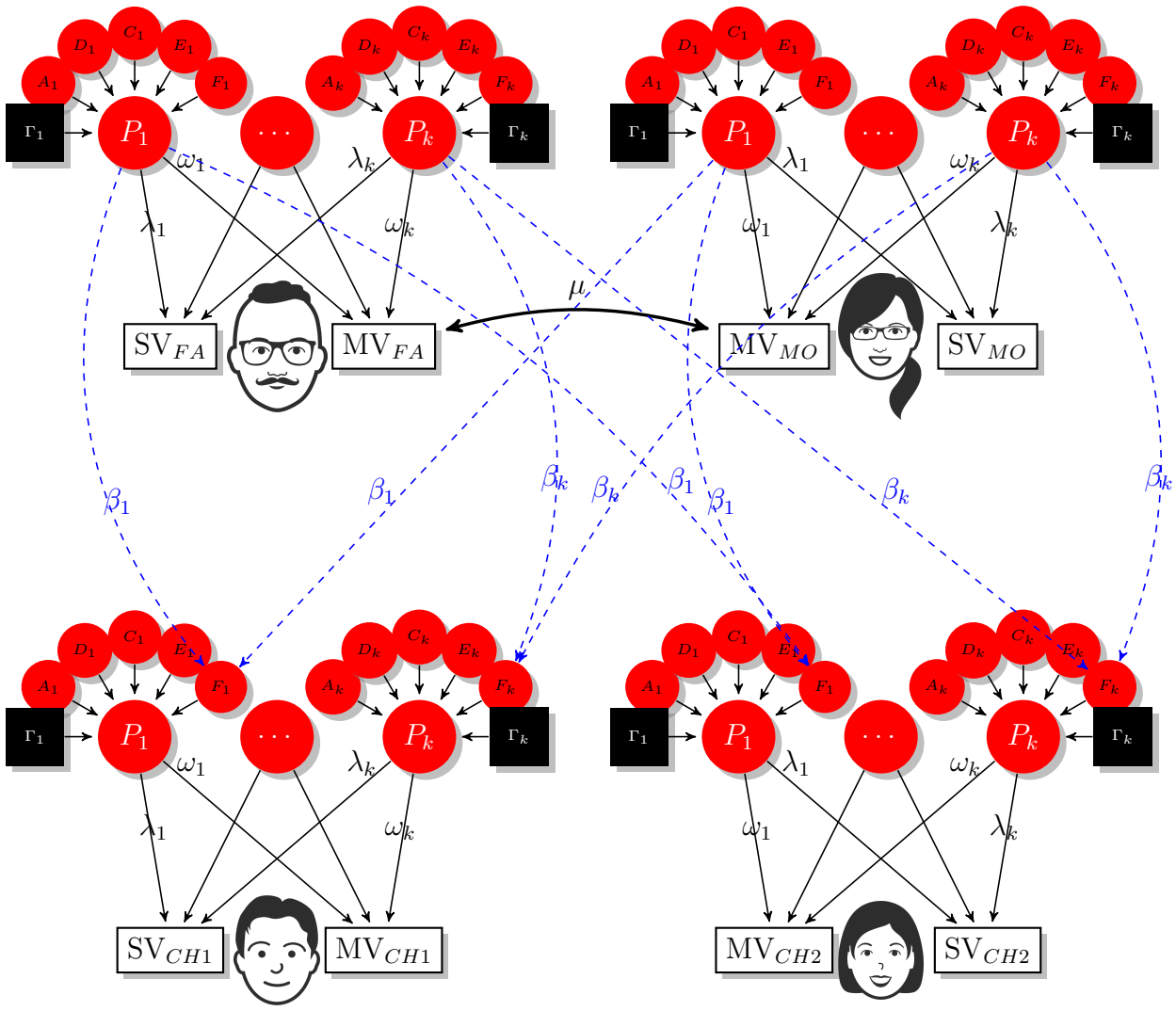


Figure 2.5: Mating path diagram and phenotyes

2.6.1 Directional selection

In population genetics, *directional selection* is a mode of natural selection in which an extreme phenotype is favored over other phenotypes, causing the allele frequency to shift over time in the direction of that phenotype. Under directional selection, the advantageous allele increases as a consequence of differences in survival and reproduction among different phenotypes. The increases are independent of the dominance of the allele, and even if the allele is recessive, it will eventually become fixed.

Based on the individuals selection value and selection function, *GeneEvolve* decides to let them marry or not.

User can define the following parameters as selection function in file `file_generaions_info.txt`

```
logit p1 p2
probit p1 p2
stab p1 p2
thr p1 p2
```

where $p1$ and $p2$ are its parameters. For more information about `file_generaions_info.txt`, see Figure 2.1.

For the `logit` function, the probability of mating can be computed from inverse *logit* function, i.e.,

$$\mathbb{P}[\text{mating}] = \frac{\exp(p_1 + p_2 \text{SV}_i)}{1 + \exp(p_1 + p_2 \text{SV}_i)}. \tag{2.9}$$

For the `probit` function, the probability of mating can be computed from inverse *probit* function, i.e., normal CDF with mean p_1 and standard deviation p_2.

2.6.2 Stabilizing selection

For the `stab` function (stabilizing selection), the probability of mating can be computed from the normal PDF with mean p_1 and standard deviation p_2.

2.6.3 Threshold selection

For the `thr` function (threshold selection), the probability of mating can be computed from

$$\mathbb{P}[\text{mating}] = \begin{cases} p_1 & \text{if } \text{SV}_i < p_2 \\ 1 & \text{if } \text{SV}_i \geq p_2 \end{cases}. \tag{2.10}$$

2.7 Familial effect

For each offspring i, the familial effect can be computed from the following equation

$$F_i = \beta \frac{P_i(FA) + P_i(MO)}{\sqrt{2}}, \tag{2.11}$$

where $P_i(FA)$ and $P_i(MO)$ are parent's phenotypes (see Figure 2.5) and β is an arbitrary coefficient defined in `--beta [number]`. User can define the variance of familial effect in *GeneEvolve* by `--vf [number]`.

If there are more than one phenotype, user can define different β's and different variances for familial effect for them.

2.8 Shared sibling (common) effect

For each sibling i in a family f, we add a random number c_f to its phenotype. The generated random number c_f comes from a standard Gaussian distribution with mean zero and the user specified variance VC. User can define VC in *GeneEvolve* by assigning a positive number in `--vc [VC]`.

If there are more than one phenotype, user can define different values for VC for each phenotype.

2.9 Environmental effects specific to each population

In this subsection we want to define the Γ_p in Equation 2.1. For simplicity, assume there is just one phenotype. For k phonetypes, the idea is the same and user should input $\Gamma^{(i)}$ for $i \in \{1, \ldots, k\}$.

Assume that there are P populations and each population p has some individuals with phenotype $P_i^{(p)}$. For a given Γ, the environmental effects specific to population p, is Γ_p, which can be obtained from solving the following equation

$$\text{var}(Y) = (1 + \Gamma)\text{var}(X), \tag{2.12}$$

where, X is all the phenotypes obtained from the combined populations, i.e.

$$Y = \bigcup_{p \in P} \bigcup_{i \in \text{pop}(p)} S_{i,p},$$

and

$$X = \bigcup_{p \in P} \bigcup_{i \in \text{pop}(p)} T_{i,p},$$

where for each population p,

$$S_i = A_i + D_i + F_i + E_i + C_f + b_p, \tag{2.13}$$

$$T_i = A_i + D_i + F_i + E_i + C_f, \tag{2.14}$$

and

$$b_p = \gamma(\frac{2(p-1)}{P-1} - 1).$$

After finding γ, the environmental effects specific to population p, becomes

$$\Gamma_p = \gamma(\frac{2(p-1)}{P-1} - 1). \tag{2.15}$$

2.10 Simulating several populations

In *GeneEvolve* you can easily simulate several population. For simulating the second population, user should use the parameter `--next_population` in order to distinguish between populations parameters. There is no limit in the number of populations, but you should use the parameter `--next_population` to separate them. See Chapter 3 for more information.

2.11 Population structure

2.11.1 Introduction

A population may have substructures – different in genetic variation among its constituent parts – for several different evolutionary reasons. Exchange of individuals may not have equal probabilities throughout a population, or selection may have different effects in different parts of population **??**. To model it, assume that a population consists of p subpopulations and that the proportion of individuals migrating from subpopulation j to subpopulation i each generation is m_{ij}. As a result there is a matrix of gene flow parameters, called the backward *migration matrix*, that describes the gene flow pattern among subpopulations (see Table 2.1). The proportion of non-migrants (or residents) for subpopulation i is given by m_{ii} (see figure 2.6). Each row of this matrix sums to unity because it describes the proportion coming from every other possible subpopulation to that particular subpopulation or

$$\sum_{j=1}^{p} m_{ij} = 1.$$

Table 2.1: The migration matrix

Subpopulations in generation $t+1$	Subpopulations in generation t				Total
	1	2	...	p	
1	m_{11}	m_{12}		m_{1p}	1
2	m_{21}	m_{22}		m_{2p}	1
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
p	m_{p1}	m_{p2}		m_{pp}	1

The columns of the matrix will generally not sum to unity. The migration matrix is denoted by

$$\mathrm{M} = \begin{bmatrix} m_{11} & m_{12} & \dots & m_{1p} \\ m_{21} & m_{22} & \dots & m_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ m_{p1} & m_{p2} & \dots & m_{pp} \end{bmatrix}.$$

Each of the subpopulations may have a different frequency of A_2, and let us indicate the allele frequency in the jth subpopulation as q_j. Therefore, the frequency of A_2 in the ith subpopulation after gene flow is

$$q_i' = \sum_{j=1}^{p} m_{ij} q_j.$$

We can symbolize the process of allele frequency change over all the subpopulations by using matrix notation. First we can indicate the migration matrix as $\boldsymbol{M}$ and the vector of allele frequency for the different subpopulations in generation t with $\boldsymbol{Q}_t$. Therefore,

$$\boldsymbol{Q}_{t+1} = \boldsymbol{M}\boldsymbol{Q}_t.$$

It is also possible that the migration matrix $\boldsymbol{M}$ changes over generation, so we denote it by $\boldsymbol{M}_t$.

$$\boldsymbol{Q}_{t+1} = \boldsymbol{M}_t\boldsymbol{Q}_t.$$

2.11.2 Population structure in *GeneEvolve*

The migration matrix can be inputed to *GeneEvolve* using the parameter

```
--file_migration [file_migration.txt]
```

The inputed file has no header and it has n rows, where n in the number of generations. This file should have k^2 columns as follows:

$$m_{11}, m_{12}, \dots, m_{1k}, m_{21}, \dots, m_{2k}, \dots, m_{k1}, \dots, m_{kk}$$

Therefore, the user can use different migration matrix $\boldsymbol{M}_t$ for each generation.

Example 6 (Migration file format)**.** In the following listing file, there are 10 generations with 2 subpopulations.

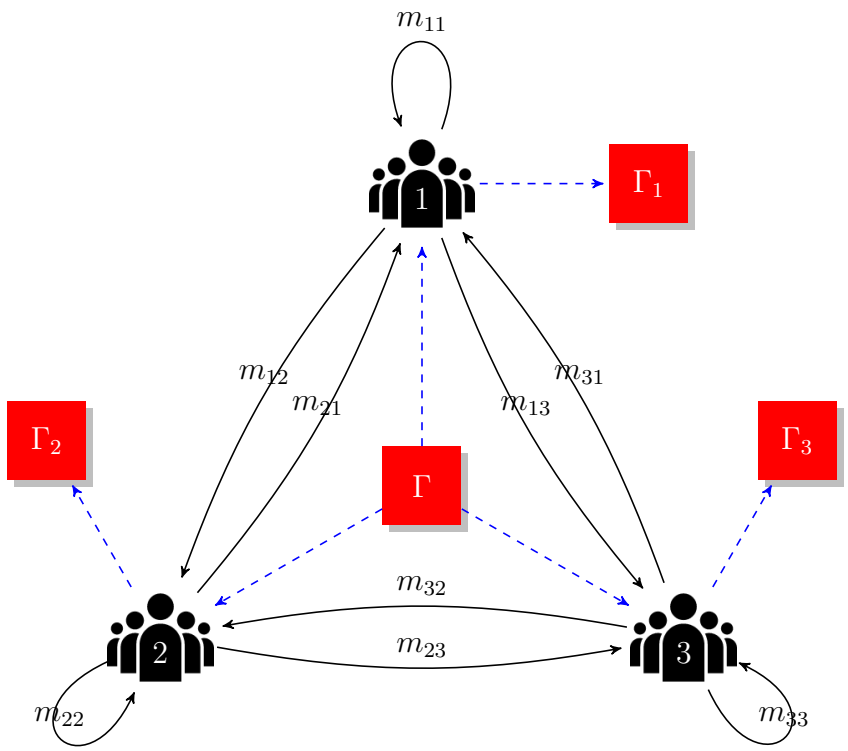


Figure 2.6: Path diagram for migration and environmental effects specific to each population

```
1 .0 .05 .95
.9 .1 .05 .95
.8 .2 .05 .95
.9 .1 .05 .95
.9 .1 .05 .95
.9 .1 .05 .95
.9 .1 .05 .95
.9 .1 .05 .95
.9 .1 .05 .95
.9 .1 .05 .95
```

file_migration.txt

For generation 3, we have

$$\boldsymbol{M}_3 = \begin{bmatrix} .8 & .2 \\ .05 & .95 \end{bmatrix}$$

Chapter 3

Quick Reference

Here is the list of all the parameters used in *GeneEvolve*.

```
GeneEvolve \
--file_gen_info par.pop1.info.txt \
--file_hap_name par.pop1.hap_sample_address.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_mutation_map mutation.Map.b37.50KbDiff \
--file_cv_info par.pop1.cv1_info.txt \
--file_cvs par.pop1.cv1_hap_files.txt \
--va 1 \
--vd .2 \
--vc .1 \
--ve 1 \
--vf .1 \
--omega .7 \
--beta .7 \
--lambda 1 \
--file_cv_info par.pop1.cv2_info.txt \
--file_cvs par.pop1.cv2_hap_files.txt \
--va 2 \
--va .1 \
--vc .1 \
--ve 1 \
--vf .1 \
--omega .3 \
--beta .3 \
--lambda 1 \
--next_population \
--file_gen_info par.pop2.info.txt \
--file_hap_name par.pop2.hap_sample_address.txt \
--file_cv_info par.pop2.cv1_info.txt \
--file_cvs par.pop2.cv1_hap_files.txt \
--file_cv_info par.pop2.cv2_info.txt \
--file_cvs par.pop2.cv2_hap_files.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_mutation_map mutation.Map.b37.50KbDiff \
--va 2 \
--vd .2 \
--vc 1 \
```

```
--ve 1 \
--vf 0 \
--va 2 \
--vd .2 \
--vc 1 \
--ve 1 \
--vf 0 \
--omega 1 \
--omega 1 \
--beta 0 \
--beta 0 \
--lambda 1 \
--lambda 1 \
--file_migration par.migration.txt \
--prefix out1 \
--gamma 1 \
--gamma 1\
--avoid_inbreeding
```

The above listing simulate 2 populations, each with 2 phenotypes. In the following subsections, we explain each of the parameters, briefly.

3.1 --file_gen_info

`--file_gen_info par.pop1.generaions_info.txt`

where `par.pop1.generaions_info.txt` is a text file with 6 columns and $n + 1$ lines, where n is the number of generations to be simulated by *GeneEvolve*. This file should have a header and the first column is the population size. The structure of this file is listed in Figure 2.1. *GeneEvolve* can simulate different population sizes in each generation, by specifying some numbers in the first column.

3.2 --file_hap_name

`--file_hap_name par.pop1.hap1_sample_address.txt`

where `par.pop1.hap1_sample_address.txt` is a text file with 4 columns, counting the address of the initial hap, legend and sample files.

3.3 --file_recom_map

`--file_recom_map Recom.Map.b37.50KbDiff`

where `Recom.Map.b37.50KbDiff` is a text file with 4 columns, counting the cM distance for all the snap panel.

3.4 --file_cv_info

`--file_cv_info par.pop1.cv1_info.txt`

where the CV information are in file `par.pop1.cv1_info.txt`.

3.5 --file_cvs

`--file_cvs par.pop1.cv1_hap_files.txt`

where the actual haplotype address are saved in file `par.pop1.cv1_info.txt`.

3.6 --va

`--va 2`

GeneEvolve will transforms the additive variance to 2, in generation zero.

3.7 --vd

`--vd 0.1`

GeneEvolve will transforms the dominance variance to 0.1, in generation zero.

3.8 --vc

`--vc 0.1`

GeneEvolve will transforms the variance of sibling environment (common) effect to 0.1, in all generations.

3.9 --ve

`--ve 1`

GeneEvolve will transforms the environmental effect variance to 1, in all generations.

3.10 --vf

`--vf 0.1`

GeneEvolve will transforms the familial effect variance to 0.1, in generation zero.

3.11 --next_population

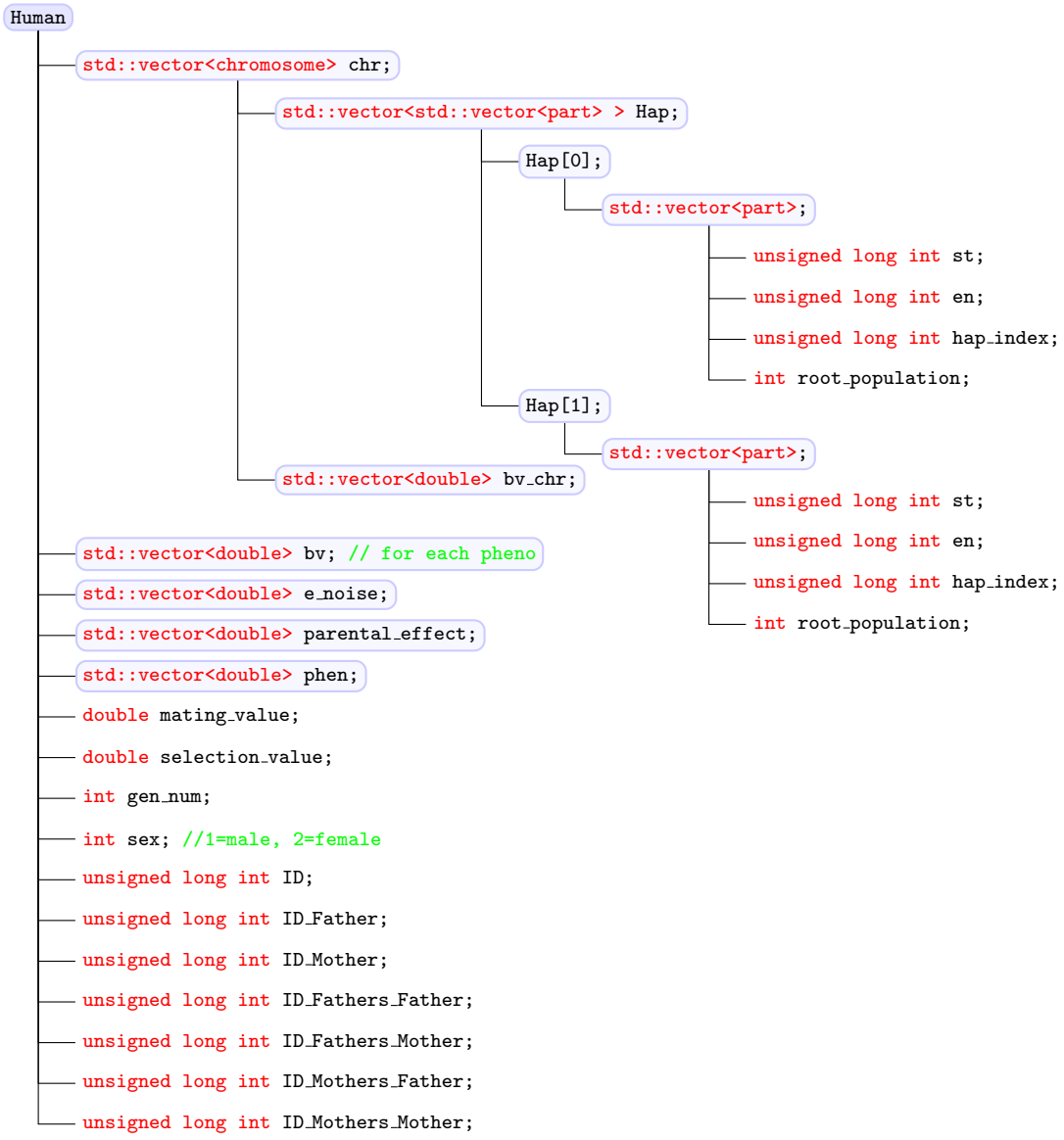
`--next_population`

This parameter will used to give the next population information.

Appendix A

C++ Classes

A.1 Human

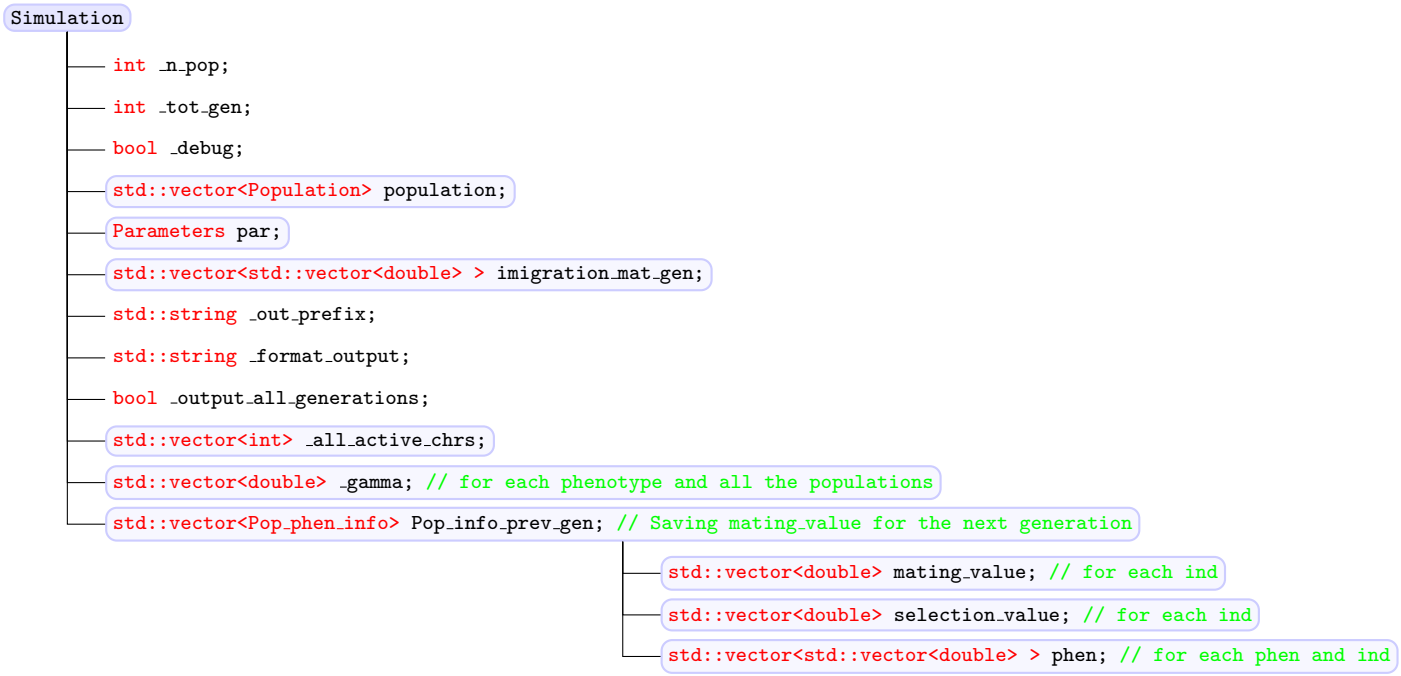


A.2 Population

```
Population
    int _pop_num;
    int _nchr;
    std::vector<Phenotype_scheme> _pheno_scheme; // for each phenotype
        std::vector<CV_INFO> _cv_info; // for each chr
            std::vector<unsigned long int> bp;
            std::vector<double> maf;
            std::vector<double> alpha;
        std::vector<CV> _cvs; // for each chr
            std::vector<std::vector<bool> > val;
        std::vector<std::string> _name_cv_hap; // for each chr
        double _va;
        double _ve;
        double _vf;
        double _alpha; // mating value coefficient
        double _beta; // transmission of environmental effects from parents to offspring
        double _delta; // selection value coefficient
    std::vector<Human> h;
    std::vector<Couples_Info> _couples_info;
        unsigned long int pos_male; // pos human, not pos hap
        unsigned long int pos_female; // pos human, not pos hap
        bool inbreed;
        int num_offspring;

    std::vector<unsigned long int> _pop_size; // for each generation
    std::vector<double> _mat_cor; // for each generation
    std::vector<std::string> _offspring_dist; // for each generation
    std::string _selection_func; // --logit 0 1
    std::vector<double> _selection_func_pars; // --logit 0 1
    std::vector<std::vector<std::string> > _hap_legend_sample_name; // for each chr, with 3 columns
    std::vector<rMap> _rmap; // for each chr
    std::vector<std::vector<double> > _recom_prob; // for each chr
    std::vector<double> _var_bv_gen0; // for each phenotype
    std::vector<int> _all_active_chrs; // for each chr
    bool _avoid_inbreeding;
    bool _no_output;
    bool _output_all_generations;
    bool _debug;
    std::string _out_prefix;
    std::string _format_output;
    double _RM_percent; // Random mating percent (inds who have 2 spouses)
    std::vector<double> ret_var_mating_value; // for each gen
    std::vector<std::vector<double> > ret_var_phen; // for each pheno and gen
    std::vector<std::vector<double> > ret_var_bv; // for each pheno and gen
    std::vector<std::vector<double> > ret_var_parental_effect; // for each pheno and gen
```

A.3 Simulation



Appendix B

File formats

B.1 Hap – Legend – Sample

B.1.1 .hap

No header.

This file is SPACE delimited. Each line corresponds to a single SNP. Each successive column pair (0, 1), (2, 3), (4, 5) and (6, 7) corresponds to the alleles carried at the 4 SNPs by each haplotype of a single individual. For example a pair "1 0" means that the first haplotype carries the B allele while the second carries the A allele as specified in the LEGEND file. The haplotypes are given in the same order than in the SAMPLE file. This file should have L lines and 2N columns, where L and N are the numbers of SNPs and individuals respectively.

	ind1.hap1	ind1.hap2	ind2.hap1	ind2.hap2	...	
snp1	0	0	1	0		
snp2	0	1	1	0		
⋮						
						nsnp × nhaps

B.1.2 .legend

Has header.

This file is SPACE delimited. The first line is a header line that describe the content of the file. Each line corresponds to a single SNP.

	col1	col2	col3	col4	
header	ID	pos	allele0	allele1	
snp1	rs17432784	17196300	T	C	
snp2	rs2845379	rs1807512	A	G	
snp3	rs17432784	17196300	G	T	
⋮					
					(nsnp+1) × 4

B.1.3 .sample

Has header.

It is SPACE delimited. The first line is a header line that describe the content of the file. Then, each line corresponds to a single individual.

	col1	col2	col3	col4	
header	sample	population	group	sex	
ind1	CEU1	CEU	EUR	1	
ind2	CEU2	CEU	EUR	2	
ind3	GBR1	GBR	EUR	2	
⋮					
					(nind+1) × 4

B.1.4 .impute.hap.indv

No header.

This file has just one column.

	col1	
ind1	5659883013-R02C01	
ind2	5648551130-R02C01	
ind3	5648560075-R02C01	
⋮		
		nind × 1

B.2 PLINK

There are two formats for PLINK: Binary format (.bed, .bim and .fam) or uncompressed format (.ped and .map).

B.2.1 .ped

No header.

Each line corresponds to a single individual.

1. Family ID
2. Sample ID
3. Paternal ID
4. Maternal ID
5. Sex (1=male; 2=female; other=unknown)
6. Affection (0=unknown; 1=unaffected; 2=affected)
7. Genotypes (space or tab separated, 2 for each marker. 0=missing)

	FID	IID	PID	MID	Sex	Aff	SNP1	SNP1	SNP2	SNP2	
IND1	fam1	ind1	0	0	1	2	A	C	C	T	
IND2	fam1	ind2	0	0	2	1	C	A	T	T	
IND2	fam2	ind1	0	0	1	1	C	C	C	T	
⋮											
											(nind) × (6+2*nsnp)

B.2.2 .map

No header.

It is SPACE delimited. Each line corresponds to a single SNP. Chromosome can be (1-22, X, Y or 0 if unplaced)

	chromosome	rs#	cM	bp	
snp1	1	rs123456	0	1234555	
snp2	1	rs234567	0	1237793	
snp3	1	rs233556	0	1337456	
⋮					
					(nsnp) × 4

B.2.3 .fam

No header.

It is SPACE delimited. Each line corresponds to a single individual.

1. Family ID ('FID')
2. Within-family ID ('IID'; cannot be '0')
3. Within-family ID of father ('0' if father isn't in dataset)
4. Within-family ID of mother ('0' if mother isn't in dataset)
5. Sex code ('1' = male, '2' = female, '0' = unknown)
6. Phenotype value ('1' = control, '2' = case, '-9'/'0'/non-numeric = missing data if case/control)

If there are any numeric phenotype values other than -9, 0, 1, 2, the phenotype is interpreted as a quantitative trait instead of case/control status. In this case, -9 normally still designates a missing phenotype;

	FID	IID	ID-Father	ID-mother	Sex	Phenotype	
ind1	1	child1	0	0	1	1	
ind2	1	child2	0	0	1	2	
ind3	2	child1	0	0	1	2	
⋮							
							(nind) × 6

Bibliography

[1] John FC Kingman. The coalescent. *Stochastic processes and their applications*, 13(3):235–248, 1982.

[2] John FC Kingman. On the genealogy of large populations. *Journal of Applied Probability*, pages 27–43, 1982.

[3] Augustine Kong, Daniel F Gudbjartsson, Jesus Sainz, Gudrun M Jonsdottir, Sigurjon A Gudjonsson, Bjorgvin Richardsson, Sigrun Sigurdardottir, John Barnard, Bjorn Hallbeck, Gisli Masson, et al. A high-resolution recombination map of the human genome. *Nature genetics*, 31(3):241–247, 2002.

[4] Zulma G Vitezica, Luis Varona, and Andres Legarra. On the additive and dominant variance and covariance of individuals within the genomic selection scope. *Genetics*, 195(4):1223–1230, 2013.