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 [3] 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.

There are three ways to download it:

1. Go to https://github.com/rtahmasbi/GeneEvolve and then push the green button ”Clone or Download” and then ”Download ZIP”.

2. You can run the following commands in the terminal:

```
git clone https://github.com/rtahmasbi/GeneEvolve
cd GeneEvolve
```

3. Or, you can run the following commands:

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

1.3 How to compile *GeneEvolve*?

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

```
make
cd bin
./GeneEvolve --help
```

You also need to load a standard C++ compiler. If you get the following error, it simply means you don't have installed ”OpenMP” for multithreading programming. This version of *GeneEvolve* does not work with multithreading and we are working on it. So, you can simply ignore it.

```
clang: error: unsupported option ’-fopenmp’
make[1]: *** [../obj/omp/Simulation.o] Error 1
```

GeneEvolve depends on the following compilers or libraries:

```
GCC 4.8.1 # or higher versions
libStatGen
Eigen # http://eigen.tuxfamily.org/
```

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 the starting generation), *recombination map*, *cv list* (position of CVs and their additive and dominance effects), and the actual *CV's haplotype*. These required parameters are listed below. The detailed explanation for each parameter is illustrated in the following chapters.

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

By default, var$[E] = 1$ and var$[A]$ and var$[D]$ are computed based on the additive and dominance effect sizes inputted by option `--file_cv_info cv.info`. A user can scale var$[A]$, var$[D]$ and var$[E]$ to any positive number. For different levels of variance of additive and dominance effects, user can use the following optional 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]`

Additional optional parameters include random mating, selection function, migration and so on, are described in Chapters 2 and 4. All the parameters and their default value are listed in Chapter 4.

1.5 Quick start

In this section, we will present several examples with different lifetime scenarios. All the required files are available in `Examples` directory. To run *GeneEvolve*, we first need genotype data. In this directory, we have simulated genotypes for 3 chromosomes. Clearly, the structure of real genotypes are more complex than this simulated data, but this simple data is useful for educational purpose.

1.5.1 Example 1 – Simple example

In this example, we simulate a population of size 3000 for 10 generations (see `ex1.popinfo.txt` file). Then we run *GeneEvolve* by the following command (you should change the `path` to the appropriate directory – If you unzip the `Example.zip` file, and then `cd Examples`, then you can use `../bin/GeneEvolve` instead of `/path/GeneEvolve`).

```
/path/GeneEvolve \
--file_gen_info ex1.popinfo.txt \
--file_hap_name par.pop1.hap_sample_address.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_cv_info cv.info \
--file_cvs par.pop1.cv_hap_files.txt \
--seed 12345 \
--prefix out.ex1
```

Code for Example 1

The population information for each generation is in the file `ex1.popinfo.txt`

```
pop_size mat_cor offspring_dist selection_func selection_func_par1 selection_func_par2
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
```

ex1.popinfo.txt

As can be seen, the first line is header and there are 10 lines (for each generation) with a population of size 3000. The columns `thr 1 1` simply means no selection. The other parameters (columns 2-6) will be explained in Chapter 2.

For each generation, the program simulates phenotypes and reports the additive (A), dominance (D), genotypic value (G=A+D), common sibling effect (C), environmental effect (E), parental effect (F), phenotypic value (P=A+D+C+E+F) for each phenotype, and reports mating value (MV), selection value (SV) and its selection function (see the `out.ex1.info.pop1.gen10.txt` listing for generation 10).

```
ID ID_Father ID_Mother ID_Fathers_Father ID_Fathers_Mother ID_Mothers_Father ID_Mothers_Mother sex ph1_A ph1_D
     ph1_G ph1_C ph1_E ph1_F ph1_P MV SV SV_f
1 1458 1693 843 1465 2331 2834 1 7.3162 -2.34847 4.96773 0 1.3855 0 6.35323 6.35323 0.345615 1
2 1458 1693 843 1465 2331 2834 1 16.2473 -0.0768416 16.1704 0 -0.467283 0 15.7032 15.7032 0.866145 1
3 1927 2090 533 2891 2550 2784 2 -4.23934 5.00247 0.763128 0 0.0717792 0 0.834907 0.834907 0.038398 1
4 815 1099 2175 2761 480 2965 2 4.91611 -1.56453 3.35158 0 -0.738968 0 2.61261 2.61261 0.137367 1
5 1951 1258 481 2733 2357 81 1 -10.3241 -1.28337 -11.6074 0 0.356487 0 -11.2509 -11.2509 -0.634446 1
6 1951 1258 481 2733 2357 81 2 -0.513121 7.27903 6.76591 0 0.667059 0 7.43297 7.43297 0.405726 1
7 1951 1258 481 2733 2357 81 2 -4.36772 2.85272 -1.515 0 -2.36603 0 -3.88103 -3.88103 -0.224148 1
8 1951 1258 481 2733 2357 81 2 -12.2801 0.0206914 -12.2594 0 -0.58552 0 -12.8449 -12.8449 -0.723186 1
9 1951 1258 481 2733 2357 81 1 28.34 2.31039 30.6504 0 -1.078 0 29.5724 29.5724 1.63827 1
...
```

out.ex1.info.pop1.gen10.txt

In order to report the same results, we use `--seed 12345`. If you do not specify it, the program will choose a random number as seed.

The program reports the summary statistics for each generation at one output file:

```
gen ph1_var_A ph1_var_D ph1_var_G ph1_var_C ph1_var_E ph1_var_F ph1_var_P ph1_h2 ph1_var_G_std var_mating_value
     var_selection_value
0 288.273 30.0114 321.072 0 1 0 322.644 0.893469 1 322.644 1
1 289.082 30.0988 320.627 0 1 0 320.877 0.900913 0.998615 320.877 0.994521
2 272.942 29.6107 300.744 0 1 0 301.769 0.904475 0.936687 301.769 0.935298
3 266.1 29.7408 298.834 0 1 0 299.073 0.889751 0.930741 299.073 0.926941
4 270.257 30.4616 306.094 0 1 0 307.375 0.879241 0.953351 307.375 0.952675
5 277.644 31.8677 305.793 0 1 0 306.063 0.907147 0.952414 306.063 0.948607
6 281.216 30.937 314.644 0 1 0 315.33 0.891813 0.979981 315.33 0.977331
7 274.844 30.6878 307.627 0 1 0 308.238 0.89166 0.958125 308.238 0.955349
8 271.603 30.668 301.843 0 1 0 302.074 0.899126 0.94011 302.074 0.936245
9 280.281 32.5815 321.548 0 1 0 321.991 0.870462 1.00148 321.991 0.997976
10 293.717 29.7898 328.268 0 1 0 328.082 0.895253 1.02242 328.082 1.01685
```

out.ex1.pop1.summary

In this example var$[E]$ = 1 by default, and var$[A]$ and var$[D]$ are computed based on the additive and dominance effect sizes inputted by option `--file_cv_info cv.info`. A user can scale var$[A]$, var$[D]$ and var$[E]$ to any positive number. Next example illustrates scaling the variances.

1.5.2 Example 2 – Scaling the variance of additive, dominance and unique environmental effects

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

```
/path/GeneEvolve \
--file_gen_info ex1.popinfo.txt \
--file_hap_name par.pop1.hap_sample_address.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_cv_info cv.info \
--file_cvs par.pop1.cv_hap_files.txt \
--va 3 \
--vd 1 \
--ve 2 \
--no_output \
--seed 12345 \
--prefix out.ex2
```

Code for Example 2

The summary statistics output file is:

```
1  gen ph1_var_A ph1_var_D ph1_var_G ph1_var_C ph1_var_E ph1_var_F ph1_var_P ph1_h2 ph1_var_G_std var_mating_value
       var_selection_value
2  0 3 1 4.05191 0 2 0 6.11812 0.490347 1 6.11812 1
3  1 3.08758 1.0306 4.12038 0 2 0 5.8962 0.523655 1.0169 5.8962 0.963728
4  2 2.92228 1.03073 4.08022 0 2 0 6.01847 0.485552 1.00699 6.01847 0.983711
5  3 2.88241 1.0063 3.81344 0 2 0 5.80154 0.496836 0.941147 5.80154 0.948255
6  4 3.05186 1.03093 4.08674 0 2 0 5.96072 0.511995 1.0086 5.96072 0.974273
7  5 2.99898 0.974803 3.9111 0 2 0 5.70741 0.525454 0.96525 5.70741 0.932869
8  6 2.80243 1.07896 3.89096 0 2 0 5.84207 0.479698 0.960279 5.84207 0.954879
9  7 2.83754 1.02503 3.8296 0 2 0 5.78649 0.490374 0.945136 5.78649 0.945796
10 8 2.81702 1.08931 3.82967 0 2 0 5.60417 0.502664 0.945154 5.60417 0.915996
11 9 2.90395 1.06076 3.93 0 2 0 5.79982 0.500698 0.969914 5.79982 0.947973
12 10 3.15495 0.988925 4.30503 0 2 0 6.36056 0.496018 1.06247 6.36056 1.03963
```

out.ex2.pop1.summary

In this listing, the variances are scaled to var$[A] = 3$, var$[D] = 1$ and var$[E] = 2$ for generation zero (initial population at line 2), and therefore the narrow sense heritability should be $3/(3+1+2) = 0.5$. The variances for the next generations are computed and scaled according to the initial values. Note that we use the option `--no_output` to save space by not creating the output hap files. To save disk space, the option `--no_output` is used to not print the output haplotype files.

1.5.3 Example 3 – No dominance effect

In order to work just with the additive effect and not dominance effect, we run the following command by setting `--vd 0`:

```
/path/GeneEvolve \
--file_gen_info ex1.popinfo.txt \
--file_hap_name par.pop1.hap_sample_address.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_cv_info cv.info \
--file_cvs par.pop1.cv_hap_files.txt \
--va 3 \
--vd 0 \
--ve 2 \
--avoid_inbreeding \
--no_output \
--seed 12345 \
--prefix out.ex3
```

Code for Example 3

The summary statistics output file is:

```
1  gen ph1_var_A ph1_var_D ph1_var_G ph1_var_C ph1_var_E ph1_var_F ph1_var_P ph1_h2 ph1_var_G_std var_mating_value
       var_selection_value
```

```
0 3 0 3 0 2 0 5.12406 0.585473 1 5.12406 1
1 3.00141 0 3.00141 0 2 0 4.99323 0.601096 1.00047 4.99323 0.974467
2 2.96684 0 2.96684 0 2 0 5.20192 0.570334 0.988945 5.20192 1.0152
3 3.00826 0 3.00826 0 2 0 5.0091 0.600559 1.00275 5.0091 0.977564
4 3.05682 0 3.05682 0 2 0 4.99159 0.612395 1.01894 4.99159 0.974146
5 3.05633 0 3.05633 0 2 0 4.99782 0.611532 1.01878 4.99782 0.975363
6 2.96539 0 2.96539 0 2 0 4.87457 0.60834 0.988464 4.87457 0.951309
7 2.85157 0 2.85157 0 2 0 4.79505 0.594692 0.950524 4.79505 0.93579
8 3.10671 0 3.10671 0 2 0 5.19461 0.598065 1.03557 5.19461 1.01377
9 2.93773 0 2.93773 0 2 0 5.00711 0.586711 0.979242 5.00711 0.977176
10 3.04165 0 3.04165 0 2 0 4.9497 0.614513 1.01388 4.9497 0.965972
```

out.ex3.pop1.summary

In this listing, the variances are scaled to var$[A] = 3$, var$[D] = 0$ and var$[E] = 2$. The option `--avoid_inbreeding` is used to not let inbreeding.

1.5.4 Example 4 – Assortative mating

In assortative mating (AM) system, couples choose each other based on the mating value, which is a function of phenotypes. For example, taller mates choose taller spouses with some correlations.

In the following example we set the mating correlation equal to 0.5. You can specify this correlation in the second column of `ex4.popinfo.txt` file, for each generation. It's known that AM, will increase the genetic variance and heritability.

```
/path/GeneEvolve \
--file_gen_info ex4.popinfo.txt \
--file_hap_name par.pop1.hap_sample_address.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_cv_info cv.info \
--file_cvs par.pop1.cv_hap_files.txt \
--va 1 \
--vd 0 \
--ve 1 \
--avoid_inbreeding \
--no_output \
--seed 12345 \
--prefix out.ex4
```

Code for Example 4

The population information for each generation is in the file `ex4.popinfo.txt`

```
pop_size mat_cor offspring_dist selection_func selection_func_par1 selection_func_par2
3000 0.5 p thr 1 1
3000 0.5 p thr 1 1
3000 0.5 p thr 1 1
3000 0.5 p thr 1 1
3000 0.5 p thr 1 1
3000 0.5 p thr 1 1
3000 0.5 p thr 1 1
3000 0.5 p thr 1 1
3000 0.5 p thr 1 1
3000 0.5 p thr 1 1
```

ex4.popinfo.txt

The summary statistics output file is:

```
gen ph1_var_A ph1_var_D ph1_var_G ph1_var_C ph1_var_E ph1_var_F ph1_var_P ph1_h2 ph1_var_G_std var_mating_value
     var_selection_value
0 1 0 1 0 1 0 2.05065 0.487651 1 2.05065 1
1 1.08515 0 1.08515 0 1 0 2.01825 0.537668 1.08515 2.01825 0.9842
2 1.18877 0 1.18877 0 1 0 2.22139 0.535145 1.18877 2.22139 1.08326
3 1.22221 0 1.22221 0 1 0 2.26725 0.539074 1.22221 2.26725 1.10562
4 1.25952 0 1.25952 0 1 0 2.30061 0.547471 1.25952 2.30061 1.12189
5 1.31114 0 1.31114 0 1 0 2.31647 0.566005 1.31114 2.31647 1.12963
6 1.30354 0 1.30354 0 1 0 2.20274 0.591782 1.30354 2.20274 1.07417
7 1.24556 0 1.24556 0 1 0 2.22085 0.560847 1.24556 2.22085 1.083
8 1.26705 0 1.26705 0 1 0 2.23781 0.566201 1.26705 2.23781 1.09127
9 1.30509 0 1.30509 0 1 0 2.29136 0.569568 1.30509 2.29136 1.11738
10 1.36352 0 1.36352 0 1 0 2.3233 0.586888 1.36352 2.3233 1.13296
```

out.ex4.pop1.summary

As it can be seen, the variance increased from 1 to 1.36352 and as a result, heritability increased from 0.487651 to 0.63.

1.5.5 Example 5 – Random mating

Another mating system is random mating (RM), where couples are chosen randomly. You can use `--RM` in *GeneEvolve*. If using RM, the second column of `ex1.popinfo.txt` will not be evaluated, since in the RM system there is no mating correlation.

```
/path/GeneEvolve \
--file_gen_info ex1.popinfo.txt \
--file_hap_name par.pop1.hap_sample_address.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_cv_info cv.info \
--file_cvs par.pop1.cv_hap_files.txt \
--va 1 \
--vd 0 \
--ve 1 \
--no_output \
--RM \
--seed 12345 \
--prefix out.ex5
```

Code for Example 5

The summary statistics output file is:

```
gen ph1_var_A ph1_var_D ph1_var_G ph1_var_C ph1_var_E ph1_var_F ph1_var_P ph1_h2 ph1_var_G_std var_mating_value
     var_selection_value
0 1 0 1 0 1 0 2.05065 0.487651 1 2.05065 1
1 0.992616 0 0.992616 0 1 0 2.03661 0.487385 0.992616 2.03661 0.993156
2 1.01697 0 1.01697 0 1 0 2.033 0.500231 1.01697 2.033 0.991392
3 0.997712 0 0.997712 0 1 0 1.97813 0.504372 0.997712 1.97813 0.964635
4 1.07132 0 1.07132 0 1 0 2.07123 0.517238 1.07132 2.07123 1.01003
5 1.02842 0 1.02842 0 1 0 2.01023 0.511593 1.02842 2.01023 0.980291
6 1.10054 0 1.10054 0 1 0 2.11954 0.519233 1.10054 2.11954 1.0336
7 1.09117 0 1.09117 0 1 0 2.10054 0.519475 1.09117 2.10054 1.02433
8 1.11951 0 1.11951 0 1 0 2.09877 0.533411 1.11951 2.09877 1.02347
9 1.07305 0 1.07305 0 1 0 1.9984 0.536954 1.07305 1.9984 0.97452
10 1.09404 0 1.09404 0 1 0 2.04759 0.534309 1.09404 2.04759 0.998508
```

out.ex5.pop1.summary

Note that if you use `--RM`, the the parameter `--avoid_inbreeding` will be ignored.

1.5.6 Example 6 – Exponential population size

In *GeneEvolve* it is possible to simulate any scenario for population growth by modifying the first column of `ex6.popinfo.txt` file. Here is an example for exponential population size.

```
/path/GeneEvolve \
--file_gen_info ex6.popinfo.txt \
--file_hap_name par.pop1.hap_sample_address.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_cv_info cv.info \
--file_cvs par.pop1.cv_hap_files.txt \
--avoid_inbreeding \
--no_output \
--seed 12345 \
--prefix out.ex6
```

Code for Example 6

The population information for each generation is in the `ex6.popinfo.txt` file:

```
pop_size mat_cor offspring_dist selection_func selection_func_par1 selection_func_par2
500 0 p thr 1 1
1000 0 p thr 1 1
2000 0 p thr 1 1
4000 0 p thr 1 1
8000 0 p thr 1 1
16000 0 p thr 1 1
32000 0 p thr 1 1
```

ex6.popinfo.txt

1.5.7 Example 7 – Bottleneck population

In this example, we simulate a bottleneck population with a sharp reduction in the size. The population size reduce to 200 in the 8th generation.

```
/path/GeneEvolve \
--file_gen_info ex7.popinfo.txt \
--file_hap_name par.pop1.hap_sample_address.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_cv_info cv.info \
--file_cvs par.pop1.cv_hap_files.txt \
--avoid_inbreeding \
--no_output \
--seed 12345 \
--prefix out.ex7
```

Code for Example 7

The population information for each generation is in the `ex7.popinfo.txt` file:

```
pop_size mat_cor offspring_dist selection_func selection_func_par1 selection_func_par2
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
3000 0 p thr 1 1
200 0 p thr 1 1
250 0 p thr 1 1
300 0 p thr 1 1
350 0 p thr 1 1
500 0 p thr 1 1
700 0 p thr 1 1
1000 0 p thr 1 1
```

ex7.popinfo.txt

1.5.8 Example 8 – Simulating several phenotypes

In this example, we simulate a population with two phenotypes. In *GeneEvolve* you can simulate any number of phenotypes simply by adding CV information with the parameters `--file_cv_info` and `--file_cvs`.

```
/path/GeneEvolve \
--file_gen_info ex1.popinfo.txt \
--file_hap_name par.pop1.hap_sample_address.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_cv_info cv.info \
--file_cvs par.pop1.cv_hap_files.txt \
--file_cv_info cv2.info \
--file_cvs par.pop1.cv2_hap_files.txt \
--avoid_inbreeding \
--no_output \
--seed 12345 \
--prefix out.ex8
```

Code for Example 8

It is possible to scale the additive variance for each phenotype using the parameter `--va`. For example, `--va 3 --va 2` sets the var$[A] = 3$ for the first phenotype and sets var$[A] = 2$ for the second phenotype. This is also true for other variances (dominance, environmental effect and so on).

The summary statistics output file is:

```
gen ph1_var_A ph1_var_D ph1_var_G ph1_var_C ph1_var_E ph1_var_F ph1_var_P ph1_h2 ph1_var_G_std ph2_var_A ph2_var_D
      ph2_var_G ph2_var_C ph2_var_E ph2_var_F ph2_var_P ph2_h2 ph2_var_G_std var_mating_value var_selection_value
0 288.273 30.0114 321.072 0 1 0 322.644 0.893469 1 335.653 82.5507 410.04 0 1 0 411.55 0.815583 1 750.889 1
1 285.529 30.6103 315.99 0 1 0 315.89 0.903887 0.984174 319.16 87.3902 401.848 0 1 0 403.903 0.79019 0.980021
     705.815 0.939972
2 281.473 29.422 305.964 0 1 0 306.946 0.917012 0.952946 310.262 85.1971 394.662 0 1 0 395.52 0.784441 0.962497
     705.468 0.939509
3 284.472 29.7745 323.048 0 1 0 323.585 0.879127 1.00616 315.256 85.6599 400.135 0 1 0 400.944 0.786284 0.975843
     730.024 0.972213
```

```
4 277.293 30.7551 315.523 0 1 0 316.513 0.876085 0.98272 317.878 78.3443 410.851 0 1 0 411.646 0.772214 1.00198 683.418 0.910145
5 290.405 31.6063 319.561 0 1 0 320.554 0.905948 0.995296 311.455 86.0347 406.524 0 1 0 407.382 0.764527 0.991424 706.894 0.941409
6 287.449 31.5157 314.514 0 1 0 315.783 0.910274 0.979577 328.588 81.8263 410.274 0 1 0 412.065 0.797419 1.00057 720.805 0.959935
7 285.576 30.8594 312.056 0 1 0 312.338 0.914317 0.971922 318.638 89.161 426.117 0 1 0 427.554 0.745257 1.03921 745.697 0.993085
8 288.398 29.9464 317.062 0 1 0 318.64 0.905093 0.987512 316.348 80.2443 381.697 0 1 0 383.557 0.824775 0.930878 704.983 0.938864
9 294.296 30.3112 318.985 0 1 0 319.554 0.920959 0.9935 314.059 80.3941 408.844 0 1 0 408.4 0.769 0.997083 739.955 0.985438
10 289.107 28.1979 320.956 0 1 0 322.193 0.89731 0.999641 322.776 83.9232 414.476 0 1 0 415.596 0.776658 1.01082 723.945 0.964116
```

out.ex8.pop1.summary

As it can be seen, there are two phenotypes (with prefixes `ph1_` and `ph2_`). Note that in the multi-phenotype simulations, the selection value and mating value are the sum of the phenotypes. You can easily choose different weights for them using the parameters `--lambda` and `--omega` (see figure 2.6 and chapter 4). By default, `--lambda 1` and `--omega 1` for each phenotype.

1.5.9 Example 9 – Selection

In this example, we simulate a population with an under selection phenotype. For each generation, you can choose different selection functions and different parameters in `ex9.popinfo.txt` file (columns 4–6) using the parameter `--file_gen_info`.

```
/path/GeneEvolve \
--file_gen_info ex9.popinfo.txt \
--file_hap_name par.pop1.hap_sample_address.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_cv_info cv.info \
--file_cvs par.pop1.cv_hap_files.txt \
--avoid_inbreeding \
--va 1 \
--vd 0 \
--ve 1 \
--no_output \
--seed 12345 \
--prefix out.ex9
```

Code for Example 9

The population information for each generation is in the `ex9.popinfo.txt` file:

```
pop_size mat_cor offspring_dist selection_func selection_func_par1 selection_func_par2
3000 0 p logit 20 0
3000 0 p logit 10 5
3000 0 p logit 1 1
3000 0 p probit 0 1
3000 0 p probit 0 2
3000 0 p stab 0 1
3000 0 p stab 1 4
3000 0 p thr .8 3
3000 0 p thr .8 10
```

```
3000 0 p thr .9 20
```

ex9.popinfo.txt

For the definition of `logit`, `probit`, `stab` and `thr` functions see Section 2.6.

1.5.10 Example 10 – Shared haplotypes by descent

One the most important features of *GeneEvolve* is the ability to report the *shared haplotypes by descent*. A user can simply save these information by adding the parameter `--interval` to the command line. For example, after running the following code, the program will create some output files with ".int" extension.

```
/path/GeneEvolve \
--file_gen_info ex1.popinfo.txt \
--file_hap_name par.pop1.hap_sample_address.txt \
--file_recom_map Recom.Map.b37.50KbDiff \
--file_cv_info cv.info \
--file_cvs par.pop1.cv_hap_files.txt \
--seed 12345 \
--interval \
--prefix out.ex10
```

Code for Example 10

A part of the `out.ex10.pop1.gen10.chr1.int` file is:

```
h_ID chr hap st en hap_index root_pop
0 1 0 738555 3558453 422 0
0 1 0 3558453 3701679 1038 0
0 1 0 3701679 5659393 41 0
0 1 0 5659393 8444886 2594 0
0 1 0 8444886 20130195 1931 0
0 1 0 20130195 29816837 144 0
0 1 0 29816837 32386428 2261 0
0 1 0 32386428 34087448 3027 0
0 1 0 34087448 53933548 3284 0
0 1 0 53933548 75176267 1691 0
0 1 0 75176267 82013636 1849 0
0 1 0 82013636 89729541 1691 0
0 1 0 89729541 104759007 28 0
0 1 0 104759007 108110606 3553 0
0 1 0 108110606 108476312 961 0
0 1 0 108476312 109417742 2790 0
0 1 0 109417742 113920490 3562 0
0 1 0 113920490 116280903 2469 0
0 1 0 116280903 126934664 1955 0
...
```

out.ex10.pop1.gen10.chr1.int

Based on this output, the first haplotype (0) of the chromosome 1 for the individual with ID 0, consists of some nucleotide sequences starting at position 738555 (base-pair position), and the nucleotides in the interval $[738555, 3558453)$ descended down from the haplotype 422 at the initial population. So, this haplotype can be represented by half-open intervals of the form $[738555, 3558453)$, $[3558453, 3701679)$, $[3701679, 5659393)$, $[5659393, 8444886)$, $[8444886, 20130195)$,

$[20130195, 29816837), \cdots$ where these interval haplotypes descended down from different individuals at the initial population with the haplotype IDs $422, 1038, 41, 2594, 1931, 144, \cdots$, respectively. A mathematical definition of this interval representation is presented in Section 2.3.

If the nucleotide sequences in a segment for two individuals have inherited from a common ancestor, then this segment is called identical by decent (IBD), that is, the segment has the same ancestral origin in these two individuals.

Given a ”.int” file as input, the program ”SharedHaplotypes” available at
https://github.com/rtahmasbi/IBG-SharedHaplotypes
can extract the true shared haplotypes between each two individuals and can report the true IBD segments.

1.5.11 Example 11 – Comprehensive example

For a comprehensive example, we simulate two populations, each with two phenotypes, in Chapter 4. The initial inputed genotype files are two different datasets. The variance components for each phenotype are scaled. Populations are under selection with different selection functions. The selection functions also change over time (over generations). The selection and mating values are computed with unequal weights. The mating system is assortative mating while avoiding inbreeding. There are also familial and siblings effects. The first phenotype has more effect on the familial effect. Populations migrate with different migration rates per generation.

1.6 Summary of features

In Table 1.1, we summarizes the basic features of *GeneEvolve*.

Table 1.1: Current features of *GeneEvolve*

- Specifiable duration of simulation (in synchronous generations)
- Univariate or multivariate phenotypes
- Additive and dominance genetic effects
- Multiple types of familial effects and unique environmental effects
- Migration between two or more populations
- Random mating or user-specified mate correlations (assortative mating)
- Monogamous or polygamous mating system
- Multiple genes and chromosomes, of arbitrary number and length
- Diploid populations
- Fixed or Poisson family size distribution
- User defined population size
- Phenotype-based natural selection of user-specified type
- Point and interval mutation rate
- Point and interval recombination rate
- Two types of familial environmental effects
- Dealing with large sequence datasets
- Input as hap format
- Output individual phenotype files and individual genotype files in hap, PLINK and plain text formats

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 [4]).

This assumes discrete, non-overlapping generations $G_0, G_1, G_2, \dots$ 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).

```
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.

The mean of Poisson distribution in each generation is obtained by

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

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 the initial population

GeneEvolve works with the haplotypes. For more information about haplotype file format, see the appendix C. 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
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].

```
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

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.

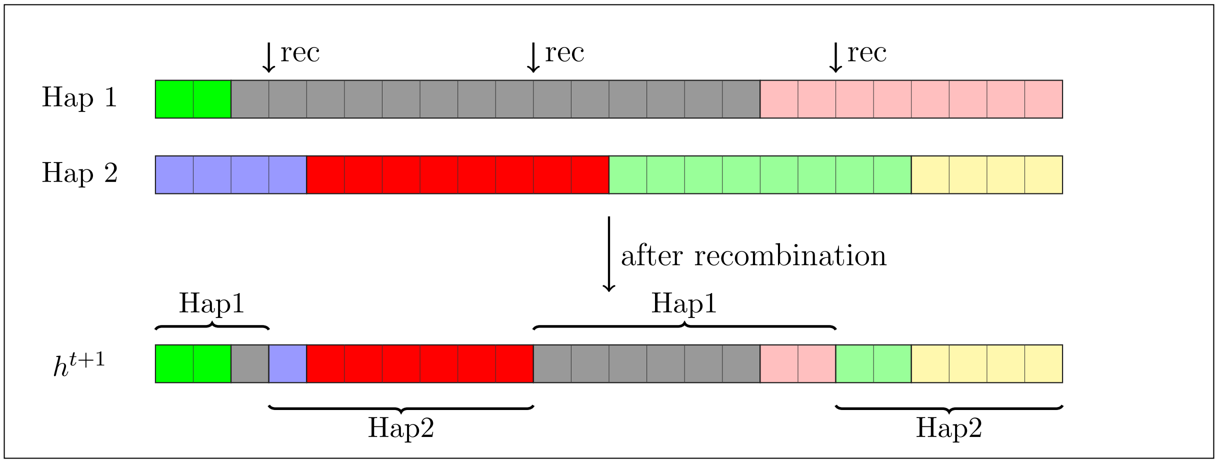


Figure 2.3: Recombination model: three recombination occurred at different positions. Each color code is an ancestral IBD.

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 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

$$h_i^{t+1} = \{[1, r_1) \cup [r_1, r_2) \cup [r_2, w) \cup [w, s_4) \cup [s_4, s_5) \cdots \cup [s_{u-1}, L]\}. \tag{2.1}$$

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. [5] 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.).

2.4.1 Genotypic value for single locus

If we could replicate a particular genotype in a number of individuals and measure them under environmental conditions normal for the population, their mean environmental deviation would be zero, and their mean phenotypic value would consequently be equal to the genotypic value of that particular genotype. This is the meaning of the genotypic value of an individual. In principle it is measurable, but in practice it is not, except when we are concerned with a single single

Table 2.1: Values of genotypes in a two-allele system, measured as deviations from the population mean. The population mean is $a(p-q)+2pqd$, and $\alpha = a + d(q-p)$.

	Genotypes		
	A_1A_1	A_1A_2	A_2A_2
Frequencies	p^2	$2pq$	q^2
Assigned values	a	d	$-a$
Deviation from population mean	$2q(a-pd)$	$a(q-p)+d(1-2pq)$	$-2p(a+qd)$
or	$2q(\alpha-qd)$	$\alpha(q-p)+2pqd$	$-2p(\alpha+pd)$
Breeding value	$2q\alpha$	$(q-p)\alpha$	$-2p\alpha$
Dominance deviation	$-2q^2d$	$2pqd$	$-2p^2d$

locus where the genotypes are phenotypically distinguishable, or with the genotypes represented in highly inbred lines [2].

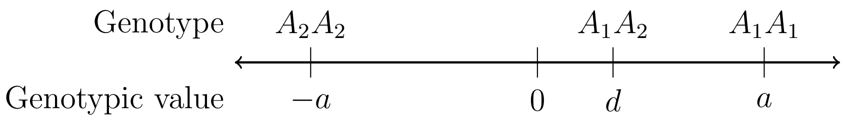


Figure 2.4: Arbitrary assigned genotypic value.

For the purposes of deduction we must assign arbitrary values to the genotypes under discussion. This is done in the following way. Considering a single locus with two alleles A_1 and A_2, we call the genotypic value of one homozygote $+a$, that of the other homozygote $-a$ and that of the heterozygote d. (We shall adopt the convention that A_1 is the allele that increase the the value). We thus have a scale of genotypic values as in figure 2.4. The origin, or point of zero value, on this scale is mid-way between the values of the two homozygotes. The value d of the heterozygote depends on the degree of dominance. If there is no dominance, $d = 0$; if A_1 is dominance over A_2, d is positive, and if A_2 is dominant over A_1, d is negative. If dominance is complete, d is equal to $+a$ or $-a$, and if there is overdominance, d is greater than $+a$ or less than $-a$. The degree of dominance may be expressed as d/a. The deviation from population mean is summarized in Table 2.1.

2.4.2 One phenotype

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

$$P_i = A_i + D_i + F_i + E_i + C_f + \gamma_p \tag{2.2}$$

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 *causal variants* (CVs), $\{\text{cv}_j\}_{j=1}^m$, and their additive (a_j) and dominance (d_j) effects. The value a_j and d_j are defined in Section 2.4.1. Typically, a_j and d_j are chosen such that $a_j \sim N(0, \sigma_g^2)$ and $d_j \sim N(0, \sigma_d^2)$.

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

$$A_i = \sum_{j=1}^{m} \frac{(x_{ij} - 2p_j)}{\sqrt{2p_j q_j}} \alpha_j, \tag{2.3}$$

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 2.5.

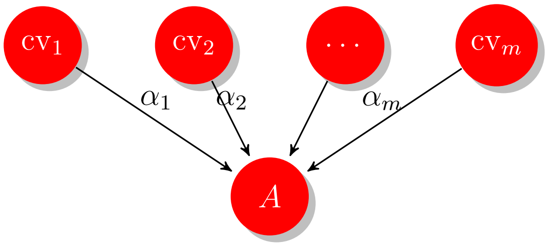


Figure 2.5: Additive term

For each individual i, the dominance term D_i is,

$$D_i = \sum_{j=1}^{m} t_{ij} \delta_j, \tag{2.4}$$

where

$$\delta_j = \frac{d_j}{2p_j q_j},$$

is the dominance effect size, and

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

For more information, see [2, 10].

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

$$\begin{aligned} \text{var}[A_i] &= \sum_{j=1}^{m} \alpha_j^2 \text{var}[\frac{x_{ij} - 2p_j}{\sqrt{2p_j q_j}}] \\ &= \sum_{j=1}^{m} [a_j + d_j(q_j - p_j)]^2, \end{aligned} \tag{2.5}$$

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,

$$\begin{aligned}\text{var}[D_i] =&\sum_{j=1}^{m} \delta_j^2 \text{var}[t_{ij}]\\ =&\sum_{j=1}^{m} \frac{d_j^2}{4p_j^2q_j^2} 4p_j^2q_j^2\\ =&\sum_{j=1}^{m} d_j^2. \end{aligned} \tag{2.6}$$

2.4.3 Heritability

Heritability estimates how much of the phenotypic variation can be explained by genetic, or genetic-environmental effects. Broad-sense heritability (H^2) refers to the inclusion of all potential sources of genetic variation (additive, dominance, epistatic, maternal and paternal effects):

$$H^2 = \frac{V_G}{V_P} = \frac{V_A + V_D}{V_P},$$

where $V_A := \text{var}[A]$, $V_D := \text{var}[D]$ and so on.

To only know the ratio of additive genetic variation to the total phenotypic variation observed, V_A can be used in the equation instead of V_G, and this becomes the narrow-sense heritability (h^2):

$$h^2 = \frac{V_A}{V_P}.$$

2.4.4 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

2.4.5 Scaling VA and VD

In Equations 2.5 and 2.6 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]`

If there are more than one phenotype, the user can add more scaling parameters.

2.4.6 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

$$\mathrm{MV}_i = \sum_{j=1}^{k} \omega_j P_{ij}, \tag{2.7}$$

where P_{ij} is the jth phenotype for individual i and ω_j's are some coefficients (see Figure 2.6).

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

$$\mu = \mathrm{corr}[\mathrm{MV}_{FA}, \mathrm{MV}_{MO}]. \tag{2.8}$$

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.

Natural selection is a process that favors or induces survival and perpetuation of one kind of organism over others. Selection can be positive (or advantageous) or negative (or purifying) and has a profound impact on the evolution of the human population. In addition, selection can be balancing in which the genotypes have a mixture of positive and negative selection pressures so that there is no net effect of selection on the individual alleles.

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

$$\mathrm{SV}_i = \sum_{j=1}^{k} \lambda_j P_{ij}, \tag{2.9}$$

where P_{ij} is the jth phenotype for individual i and λ_j's are some user define coefficients (see Figure 2.6).

2.6.1 No selection

In order to have no selection, you can easily use the `thr 1 1` in columns 4-6 of `file_generaions_info.txt` file (see Example 1.5.1).

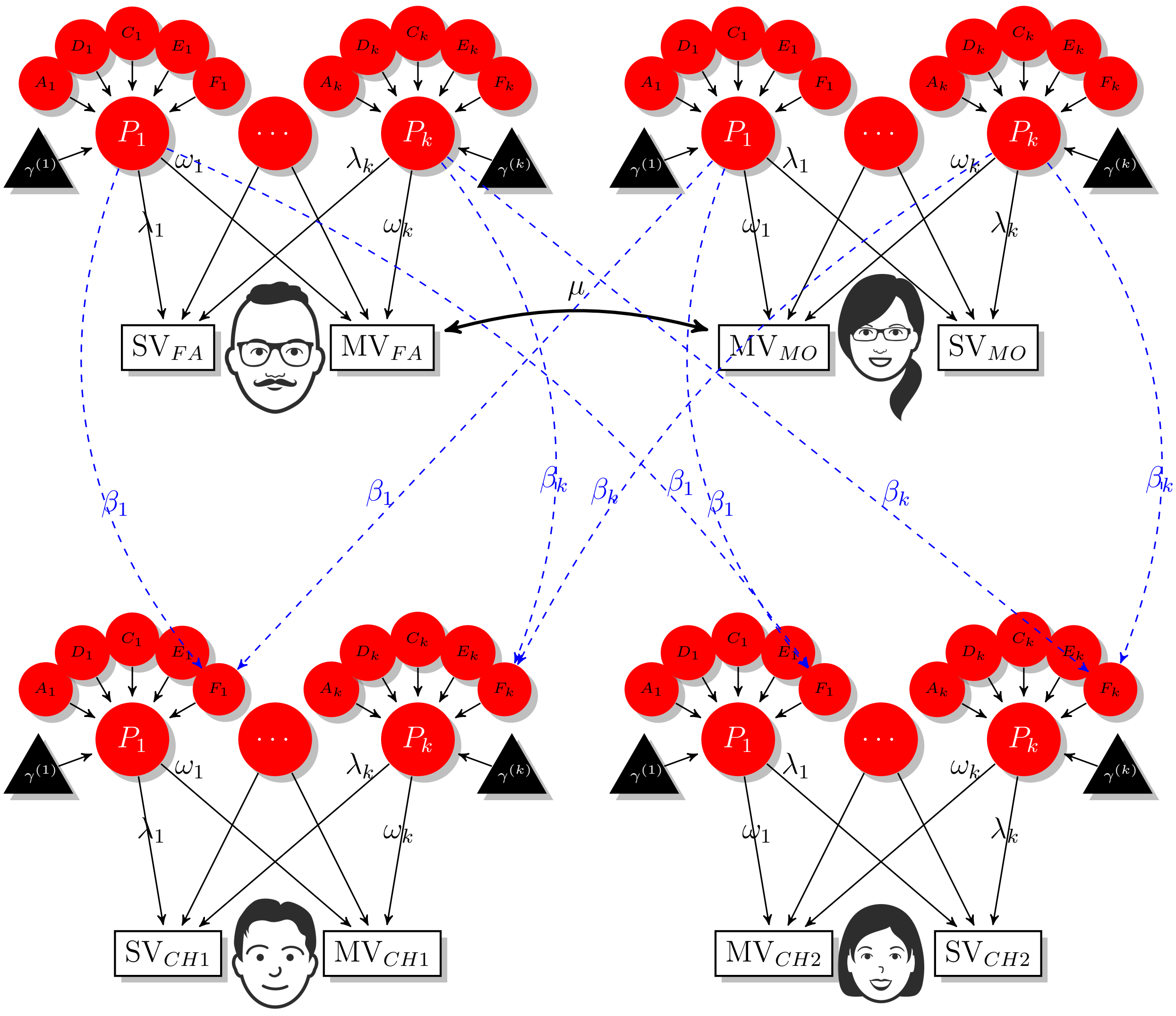


Figure 2.6: Mating path diagram and phenotyes

2.6.2 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` (columns 4-6):

```
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.,

$$\mathbb{P}[\text{mating}] = \frac{\exp(p_1 + p_2\text{SV}_i)}{1 + \exp(p_1 + p_2\text{SV}_i)}. \tag{2.10}$$

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.

2.6.3 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.

2.6.4 Threshold selection

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

$$\mathbb{P}[\text{mating}] = \begin{cases} p_1 & \text{if } \text{SV}_i < p_2 \\ 1 & \text{if } \text{SV}_i \geq p_2 \end{cases}. \tag{2.11}$$

2.7 Familial effect

Sometimes non-genetics characters run in families and influence the phenotypes of offsprings. For example, families with higher education tend to have more educated offspring. This is also true form wealth. We can model this familial effect in *GeneEvolve*, easily.

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

$$F_i = \beta\frac{P_i(FA) + P_i(MO)}{\sqrt{2}}, \tag{2.12}$$

where $P_i(FA)$ and $P_i(MO)$ are parent's phenotypes (see Figure 2.6) 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

Shared sibling (common) effect is defined as the environmental effects shared by groups of individuals, for example effects shared by groups of relatives that are not due to genetic effects.

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

The aim of this subsection is to define the γ_p term in Equation 2.2. For simplicity, assume there is just one phenotype. For k phonetypes, the idea is the same and user should input $\gamma^{(i)}$ for $i \in \{1, \dots, k\}$ (see Figure 2.6).

Assume that there are P populations. For a user-specified γ, the environmental effects specific to a population $p \in P$, is γ_p, which can be obtained from solving the following equation

$$\text{var}(Y) = (1 + \gamma)\text{var}(X), \tag{2.13}$$

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 individual i in population p,

$$S_{i,p} = A_i + D_i + F_i + E_i + C_f + b_p, \tag{2.14}$$

$$T_{i,p} = A_i + D_i + F_i + E_i + C_f, \tag{2.15}$$

where

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

After solving for Γ, the environmental effects specific to population p, becomes

$$\gamma_p = \Gamma(\frac{2(p-1)}{P-1} - 1). \tag{2.16}$$

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 4 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

Table 2.2: The migration matrix

Subpopulations	Subpopulations in generation t				Total
in generation $t+1$	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

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 2.2). The proportion of non-migrants (or residents) for subpopulation i is given by m_{ii} (see figure 2.7). Each row of this matrix sums to unity because it describes the proportion coming from every other possible subpopulation to that particular subpopulation or

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

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]
```

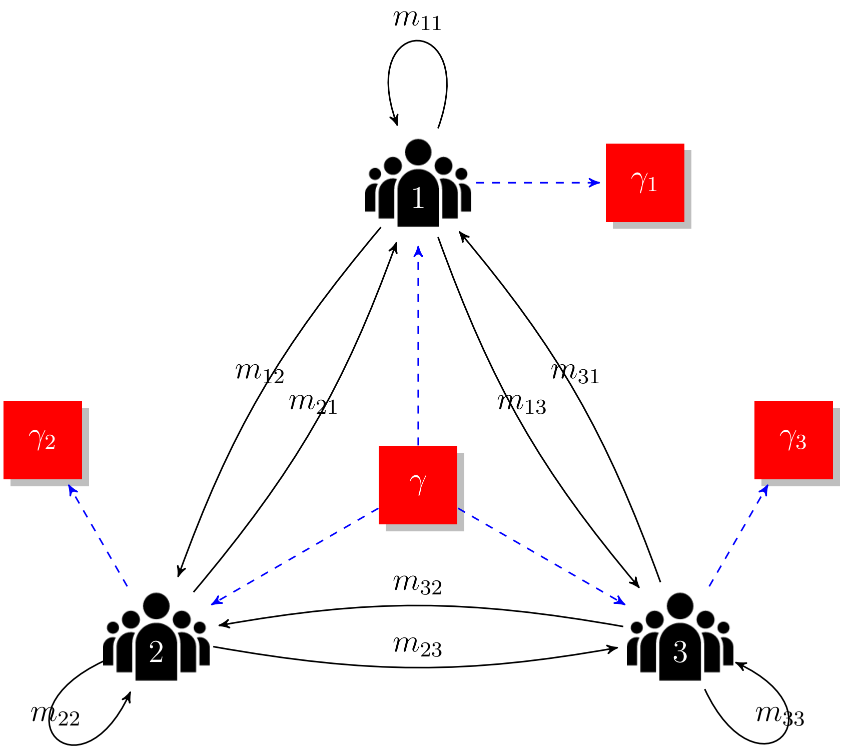


Figure 2.7: Path diagram for migration and environmental effects specific to each population

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 where k is the number of populations:

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

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.

```
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

$$\boldsymbol{M}_3 = \left[\begin{array}{cc} .8 & .2 \\ .05 & .95 \end{array} \right]$$

Chapter 3

Results

GeneEvolve is a stand-alone program. Its main advantages are its speed, memory efficiency, ability to simulate realistic SNP and/or sequence data given potentially complex evolutionary events, and ease of use. *GeneEvolve* has been developed and tested under a Linux and Mac environments. It can also be installed on a Windows platform.

simuPOP [8] is a forward-time program that can also simulate complex evolutionary events. However, it is slower and requires knowledge of Python programming.

3.1 Time and memory

In order to show the application of *GeneEvolve* on real data, we use a sample of 33,253 individuals (after cleaning and phasing genotypes) from 9 datasets in the NCBI dbGaP Database [6, 9] and, a sample of 3,781 individuals from the UK10K project [7, 1] as the initial reference panels (see Acknowledgements).

The time and memory used by *GeneEvolve* per generation and under different population sizes are reported in Table 3.1, with both SNP and sequence data as reference panels. In this table, we just report the time used by the main body of simulation and not consider the reading and writing whole genome SNP/sequence data, which depend on the physical computer system configurations.

Table 3.1: Average time (seconds) and memory used (Megabytes) in *GeneEvolve* per generation. For the whole-genome SNP data 320,926 SNPs are used and for the whole-genome sequence data 22,989,093 SNPs are used.

Population size	Type	Spousal correlation	Memory used (MB)	Time (s)*
3,000	SNP	0	28.8	5.8
3,000	SNP	0.4	29.6	5.5
30,000	SNP	0	254.9	57.7
30,000	SNP	0.4	257.3	56.2
300,000	SNP	0	2,526.4	1,121.8
300,000	SNP	0.4	2,530.1	991.8
300,000	SEQ	0	2,566.3	1,277.5

* The time for reading and writing genome SNP/sequence data is not considered.

3.2 Minor allele frequency

Under random mating and no selection model, the changes in minor allele frequency (MAF) for five randomly selected SNPs over 40 generation is plotted in Figure 3.1. In this example, we used a simulated genotype as the initial reference panel.

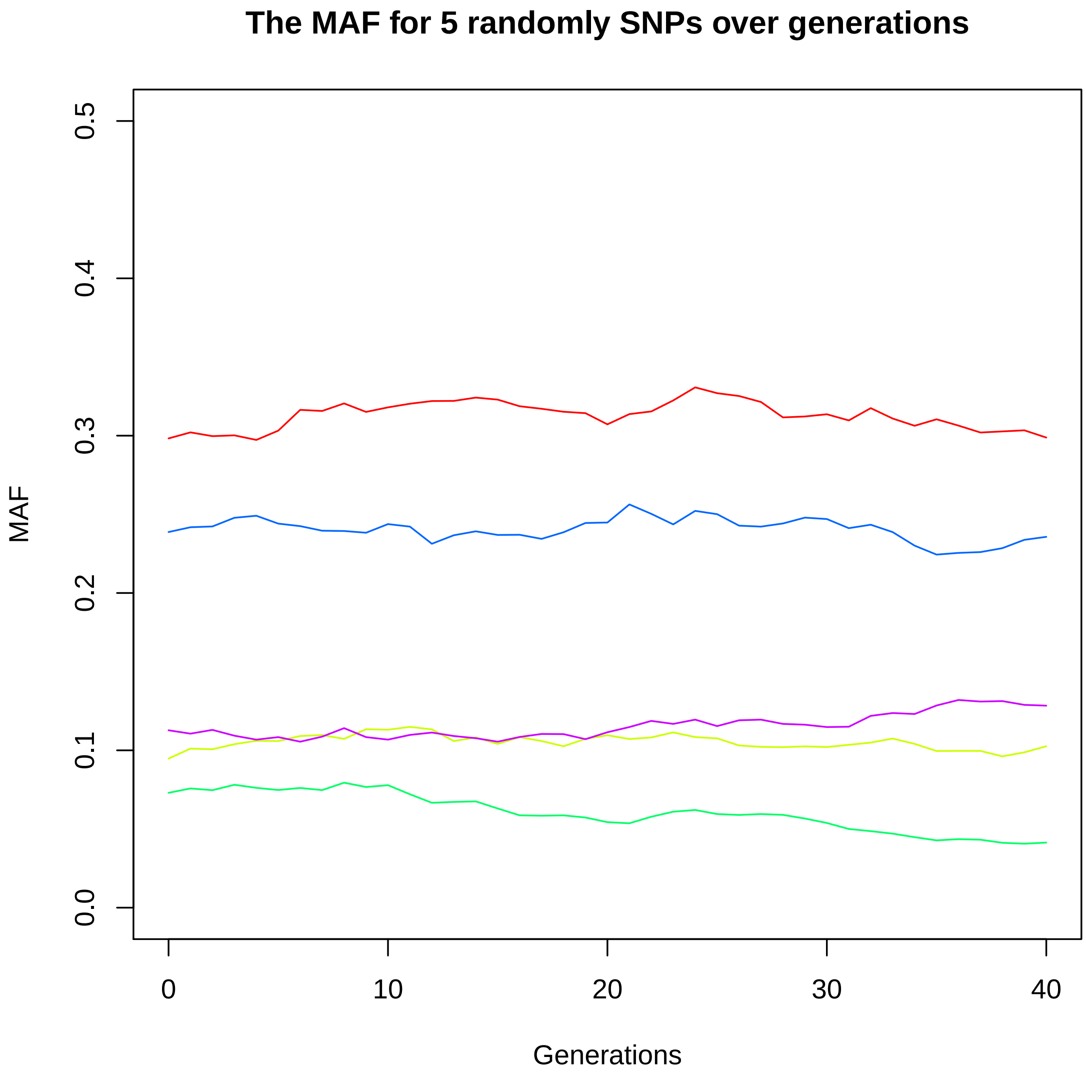


Figure 3.1: Minor allele frequency (MAF) for five randomly selected SNPs over 40 generation.

3.3 The effects of genetic drift on genetic variation

The *heterozygosity* (or allelic diversity) of the population in generation t is defined as

$$H_t^0 = \frac{2X_t(2N - X_t)}{2N(2N-1)} = \frac{2N}{2N-1} 2p_t(1-p_t), \tag{3.1}$$

Table 3.2: Summary statistics for MSEs in each category of SNPs (chromosome 22).

Percentile	Mean	Std	Median	Min	Max	# SNPs
10	1.40e-04	1.63e-04	8.59e-05	6.21e-06	1.32e-03	477
20	1.57e-04	1.79e-04	9.23e-05	7.80e-06	1.17e-03	476
30	1.79e-04	2.25e-04	9.63e-05	8.47e-06	1.68e-03	476
40	1.77e-04	2.07e-04	1.05e-04	8.84e-06	1.55e-03	480
50	1.53e-04	1.68e-04	9.25e-05	8.33e-06	1.16e-03	473
60	1.28e-04	1.46e-04	7.72e-05	5.64e-06	1.04e-03	478
70	9.74e-05	1.23e-04	5.12e-05	4.10e-06	1.07e-03	475
80	5.33e-05	6.33e-05	3.34e-05	1.40e-06	5.08e-04	477
90	2.06e-05	2.81e-05	1.21e-05	4.77e-07	2.71e-04	477
100	4.65e-06	1.05e-05	1.25e-06	3.53e-08	1.32e-04	476
All	1.11e-04	1.61e-04	5.21e-05	3.53e-08	1.68e-03	4766

where X_t is the number of copies of allele A in generation t, N is the population size, and $p_t = X_t/2N$. As is clear from the first definition, the heterozygosity is equal to the probability of sampling both alleles when we sample two chromosomes at random and without replacement from the population.

It is well known that under the Wright–Fisher model, the expected heterozygosity $h(t) = \mathbb{E}[H_t^0]$ decreases geometrically at rate $(1 - 1/2N)$:

$$h(t) = \left(1 - \frac{1}{2N}\right)^t h(0). \tag{3.2}$$

In this section, we simulate a population of size 33,253 individuals under the Wright–Fisher model for 100 generations starting from NCBI dbGaP Database [6, 9] as the reference panel. In order to check the effects of genetic drift on genetic variation, we computed the heterozygosity (eq 3.1) and the the expected heterozygosity (eq 3.2).

For each SNP, we define the mean square difference between equations 3.1 and 3.2 as

$$MSE = \frac{1}{g}\sum_{t=1}^{g}(H_t^0 - h(t))^2, \tag{3.3}$$

where g is the number of generations (here, $g = 100$).

Figures 3.2 and 3.3 shows the heterozygosity and its expected value for two randomly selected SNP. As can be seen, their MSE is very small. In Table 3.2 we categorized SNPs based on their MAF using their decile and reported the summary statistics for MSEs in each category. This table just shows the results of chromosome 22, the results for other chromosomes were the same. Clearly, the difference between heterozygosity and its expected value is negligible for each quantile category and for all the SNPs (last line).

3.4 LD structure

In this section, we simulate a population of size 10,000 individuals under the Wright–Fisher model for 30 generations starting from NCBI dbGaP Database [6, 9] as the reference panel. In order to check the LD structure, we computed the r^2 in PLINK using the following parameters for chromosome 1:

```
--r2 --ld-window-r2 0 --ld-window 100 --ld-window-kb 100000
```

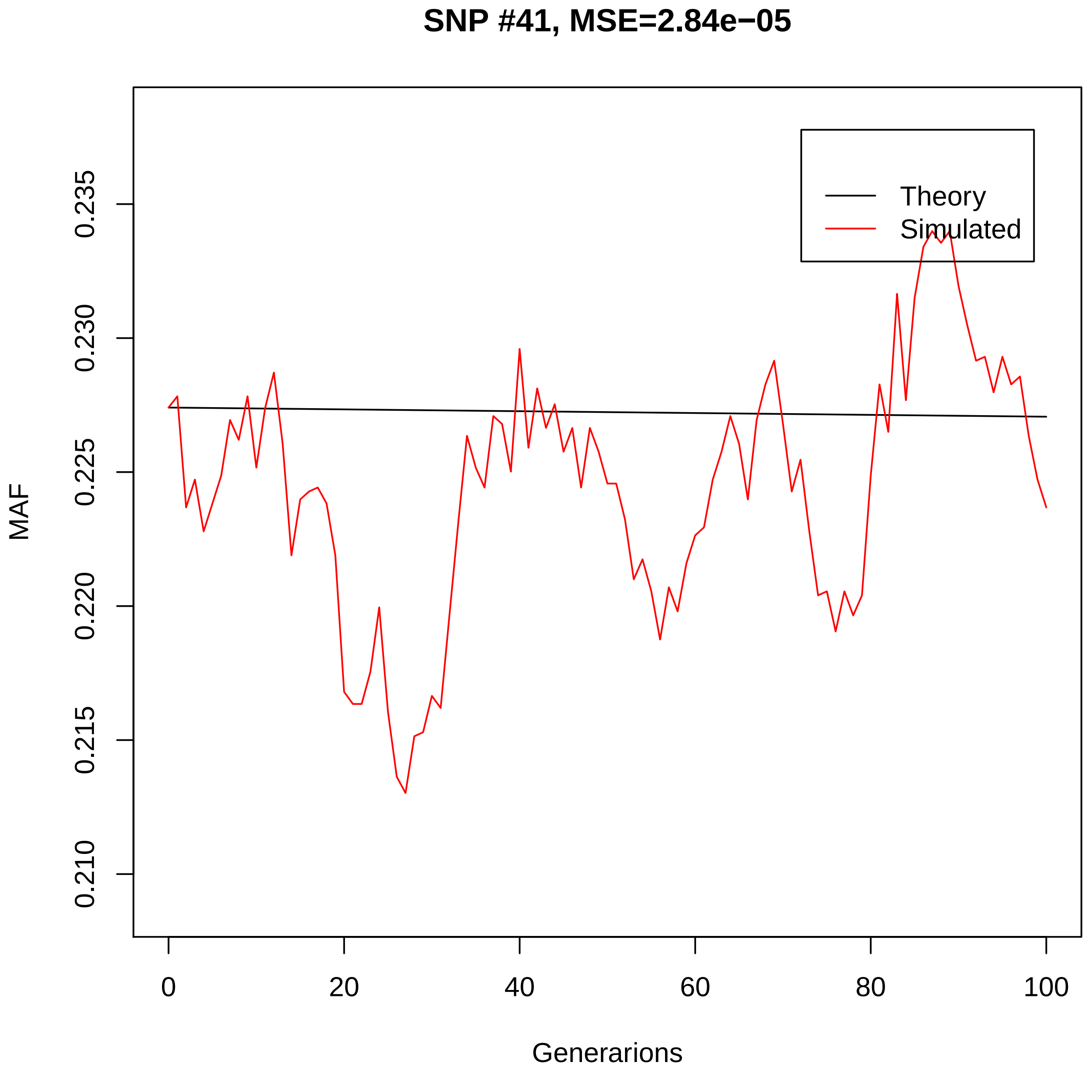


Figure 3.2: Comparing the the simulated heterozygosity (red) and the its expected value (black) obtained by theory per generation. The MSE is 2.84e-05

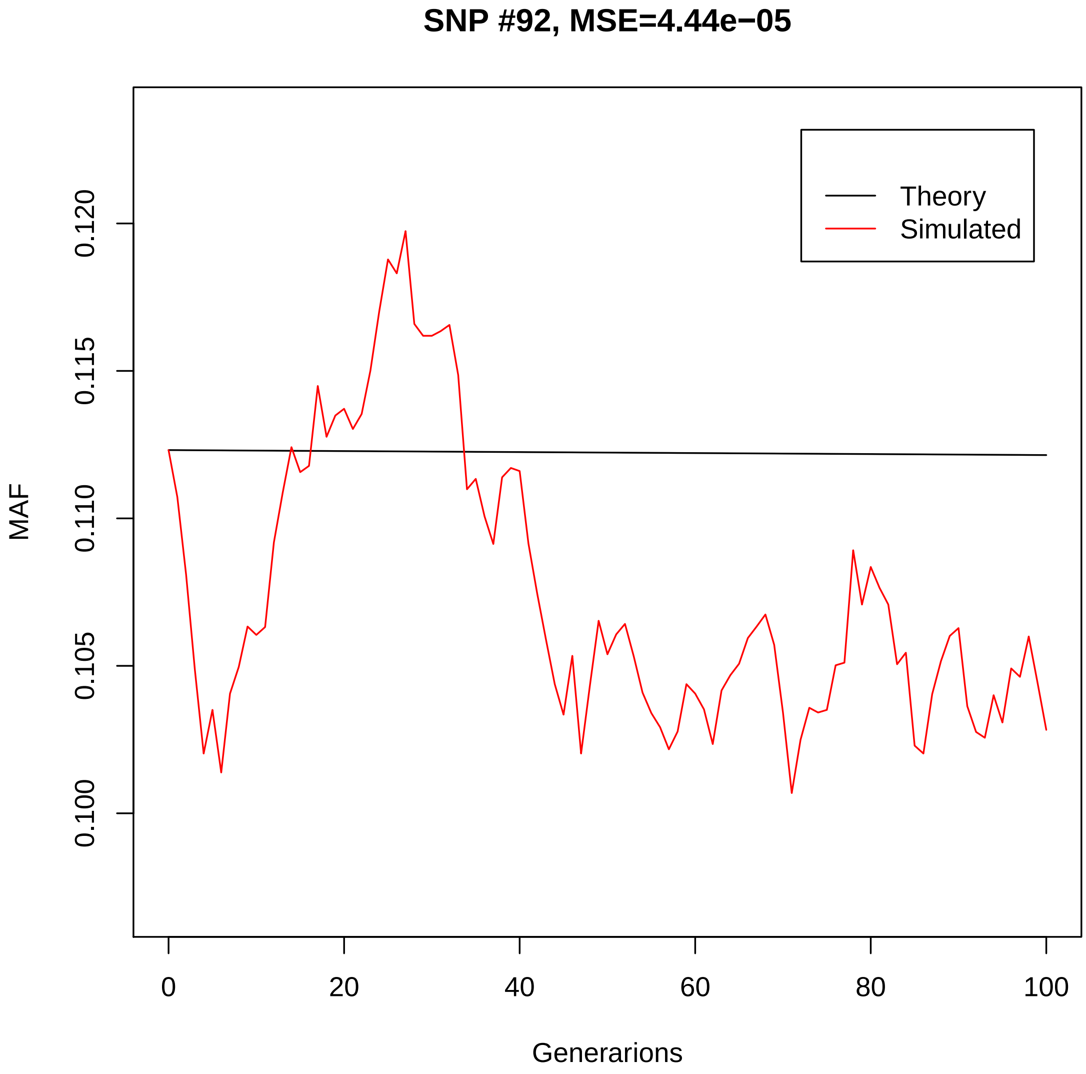


Figure 3.3: Comparing the the simulated heterozygosity (red) and the its expected value (black) obtained by theory per generation. The MSE is 4.44e-05

We then divided the pairwise estimated LDs uniformly in 20 groups with equal number of SNPs. Table 3.3 summarized some statistics. For each group, we computed the following quantities:

$$MSE = \frac{1}{N}\sum_{i\leq j}(LD_{ij}(30) - LD_{ij}(0))^2, \tag{3.4}$$

where N is the number of pairwise LDs in each group, and $LD_{ij}(30)$ and $LD_{ij}(0)$ are the LD between SNPs i and j at generation 30 and 0, respectively. The min absolute difference ($\min AD$) and max absolute difference ($\max AD$) are defined as

$$\min AD = \min_{i,j} |LD_{ij}(30) - LD_{ij}(0)|, \tag{3.5}$$

$$\max AD = \max_{i,j} |LD_{ij}(30) - LD_{ij}(0)|. \tag{3.6}$$

In the last column of Table 3.3, we computed the correlation between LDs at generation 30 and 0, i.e.,

$$\rho = \mathrm{corr}[LD_{ij}(30), LD_{ij}(0)]. \tag{3.7}$$

This table shows that there is no meaningful difference between the LD structure at generation 30 and 0, although there are small changes in the their values due to the stochastic nature of recombinations. The overall correlation is %99 and the maximum absolute difference between LDs is 0.28. Figure 3.4 plots the histogram of absolute difference of LDs at the initial and the last generations.

To check graphically, we also selected 39 SNPs around a randomly selected SNP and computed their pairwise LDs at the initial population and the last generated population (generation 30). The LD heatmap for generation 0 and generation 30 are plotted in Figures 3.5 and 3.6, respectively. As expected, there are a little changes in the colors.

3.5 Additive variance in assortative mating

To compare the estimated variance of the additive term under assortative mating with its theoretical value, we plot 3 simulation runs with population size of 30,000 in Figure 3.7 for spousal correlation of 0.4.

As can be seen, the estimated variance of the additive term under assortative mating is close to its theoretical value.

Table 3.3: Summary statistics for the differences in LD structure at the initial and the last generations (chromosome 1).

Group	N	MSE	$\min AD$	$\max AD$	ρ
1	128244	7.85e-05	2.13e-09	1.71e-01	0.9940
2	128244	8.92e-05	1.85e-09	2.05e-01	0.9940
3	128244	9.82e-05	2.36e-09	2.00e-01	0.9933
4	128244	9.08e-05	1.01e-10	2.43e-01	0.9936
5	128244	1.07e-04	7.30e-10	1.66e-01	0.9945
6	128244	8.54e-05	0.00e+00	1.75e-01	0.9938
7	128244	1.04e-04	2.27e-09	1.98e-01	0.9943
8	128244	9.40e-05	0.00e+00	1.65e-01	0.9936
9	128244	1.16e-04	5.00e-10	1.92e-01	0.9942
10	128244	1.03e-04	7.84e-10	2.52e-01	0.9937
11	128244	1.05e-04	8.80e-09	2.05e-01	0.9948
12	128244	9.98e-05	4.51e-10	1.81e-01	0.9937
13	128244	1.04e-04	0.00e+00	2.19e-01	0.9938
14	128244	1.14e-04	2.00e-10	2.22e-01	0.9940
15	128244	1.03e-04	1.56e-09	1.81e-01	0.9948
16	128244	8.05e-05	0.00e+00	1.78e-01	0.9940
17	128244	1.00e-04	6.00e-10	2.35e-01	0.9923
18	128244	1.13e-04	1.09e-09	2.82e-01	0.9931
19	128244	7.50e-05	2.00e-09	1.68e-01	0.9944
20	128244	8.15e-05	5.60e-09	1.51e-01	0.9937
All	2564892	9.71e-05	0.00e+00	2.82e-01	0.9939

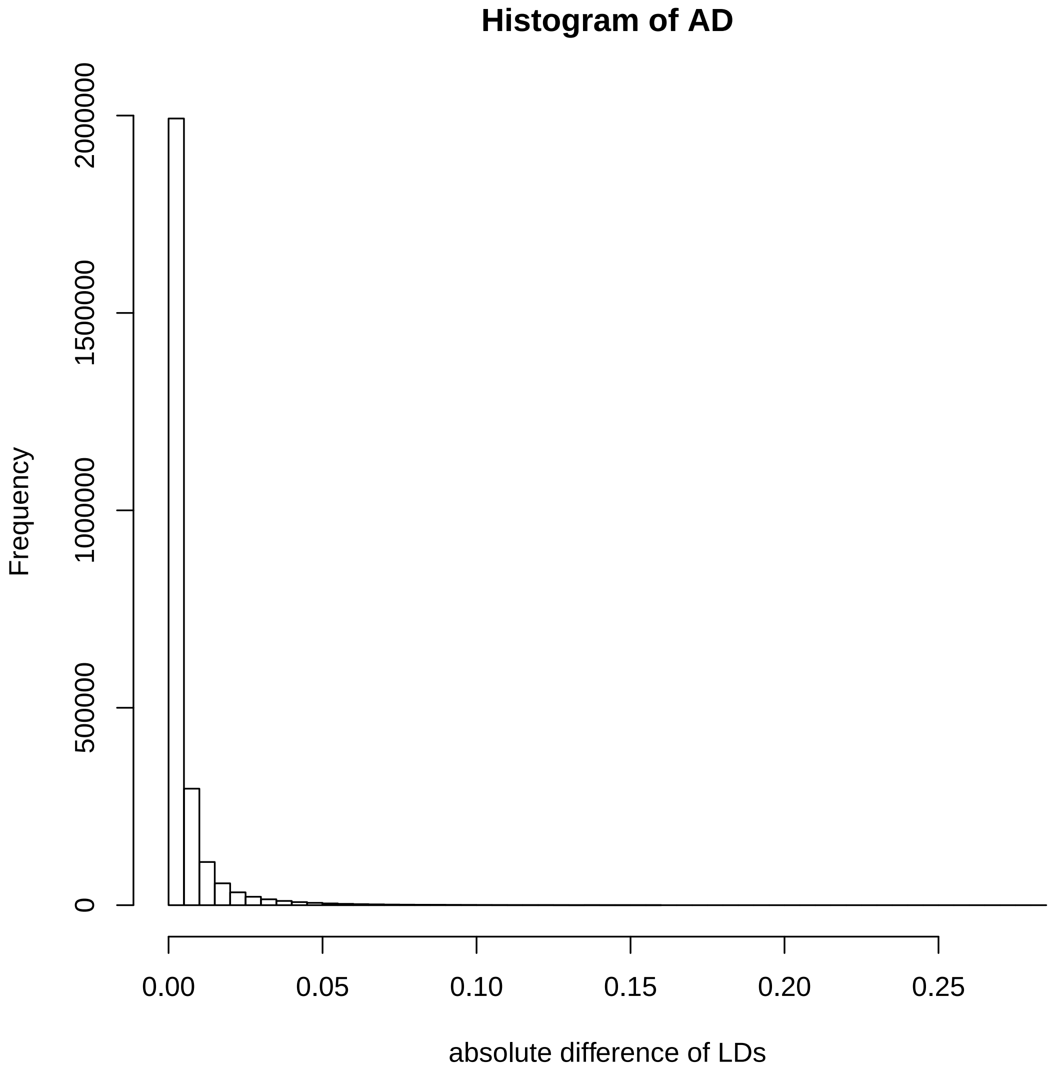


Figure 3.4: Histogram of the absolute difference of LDs.

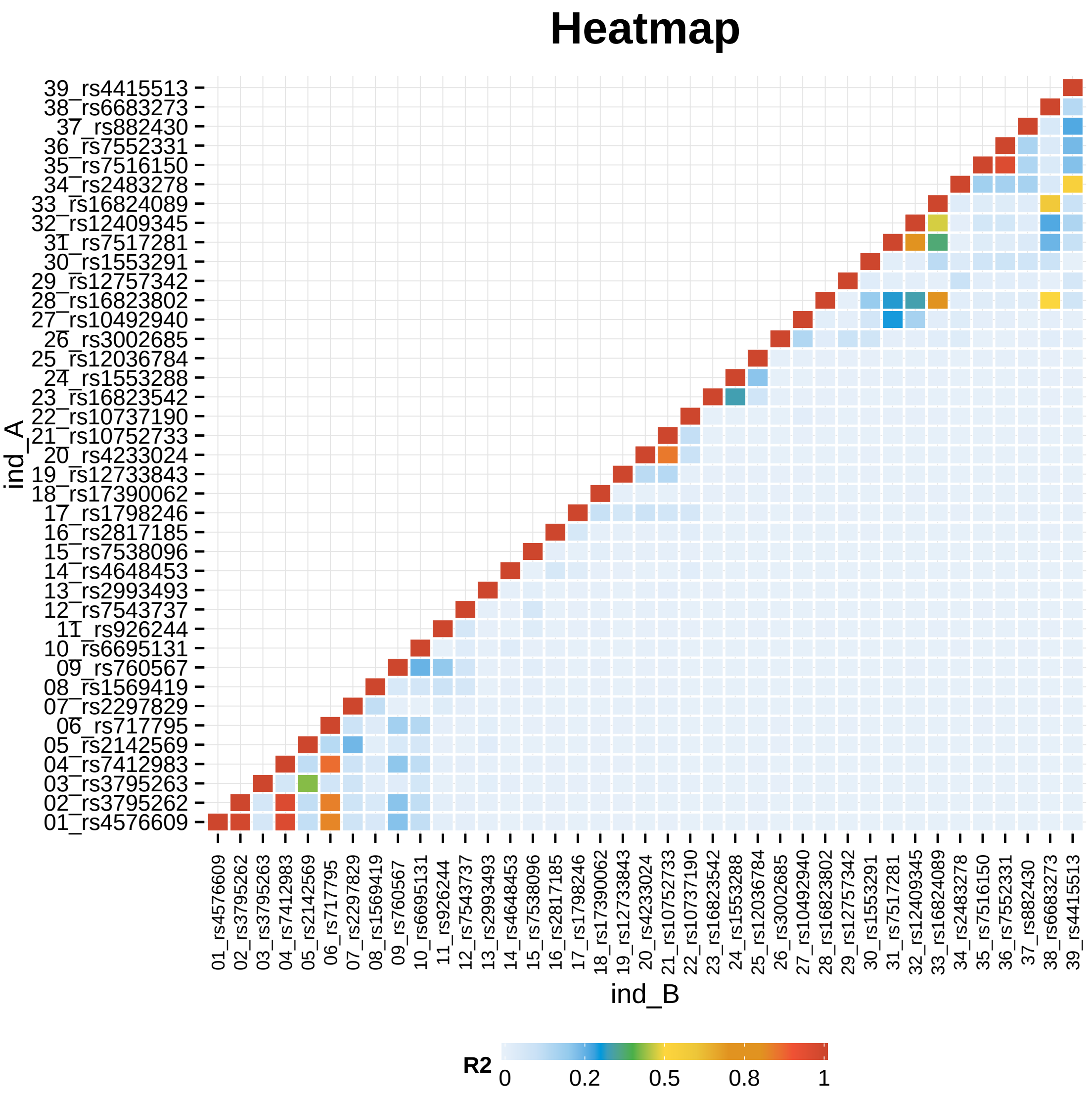


Figure 3.5: LD heatmap for the generation 0 (initial population).

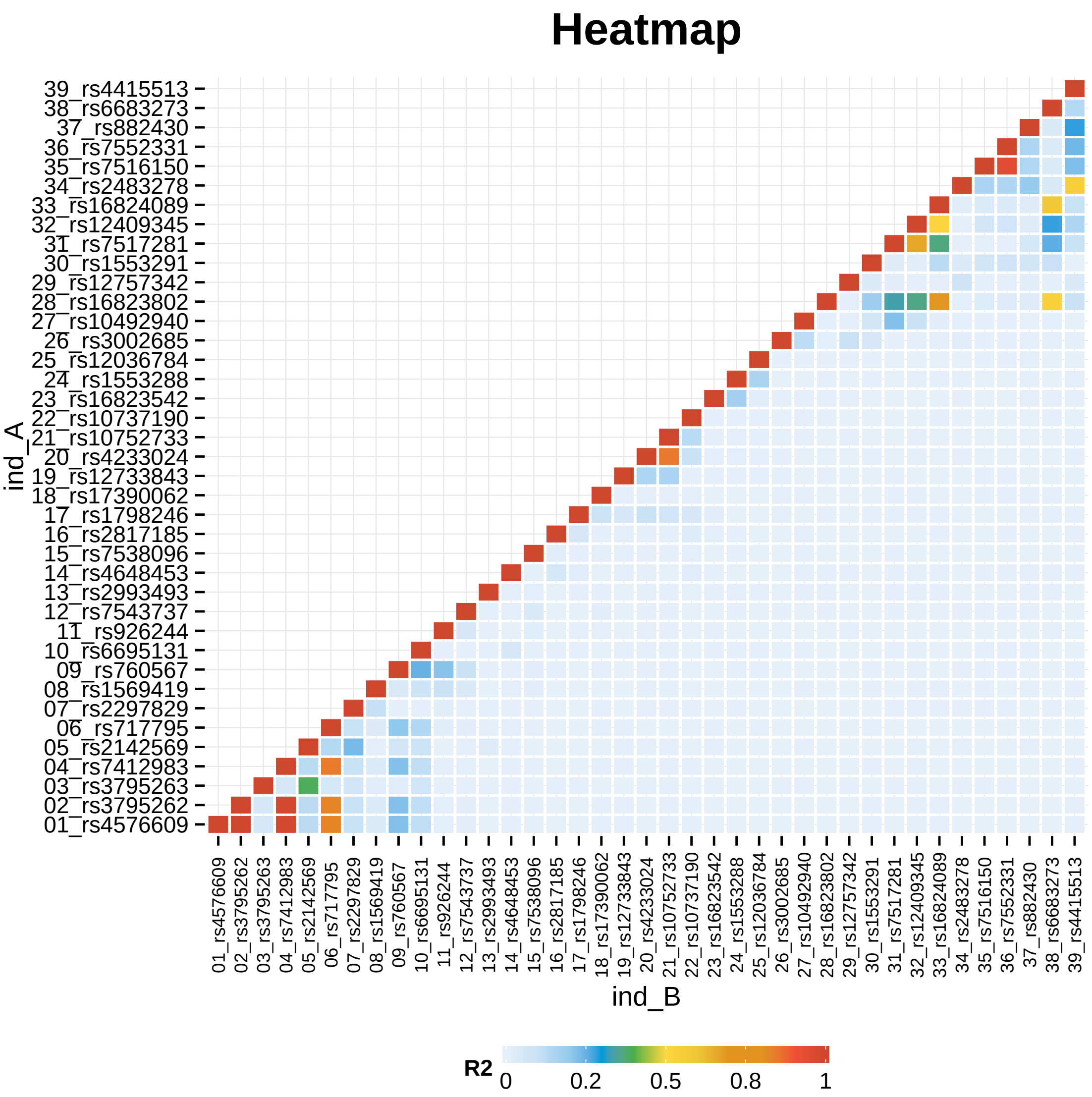


Figure 3.6: LD heatmap for the generation 30.

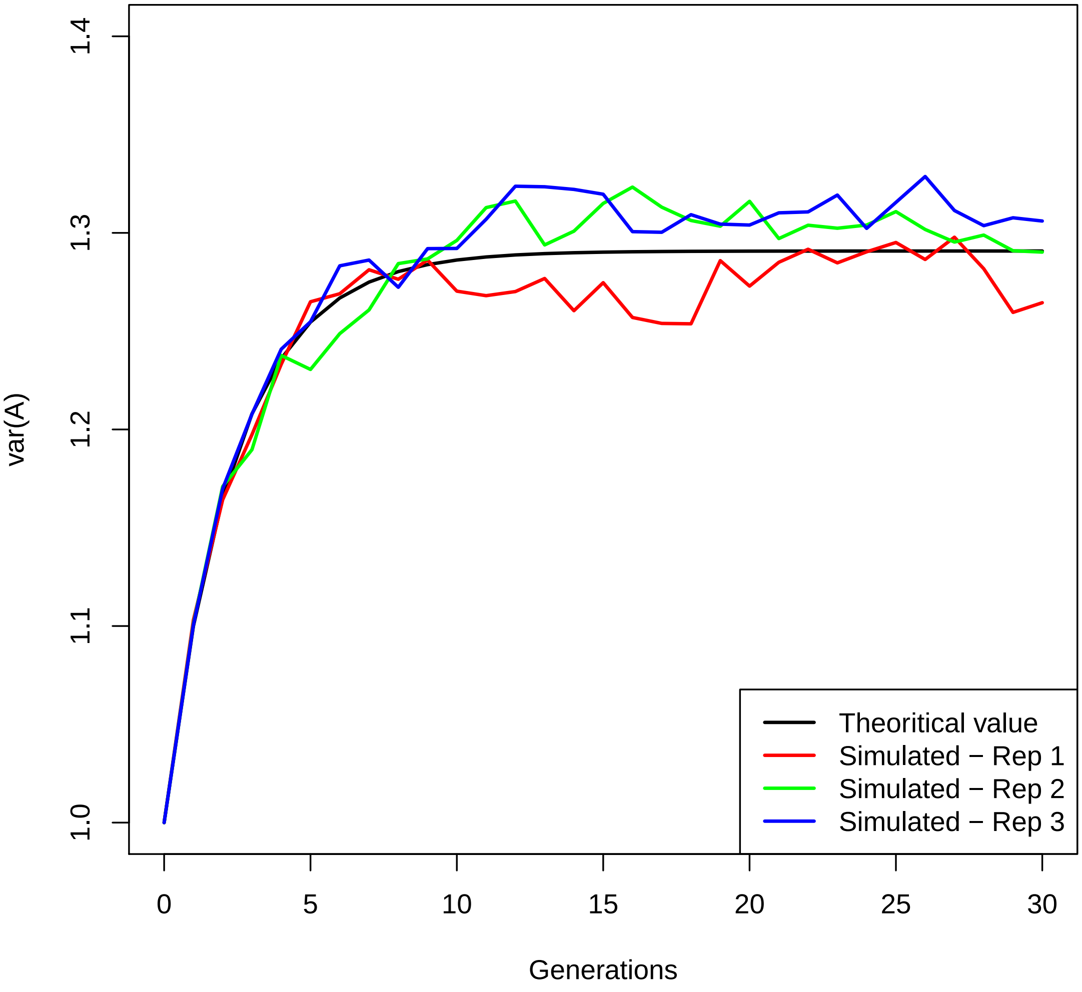


Figure 3.7: Comparing the simulated variance of additive term under assortative mating for three replications (red, green and blue) with its theoretical value (black line).

Chapter 4

Parameters 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
```

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

4.1 The order of the parameters

For simulating one population and one phenotype, the order of the parameters do not matter. So for example, you can put `--va` prior to the `--file-cv_info`. But, with multiple phenotypes and one populations, *GeneEvolve* sets the first parameter for the first phenotype and the second parameter for the second phenotype and so on. For example, if you use `--va 3 --va 2`, then the additive variances for the first and second phenotypes will scaled to 3 and 2, respectively. For multiple phenotypes and multiple populations, *GeneEvolve* considers all the parameters prior to the `--next_population` as a set of parameters for the first population and treat it as one population with multiple phenotypes. The parameters after `--next_population` (and possibly before the second `--next_population`) will be considered for the second population and so on.

Note that the number of phenotypes should be the same for each population.

4.2 Parameters

4.2.1 `--file_gen_info`

This is a required parameter with no default value. You should use it for each population.

```
--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.

If you want to simulate p populations ($p > 1$), then you need p of this file (possibly with different parameters) for each population with the same number of lines (generations), i.e., the number of generations should be the same for all the populations.

4.2.2 `--file_hap_name`

This is a required parameter with no default value. You should use it for each population. This file addresses hap, legend and sample files for the reference panel. The refrence panel can be SNP or sequence data. Since the SNP/sequence reference panels can be huge, so they should be separated for each chromosome.

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

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

If you want to simulate p populations ($p > 1$), then you should use this parameter p times for each population. You can use different initial hap files for for each population.

4.2.3 `--file_recom_map`

This is a required parameter with no default value. You should use it for each population.

```
--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.

If you want to simulate p populations ($p > 1$), then you should use this parameter p times for each population. You can use different recombination map files for for each population.

4.2.4 `--file_cv_info`

This is a required parameter with no default value. You should use it for each phenotype.

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

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

Clearly, If you want to simulate m phenotypes ($m > 1$), then you should use this parameter m times for each phenotype with different sets of CVs. Importantly, you can control the correlation between phenotypes by choosing overlap sets of CVs.

4.2.5 `--file_cvs`

This is a required parameter with no default value. You should use it for each phenotype.

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

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

Clearly, If you want to simulate m phenotypes ($m > 1$), then you should use this parameter m times for each phenotype with different sets of CVs.

4.2.6 `--va [-1]`

This is an optional parameter with -1 as the default value, where -1 means its value will be computed based on the user-specified additive and dominance effects and will not be scaled. We know that variance is not negative, but this value is useful when you want to simulate three phenotypes and you want to scale just the first and third one. For example `--va 3 --va -1 --va 2` means the the additive variance of the first and second phenotype should be scaled to 3 and 2, respectively, while the second variance will not scaled. You can use it for each phenotype.

```
--va 2
```

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

4.2.7 `--vd [-1]`

This is an optional parameter with -1 as the default value, where -1 means its value will be computed based on the user-specified additive and dominance effects and will not be scaled. We know that variance is not negative, but this value is useful when you want to simulate three phenotypes and you want to scale just the first and third one. For example `--vd 3 --vd -1 --vd 2` means the the dominance variance of the first and second phenotype should be scaled to 3 and 2, respectively, while the second variance will not scaled. You can use it for each phenotype.

```
--vd 0.1
```

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

4.2.8 `--vc [0]`

This is an optional parameter with 0 as the default value. You can use it for each phenotype.

```
--vc 0.1
```

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

4.2.9 `--ve [1]`

This is an optional parameter with 1 as the default value. You can use it for each phenotype.

```
--ve 1
```

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

4.2.10 `--vf [0]`

This is an optional parameter with 0 as the default value. You can use it for each phenotype.

```
--vf 0.1
```

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

4.2.11 `--RM`

This is an optional parameter with no default value.

```
--RM
```

There are two mating systems in *GeneEvolve*: random mating and assortative mating (mating based on the phenotype). Assortative mating is the default random system for *GeneEvolve*, where the spousal correlation is specified in the `--file_gen_info` file. User can choose random mating system using `--RM`. In this case the spousal correlation specified in the `--file_gen_info` file and the parameter `avoid_inbreeding` will be ignored.

4.2.12 `--avoid_inbreeding`

This is an optional parameter. If you use it, then *GeneEvolve* avoids inbreeding (avoids sibs and cousin mating).

```
--avoid_inbreeding
```

This parameter will be ignored if you use the random mating system by `--RM`.

4.2.13 `--next_population`

This is a required parameter when you want to simulate more than one population.

```
--next_population
```

This parameter will used to give the next population information. Clearly, you should use it $p - 1$ times, if you want to simulate p populations, since we do not use it for the first population.

4.2.14 `--omega [1]`

This is an optional parameter with 1 as the default value. It is the coefficient for mating value (see Figure 2.6). You can use it for each phenotype.

```
--omega .3
```

4.2.15 `--beta [1]`

This is an optional parameter with 1 as the default value. It is the coefficient for selection value (see Figure 2.6). You can use it for each phenotype.

```
--beta .5
```

4.2.16 `--lambda [1]`

This is an optional parameter with 1 as the default value. It is the coefficient for familial effect (see Figure 2.6). You can use it for each phenotype.

```
--lambda .7
```

4.2.17 `--gamma [0]`

This is an optional parameter with 0 as the default value. It is the environmental effects specific to each population (see Section 2.9). If you set it to zero or not use it, then *GeneEvolve* will not generate population specific environmental effects.

```
--gamma 1
```

4.2.18 `--file_migration`

This is a required parameter if you simulate more than one populations.

```
--file_migration par.migration.txt
```

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

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

where m_{ij} is the probability of migrating from population i to j. By this format, a user can use different migration matrix per generation. For example in the following listing file, the probability of migrating for 10 generations and 2 populations are listed.

```
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

As and example, for generation 3, the migration matrix is

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

4.2.19 `--format_output [hap]`

This is an optional parameter with "hap" as the default value. This parameter can be "hap" or "plink" and determines the output file format for genotypes.

```
--format_output plink
```

For more information about hap and plink file formats, see the appendix C.

4.2.20 `--interval`

This is an optional parameter with no default value. If you use it, the program makes another output for genotypes in ”interval” format.

```
--interval
```

The ”interval” output format is the true identity by descent information for haplotypes. For more information about interval file format, see the appendix C.

4.2.21 `--no_output`

This is an optional parameter with computer specified default value. You can use it to save disk space.

```
--no_output
```

If you use this parameter, then the output genotype files will not created. Note that *GeneEvolve* always reports the summary and population information at each generation.

4.2.22 `--output_all_generations`

This is an optional parameter with no default value.

```
--output_all_generations
```

Since the size of genotype files are big, so *GeneEvolve* outputs the genotypes of the last generation, by default. You can save the genotypes for all the generations using this parameter.

4.2.23 `--prefix [out]`

This is an optional parameter with ”out” as the default value. This parameter determines the prefix for the output files.

```
--prefix example1
```

Acknowledgements

For the base population used to simulate Figure S2 and Table S2, we used a sample of 33,253 individuals from 9 datasets in the NCBI dbGaP Database of Genotypes and Phenotypes: the Health and Retirement Study (dbGaP accession phs000428.v1.p1), the Genetic Epidemiology of COPD (dbGaP accession phs000179.v4.p2), the Chronic Renal Insufficiency Cohort Study (dbGaP accession phs000524.v1.p1), the Geisinger eMERGE Abdominal Aortic Aneurysm Project (dbGaP accession phs000387.v1.p1), the Geisinger eMERGE MyCode Project (dbGaP accession phs000381.v1.p1), the Genes and Blood Clotting Study (dbGaP accession phs000304.v1.p1), the Polycystic Ovary Syndrome (PCOS) Genetics study (dbGaP accession phs000368.v1.p1), the WHI GARNET study (dbGaP accession phs000315.v6.p3, and the WHIMS+ study (dbGaP accession phs000675.v2.p3) from the NIH Genotype and Phenotype database.

For the base population used to simulate the last row of Table S2, we used the UK10K project sequence data with a sample of 3,781 individuals (with accession numbers EGAD00001000776).

Appendix A

Genotype simulation for an initial population

To run *GeneEvolve*, we first need genotype data. Here, is a simple R cod for simulating genotypes for 3 chromosomes. Clearly, the structure of real genotypes are more complex than this simulated data, but this simple data is useful for educational purpose.

```
##################################################
# 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 <- 2000
VAR.A <- 1
VAR.D <- .1
VAR.A2 <- 1 # for the second CV set
VAR.D2 <- .3 # for the second CV set
INCLUDE.CHRS <- 1:NCHR

# We create this map in order to use the real genomic map distance
# genotypes and cvs should be in range of genomic map
map.pos <- matrix(ncol=2,nrow=22)
map.pos[ 1 ,1] <-  738555
map.pos[ 2 ,1] <-  1
map.pos[ 3 ,1] <-  1
map.pos[ 4 ,1] <-  1
map.pos[ 5 ,1] <-  1
map.pos[ 6 ,1] <-  105878
map.pos[ 7 ,1] <-  567276
map.pos[ 8 ,1] <-  64984
map.pos[ 9 ,1] <-  88894
map.pos[ 10 ,1] <-  26070
map.pos[ 11 ,1] <-  102856
map.pos[ 12 ,1] <-  91619
map.pos[ 13 ,1] <-  19198564
map.pos[ 14 ,1] <-  20326742
map.pos[ 15 ,1] <-  22684095
map.pos[ 16 ,1] <−  1263
map.pos[ 17 ,1] <−  1
map.pos[ 18 ,1] <−  12535
map.pos[ 19 ,1] <−  160912
map.pos[ 20 ,1] <−  1
map.pos[ 21 ,1] <−  15107860
map.pos[ 22 ,1] <−  17096300

map.pos[ 1 ,2] <−  249238555
map.pos[ 2 ,2] <−  242900000
map.pos[ 3 ,2] <−  197900000
map.pos[ 4 ,2] <−  1.91e+08
map.pos[ 5 ,2] <−  180750000
map.pos[ 6 ,2] <−  170955878
map.pos[ 7 ,2] <−  159167276
map.pos[ 8 ,2] <−  146364984
map.pos[ 9 ,2] <−  141088894
map.pos[ 10 ,2] <−  135526070
map.pos[ 11 ,2] <−  135002856
map.pos[ 12 ,2] <−  133841619
map.pos[ 13 ,2] <−  115148564
map.pos[ 14 ,2] <−  105826742
map.pos[ 15 ,2] <−  102484095
map.pos[ 16 ,2] <−  90201263
map.pos[ 17 ,2] <−  81100000
map.pos[ 18 ,2] <−  78062535
map.pos[ 19 ,2] <−  59160912
map.pos[ 20 ,2] <−  6.3e+07
map.pos[ 21 ,2] <−  48157860
map.pos[ 22 ,2] <−  51246300



NCHR <− length(NSNP)
cv.info <− c()
cv2.info <− c()

for (ichr in 1:NCHR)
{
   print(”−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−”)
   nsnp_chr <− NSNP[ichr]
   p <− runif(nsnp_chr,min=.05,max=.95)
   hap <− matrix(0, nrow=nsnp_chr, ncol=2∗NIND)
   for (isnp in 1:nsnp_chr)
   {
      hap[isnp,] <− rbinom(2∗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,1]:map.pos[ichr,2], 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=c(”id”,”pos”,”al0”,”al1”))
    # 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 set 1
    cv_chr <− sort(sample(1:nsnp_chr, ncv_chr, replace = FALSE))
    cvs <− hap[cv_chr,]
    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)
    # cv set 2
    cv2_chr <− sort(sample(1:nsnp_chr, ncv_chr, replace = FALSE))
    cvs2 <− hap[cv2_chr,]
    write.table(cvs2, file=paste(”cv2.chr”,ichr,”.hap”,sep=’’), quote=FALSE,row.names=FALSE, col.names=FALSE)
    cv2.pos <− legend.pos[cv2_chr]
    cv2.maf <− apply(matrix(c(cvs2),nrow=2∗NIND),2,mean)
    cv2.maf[which(cv2.maf>.5)] <− 1−cv2.maf[which(cv2.maf>.5)]
    cv2.a <− rnorm(ncv_chr,mean=0,sd=sqrt(VAR.A2))
    cv2.d <− rnorm(ncv_chr,mean=0,sd=sqrt(VAR.D2))
    cv2.info.chr <− cbind(ichr,cv2.pos,cv2.maf,cv2.a,cv2.d)
    cv2.info <− rbind(cv2.info,cv2.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)

colnames(cv2.info) <− c(”chr”,”pos”,”maf”,”a”,”d”)
write.table(cv2.info, file=paste(”cv2.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)

b <− paste(”cv2.chr”,INCLUDE.CHRS,”.hap”,sep=’’)
b <− cbind(INCLUDE.CHRS,b)
write.table(b,file=paste(”par.pop1.cv2_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”))
```

create.genotypes.R

To create population information file, you can also use the following code:

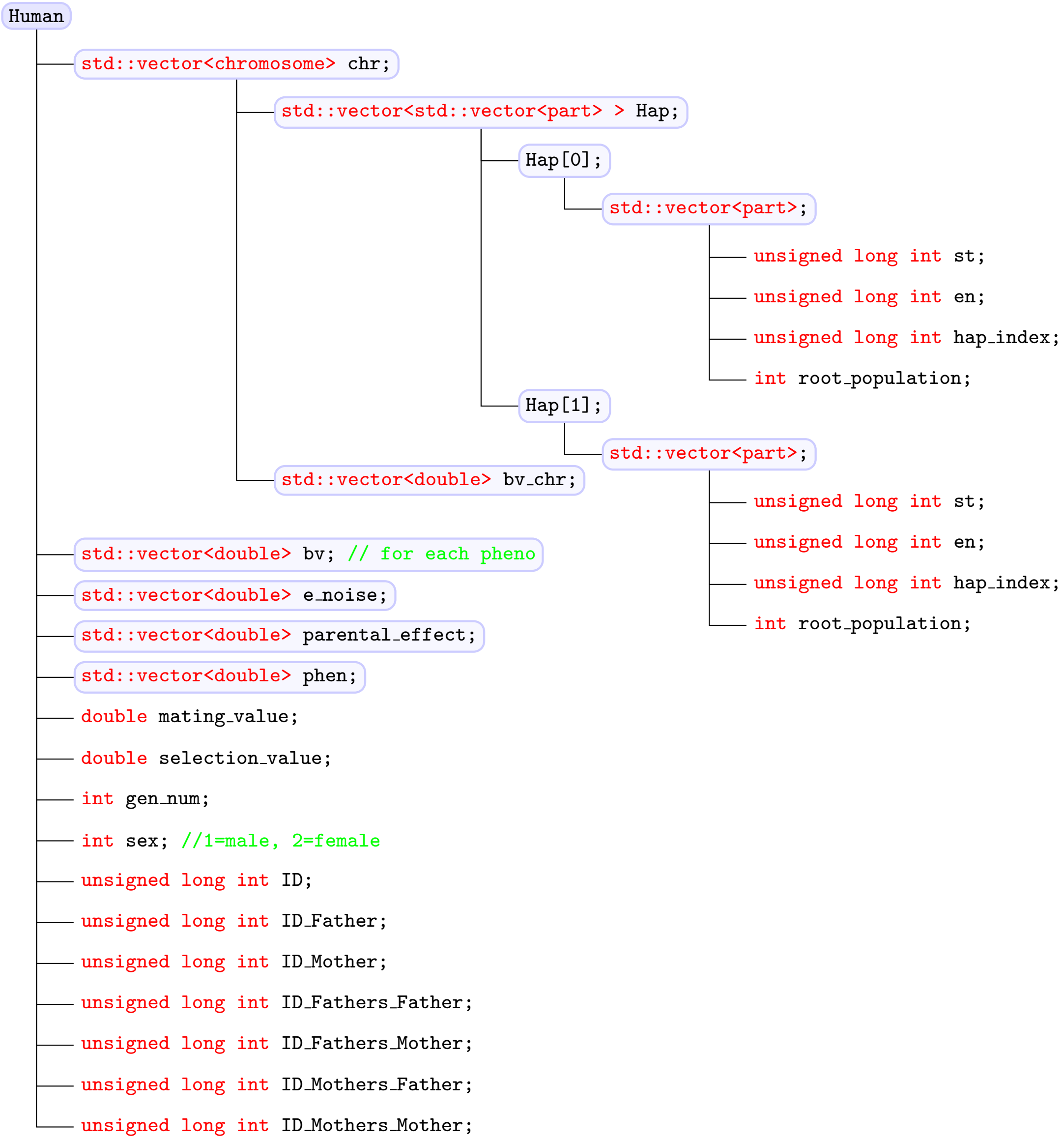
```
# creating population info

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

NUM.GENERATIONS <− 10    #number of generations
POP.SIZE <− rep(3000,NUM.GENERATIONS)  #population size over time
PHENO.MATE.COR <− rep(0,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(”thr”,NUM.GENERATIONS) #Selection function for each generation
SELECTION.FUNCTION.PAR1 <− rep(1,NUM.GENERATIONS) #Selection function for each generation
SELECTION.FUNCTION.PAR2 <− rep(1,NUM.GENERATIONS) #Selection function for each generation

##########popinfo.txt
write.table(cbind(POP.SIZE,PHENO.MATE.COR,OFFSPRING_DIST,SELECTION.FUNCTION,SELECTION.FUNCTION.PAR1,SELECTION.FUNCTION.PAR2), file=paste(”ex1.popinfo.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

Appendix B

C++ Classes

B.1 Human



```
Human
    std::vector<chromosome> chr;
        std::vector<std::vector<part> > Hap;
            Hap[0];
                std::vector<part>;
                    unsigned long int st;
                    unsigned long int en;
                    unsigned long int hap_index;
                    int root_population;
            Hap[1];
                std::vector<part>;
                    unsigned long int st;
                    unsigned long int en;
                    unsigned long int hap_index;
                    int root_population;
        std::vector<double> bv_chr;
    std::vector<double> bv; // for each pheno
    std::vector<double> e_noise;
    std::vector<double> parental_effect;
    std::vector<double> phen;
    double mating_value;
    double selection_value;
    int gen_num;
    int sex; //1=male, 2=female
    unsigned long int ID;
    unsigned long int ID_Father;
    unsigned long int ID_Mother;
    unsigned long int ID_Fathers_Father;
    unsigned long int ID_Fathers_Mother;
    unsigned long int ID_Mothers_Father;
    unsigned long int ID_Mothers_Mother;
```

B.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
```

B.3 Simulation

```
Simulation
    ├── int _n_pop;
    ├── int _tot_gen;
    ├── bool _debug;
    ├── std::vector<Population> population;
    ├── Parameters par;
    ├── std::vector<std::vector<double> > imigration_mat_gen;
    ├── std::string _out_prefix;
    ├── std::string _format_output;
    ├── bool _output_all_generations;
    ├── std::vector<int> _all_active_chrs;
    ├── std::vector<double> _gamma; // for each phenotype and all the populations
    └── std::vector<Pop_phen_info> Pop_info_prev_gen; // Saving mating_value for the next generation
                                          ├── std::vector<double> mating_value; // for each ind
                                          ├── std::vector<double> selection_value; // for each ind
                                          └── std::vector<std::vector<double> > phen; // for each phen and ind
```

Appendix C

File Formats

C.1 Hap – Legend – Sample

C.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

C.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

C.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

C.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

C.2 PLINK

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

C.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)

C.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

C.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

C.3 Interval

Has header.

Each line corresponds to a single part of a haplotype. The columns are:

1. Human ID (zero based index)

2. Chromosome number
3. Haplotype (0 or 1)
4. Start position (bp)
5. End position (bp)
6. Index position of the haplotype in the initial generation (generation 0)
7. Root population index (zero based index)

In the following example, the first haplotype (haplotype 0) for chromosome 1 of an individual with ID 0, is saved as a continues union of intervals, where each interval comes from one ancestral root. The id of the ancestral root is given in the sixth column.

```
h_ID chr hap st en hap_index root_pop
0 1 0 738555 3981099 28927 0
0 1 0 3981099 4804034 16396 0
0 1 0 4804034 5046412 47008 0
0 1 0 5046412 5917830 19541 0
0 1 0 5917830 6241920 52672 0
0 1 0 6241920 7781935 1026 0
0 1 0 7781935 10744488 17586 0
0 1 0 10744488 14786766 15382 0
0 1 0 14786766 14932265 13305 0
0 1 0 14932265 30071798 62082 0
0 1 0 30071798 30565233 34178 0
0 1 0 30565233 35873811 15383 0
0 1 0 35873811 37925508 22152 0
0 1 0 37925508 41381101 14645 0
0 1 0 41381101 45089312 28882 0
0 1 0 45089312 57225089 14645 0
0 1 0 57225089 60927267 7988 0
0 1 0 60927267 65383367 41180 0
0 1 0 65383367 67645596 34866 0
0 1 0 67645596 94508106 49280 0
0 1 0 94508106 97900471 15372 0
0 1 0 97900471 101722348 22732 0
...
```

out1.pop1.gen20.chr1.int

Given a ".int" file as input, the program "SharedHaplotypes" available at
https://github.com/rtahmasbi/IBG-SharedHaplotypes
can extract the true shared haplotypes between each two individuals and can report the true identical by decent (IBD) haplotypes.

Bibliography

[1] Andy Boyd, Jean Golding, John Macleod, Debbie A Lawlor, Abigail Fraser, John Henderson, Lynn Molloy, Andy Ness, Susan Ring, and George Davey Smith. Cohort profile: the 'children of the 90s' – the index offspring of the avon longitudinal study of parents and children. *International journal of epidemiology*, 42(1):111–27, 2013.

[2] DS Falconer and TFC Mackay. *Introduction to Quantitative Genetics*. Longman, 4 edition, 1996.

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

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

[5] 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.

[6] Matthew D Mailman, Michael Feolo, Yumi Jin, Masato Kimura, Kimberly Tryka, Rinat Bagoutdinov, Luning Hao, Anne Kiang, Justin Paschall, Lon Phan, et al. The ncbi dbgap database of genotypes and phenotypes. *Nature genetics*, 39(10):1181–1186, 2007.

[7] Alireza Moayyeri, Christopher J Hammond, Deborah J Hart, and Timothy D Spector. The uk adult twin registry (twinsuk resource). *Twin Research and Human Genetics*, 16(01):144–149, 2013.

[8] Bo Peng and Marek Kimmel. simupop: a forward-time population genetics simulation environment. *Bioinformatics*, 21(18):3686–3687, 2005.

[9] Kimberly A Tryka, Luning Hao, Anne Sturcke, Yumi Jin, Zhen Y Wang, Lora Ziyabari, Moira Lee, Natalia Popova, Nataliya Sharopova, Masato Kimura, et al. Ncbi?s database of genotypes and phenotypes: dbgap. *Nucleic acids research*, 42(D1):D975–D979, 2014.

[10] 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.