User Manual for *gs*2.0

Jing Li, Yixuan Chen
Electrical Engineering and Computer Science Department
Case Western Reserve University, Cleveland, OH 44106, USA
jingli@eecs.case.edu

March 2010

Citation:

(i) J Li and Y Chen. Generating Samples for Association Studies Based on HapMap data. BMC Bioinformatics 9:44, 2008.

(ii) Y Chen and J Li. *gs*2.0: a Simulation Tool for Generating Samples Involving Genetic and Environmental Interactions for Association Studies. *Submitted*.

Contents

Chapter 1

Introduction

1.1 Overview

The program *gs* can quickly generate a large number of samples based on real data that are useful for a variety of purposes, including evaluating methods for haplotype inference, tag SNP selection and association studies. Two approaches have been implemented to generate dense SNP haplotype/genotype data that share similar local *linkage disequilibrium* (LD) patterns as those in human populations. The first approach takes haplotype pairs from samples as inputs, and the second approach takes patterns of haplotype block structures as inputs. Both quantitative and qualitative traits have been incorporated in the program. Phenotypes are generated based on a disease model, or based on the effect of a quantitative trait nucleotide, both of which can be specified by users. In addition to single-locus disease models, two-locus disease models have also been implemented that can incorporate any degree of epistasis. Users are allowed to specify all nine parameters in a 3×3 penetrance table. For several commonly used two-locus disease models, the program can automatically calculate penetrances based on the population prevalence and marginal effects of a disease that users can conveniently specify.

We extend *gs* to generate samples involving genetic and environmental interactions. The new version of the program *gs*2.0 provides great functionalities and flexibilities to simulate various interaction models. By specifying the disease model and the parameters, one can use this program to simulate desired dataset for association studies on complex disease. Data generated by *gs*2.0 can serve as a common ground to compare different approaches in detecting interactions.

1.2 Algorithms

The program has implemented two generating models for haplotypes/genotypes, *i.e.*, extension method and block method. Both methods can be used to generate qualitative and quantitative phenotype data. We first

introduce the two methods in the context of generating case-control data, and the extension to quantitative traits will then be discussed.

1.2.1 Extension Method

The first model is an extension to the one used in [2] that takes phased haplotype pairs as its inputs. For example, one can use the haplotype results from the HapMap project as inputs, which can be downloaded from the HapMap website. Users first create a disease model by specifying the disease allele frequency (DAF) and the penetrance of each genotype. Alternatively, users can define a disease model using the population prevalence and genotype relative risks. A simple relationship exists between penetrance parameters and genotype relative risk parameters [6]. Therefore, in the following we discuss our procedure only using one set of parameters, the penetrance. The program first picks a SNP t from the input data, where one of its two alleles has the frequency approximately equal to the DAF specified in the parameter file. This allele is regarded as the high-risk variant. (Alternatively, users can specify a particular SNP as the disease susceptibility locus.) To generate the genotype at the disease locus for a case, it first calculates the conditional probability of each genotype (homozygous wild, heterozygous, and homozygous mutant) given the individual being affected based on equation 1.2.1.

$$Pr(g_i|case) = \frac{Pr(g_i)Pr(case|g_i)}{\sum_j Pr(g_j)Pr(case|g_j)} \tag{1.2.1}$$

The actual genotype g will then be selected based on the conditional distribution. The genotype frequencies in the above formula can be obtained from allele frequencies under the assumption of Hardy-Weinberg equilibrium. The probability of being affected given a particular genotype (penetrance) is given by users as a parameter. To generate the haplotype pairs h_1 and h_2 for this affected individual, the program randomly selects two haplotypes h_3 and h_4 from the inputs with the genotype at the disease locus t as required (*i.e.*, $(h_3[t], h_4[t]) = g$, where $h_i[t]$ denotes the allele at the t^{th} locus on haplotype h_i). In their original paper [2], haplotype h_1 will be given the same alleles as h_3 from locus $t-l$ to $t+l$, where l is a parameter that can be specified by users. To extend h_1 to the right for one more locus, it randomly selects another haplotype h_5 that has the same alleles as h_1 from locus $t-l+1$ to locus $t+l$, and let $h_1[t+l+1] = h_5[t+l+1]$. By iterating the above process, one can extend h_1 to the right and then to the left. We found that LD patterns from samples generated this way greatly depend on the parameter l (data not shown). Even for one particular data set, the extend of LD may vary substantially in different segments. A single l can not accommodate all the cases. We have extended the above method by introducing two parameters, l_{min} and l_{max}. The overlapped length for both the initial assignment and the extension of h_1 will be stochastically determined by two values l_l and l_r ($l_{min} \leq l_l, l_r \leq l_{max}$), one for each direction. The values of l_l and l_r depend upon the strength of local LD. More specifically, l_r is initialized as l_{min}. The value of l_r is increased by 1 if the LD measure D' between locus $t+l_r$ and locus $t+l_r+1$ is greater than a uniformly distributed random number between 0 and 1. The process will terminate when the value of D' is smaller than such a random

number or when $l_r = l_{max}$. The value of l_l can be determined similarly to the left. The haplotype h_1 will be given the same alleles as h_3 from locus $t - l_l$ to $t + l_r$ initially. At each extension step, the procedure is the same as in the original paper [2], but with differences in determining the length of the overlapped region. For example, to extend to the right for one more locus, suppose the current locus (the right most one) is t_1. The leftmost locus t_2 of the overlapped region is stochastically determined based on pairwise LD with the constraint that $l_{min} \leq t_1 - t_2 \leq l_{max}$. A haplotype that shares the same segment with h_1 from t_2 to t_1 will be randomly selected and its allele at $t_1 + 1$ will be copied to h_1. A detailed description of the above procedure can be found in the manual of the program. The haplotype h_2 can be obtained similarly. The required number of cases can be generated by repeating the process. One can generate normal individuals using the same approach based on the genotype distribution conditional on the fact that the individuals are normal. By using two parameters, the method takes both long-range LD (up to l_{max}) and short-range LD into considerations.

1.2.2 Block Method

The second generating model is actually a Markov model based on haplotype block structures inferred from real data such as HapMap data. LD patterns such as a block-like structure have been commonly observed from experimental data for dense SNPs. Instead of directly using haplotype pairs, one can also take the haplotype block structures as inputs. As a Markov chain, each block is a state that consists of several common haplotypes controlled by an emission distribution. The connections of haplotypes between adjacent blocks are specified by a transition probability matrix. More specifically, for each block, the input data consist of the number of markers, the common haplotypes with their population frequencies, and the probabilities of each common haplotype connecting the common haplotypes in the next block. An example is given in Fig. 1.1. Such a structure can be inferred based on real data using some software such as Haploview [1]. To generate samples, users first specify a disease model (DAF and penetrances/relative risks) and the program selects a locus with its allele frequency approximately equal to the DAF. Users can also specify a disease locus directly. The genotype of a case (or a control) at the disease locus is generated in the same way as the extension method does. For each allele at the disease locus, a common haplotype with the allele embedded will be selected according to their frequency distribution. The two haplotypes will then be extended independently to both directions based on the transition probabilities. A large number of samples can be generated that will share similar LD patterns with real data but with different haplotypes and genotypes. To maintain a proper level of variety, we have considered SNPs that are not in any blocks, as well as possible rare haplotypes that do not exist from the input block file. For many block definitions, not all SNPs have to be within some blocks. To incorporate those missed SNPs, the original genotype data file that is used in generating the block structure has to be provided to the program. When generating a haplotype of a sample, the program imputes the missed SNPs sequentially based on their physical positions. For each position, an allele is chosen based on the allele in the previous position, and their frequencies and the pairwise LD between them estimated from the genotype data. Furthermore, rare haplotypes are usually

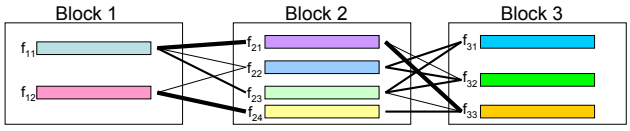

Figure 1.1: Haplotype block structure as a Markov Model. Each block is a state and each small rectangle within a block represents a common haplotype with its frequency denoted as f_{ij}. The transition probabilities between adjacent blocks are depicted by lines with width representing quantity.

been dropped in the Markov model. Only major haplotypes and their frequencies are available for each block. To incorporate rare haplotypes, we stochastically generates (only) one "rare" haplotype each time when a block is selected. The alleles of the haplotype are sampled solely based on the allele frequencies and pairwise LD. The frequency of the rare haplotype is defined so that the summation of haplotype frequencies within each block will be one (the summation of frequencies from common haplotypes only is often less than one from an input). The transition probabilities from the rare haplotype to haplotypes in the next block are proportional to the haplotype frequencies in the next block. When a block is selected again in generating another sample, a new rare haplotype will be generated and its frequency will be determined in a similar fashion. In such a way, every possible haplotype within a block will have a chance being selected as a rare haplotype for some samples. These new haplotypes will be rare overall in the samples because each time a different one might be selected. The procedure is designed so that it is slightly biased to haplotypes with common alleles.

For both methods, the disease locus can be removed from the output. Users can also specify a missing rate to control the amount of missing alleles. By combining samples generated based on different populations, one can create a data set that is a mixture of different (isolated) populations. In addition to association studies, the samples can be used in testing algorithms for tag SNP selection or haplotype inference.

Instead of providing three penetrances for one-locus model, users can also specify the prevalence p and the genotype relative risks R_1 and R_2 [5]. The disease penetrance values p_0, p_1, p_2, corresponding to homozygous wild, heterozygous, and homozygous mutant, can be obtained as follows. Let p_d denote the disease allele frequency. Since $R_1 = p_1/p_0$, $R_2 = p_2/p_0$, we have [6],

$$\begin{aligned} p &= (1-p_d)^2 p_0 + 2p_d(1-p_d)p_1 + p_d^2 p_2 \\ &= (1-p_d)^2 p_0 + 2p_d(1-p_d)R_1 p_0 + p_d^2 R_2 p_0 \end{aligned} \tag{1.2.2}$$

Solving for p_0, we have,

$$p_0 = \frac{p}{(1-p_d)^2 + 2p_d(1-p_d)R_1 + p_d^2 R_2} \tag{1.2.3}$$

By definition, p_1 and p_2 are solved based on p_0.

1.2.3 Quantitative Traits

Quantitative phenotypes can also be generated by both the extension method and the block method. To generate phenotypes for a quantitative trait, a quantitative trait nucleotide is chosen according to a specific

allele frequency or a specific marker position provided by users. The phenotypic value of each individual is generated according to the classical single-locus quantitative trait model [7]. More specifically, users can specify the additive (V_A) and dominance (V_D) genetic variances attributable to the quantitative trait nucleotide as proportions of the total phenotypic variance. Denote the proportions as π_A and π_D. Let V_O denote the variance due to all other (genetic and environmental) factors and assume its value is 1. Then V_A and V_D can be calculated based on

$$\frac{V_A + V_D}{V_A + V_D + 1} = \pi_A + \pi_D, \text{ and } \frac{V_A}{V_D} = \frac{\pi_A}{\pi_D}.$$

If one assumes that the phenotypic value of an individual can be partitioned as

$$y = z + u_i = \begin{cases} z & \text{homozygous wild} \\ z + (1+k)a & \text{heterozygous} \\ z + 2a & \text{homozygous mutant} \end{cases}, \tag{1.2.4}$$

where z follows the standard normal distribution, a is half of the difference between two homozygous genotypic values (assume the mutant allele increases the phenotypic value), and k is a parameter representing the dominance effect, it is known [7] that V_A and V_D can be written as

$$V_A = 2p(1-p)a^2(1-k(2p-1))^2, \text{ and } V_D = (2p(1-p)ak)^2, \tag{1.2.5}$$

where p is the frequency of the mutant allele. Thus a and k can be obtained based on the above equations. The phenotypic value of an individual can be calculated by substituting a and k into Equation 1.2.4.

1.2.4 Two-Locus Models

For a two-locus diallelic disease model, two interacting sites are involved, and in theory, one can specify nine penetrances, one for each genotype combination from the two sites. The *gs* program provides two distinct methods to allow users to specify a two-locus model. For the first method, all the nine parameters are free and users have the freedom to assign each penetrance with any probability value. Thus the program can generate datasets with any desired disease models. For the second method, the *gs* program has implemented nine commonly used models in the literature. For each of these nine models, users only need to specify the population prevalence p, and genotype odds ratio(s) $1 + \theta$ for each locus, which are relatively easier to obtain for a disease. The program can automatically calculate the penetrance table. For example, Table 1.1 represents a jointly dominant-dominant model and each cell represents the odds of disease given that an individual has that particular genotype combination g_i. Such a model requires at least one disease allele at both loci to increase disease odds and both loci have the same effect size. Let $Pr(D|g_i)$ denote the probability of an individual being affected given its genotype combination of g_i (*i.e.*, the penetrance of g_i), and let $Pr(\overline{D}|g_i)$ denote the probability of an individual not being affected given its genotype g_i. Based on the definition of the odds of a disease

$$ODD_{g_i} = \frac{Pr(D|g_i)}{Pr(\overline{D}|g_i)} = \frac{Pr(D|g_i)}{1 - Pr(D|g_i)}, \tag{1.2.6}$$

the penetrance of g_i can be calculated using the following formula,

$$Pr(D|g_i) = \frac{ODD_{g_i}}{1 + ODD_{g_i}}. \tag{1.2.7}$$

A corresponding penetrance table is give in Table 1.2. Once the population prevalence p and the genotype odds ratio $(1 + \theta)$ are fixed in this model, the baseline value α, which indicates the odds of disease when the two loci do not carry any disease alleles, can be calculated by plugging the terms in Table 1.2 into the following formula,

$$p = Pr(D) = \sum_i Pr(D|g_i)Pr(g_i). \tag{1.2.8}$$

The frequencies on genotype combinations ($Pr(g_i)$) can be obtained from allele frequencies under the Hardy-Weinberg Equilibrium assumption. By focusing on fully penetrant models (the probability of an individual being affected is either 1 or 0 for any given genotype combination), Li et al. [4] enumerated all the 512 possible combinations and summarized 50 unique ones. We have chosen nine commonly used models and implemented them in the program. Their parameters and usage can be found in the Section 2.4.1.

	bb	Bb	BB
aa	α	α	α
Aa	α	$\alpha(1+\theta)$	$\alpha(1+\theta)$
AA	α	$\alpha(1+\theta)$	$\alpha(1+\theta)$

Table 1.1: Odds table for the jointly dominant-dominant model.

	bb	Bb	BB
aa	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha}{1+\alpha}$
Aa	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha(1+\theta)}{1+\alpha(1+\theta)}$	$\frac{\alpha(1+\theta)}{1+\alpha(1+\theta)}$
AA	$\frac{\alpha}{1+\alpha}$	$\frac{\alpha(1+\theta)}{1+\alpha(1+\theta)}$	$\frac{\alpha(1+\theta)}{1+\alpha(1+\theta)}$

Table 1.2: Penetrance table for the jointly dominant-dominant model.

Once the three by three penetrance table is ready, the *gs* program can calculate the conditional probability of each genotype combination given affected status using a similar formula as Equation 1.2.1. Actual genotypes at the two loci will then be selected based on the conditional distribution. The haplotypes will then be selected independently for these two loci, using either the extension method or the block method.

1.3 Extension to Genetic and Environmental Interactions

We extend *gs* to generate samples involving genetic and environmental interactions. Users can freely specify interactions. Meantime, the program complexity is under control. A set of definitions are made to encapsulate different variations. The effect of a variation is described by its disease risk value. Besides genetic variations, the simulation model may also include environmental factors and other covariates. The disease risk of an individual is determined by the combination of all the risk factors.

1.3.1 Locus Interaction Group

The genetic interaction takes two forms. One is the combination of genotypes. The other is the haplotype. Both are essentially a specific configuration over a set of loci that contributes to the disease. Therefore, we define the locus interaction group as a basic unit describing genetic variations. Loci that have joint effect are put in the same locus interaction group. Loci in different locus interaction groups contribute to the disease independently. As a special case, one independent locus can form a locus interaction group by itself. Users may specify as many locus interaction groups as needed. Each group consists of a series of entries that specify the disease risks associated with each configuration. The simulation program will continue to use bi-allelic SNPs. For instance, following the conventional genotype coding that 1, 2, 3 stand for genotype 11, 12, 22 respectively, an entry "3 1 2, 1.8" means genotype combination (22, 11, 12) has a disease risk of 1.8. In the case of haplotype, an entry "1 2 2, 1.7" indicates the haplotype 122 has a disease risk of 1.7, which actually defines a haplotype effect.

1.3.2 Independent Covariate

Independent environmental factors and other covariates follow specific distributions with different means, variances, and other parameters. We have implemented some common distributions in our *gs* program, such as normal distribution, exponential distribution, Weibull distribution, etc. Users can assign a distribution to certain covariate and provide desired parameters. Obviously, complex distributions or mixed distributions cannot be built in the simulation program in advance. Alternatively, *gs* allows users to provide a file that contains desired values simulated from outside sources, such as R, MATLAB, or their own programs. *gs* will randomly assign values from the provided file to the specified covariate.

1.3.3 Covariate Interaction Rule

We define the covariate interaction rule to incorporate the correlation between genetic variations, environmental factors and other covariates. A covariate interaction rule consists of one or more genetic or environmental variations, and their relationship. In order to simplify the problem to make the simulation more practical, two basic categories are implemented: one is the genetic-environmental ($G \times E$) interaction, and the

other is the environmental-environmental ($E \times E$) interaction. A typical example of genetic-environmental interaction from GAW 15 simulation is that, only smokers with genotype Bb or BB at locus B will have an increased RA risk 1.5. For the second category, the interaction does not involve genetic variations. For example, phenotype latent IgM level and smoking status can have a joint effect to the RA risk. Covariate Interaction Rules are examined one by one after all related covariates have been simulated. The subject will obtain the corresponding risk if certain rule is satisfied.

1.3.4 Affection Status

The affection status is determined after all the genetic variations, environmental factors and other covariates are simulated. For a subject i to be simulated, let θ_{ij} denote the risk value defined by the jth locus interaction group or covariant interaction rule. The combined disease risk for this subject is $R_i = \prod_j \theta_{ij}$. The reciprocal of the combined R_i serves as the mean of an exponential distribution,

$$f(x) = (R_i)e^{-(R_i)x} \tag{1.3.1}$$

from which a random variable x will be drawn. If x is less than a preset threshold, the subject is affected. Otherwise, the subject is normal. This threshold effectively controls the baseline penetrance. Let λ denote the fixed threshold, and $F(x)$ denote the cumulative distribution function of $f(x)$. The probability of an individual getting affected is,

$$Pr(Affected) = Pr(x < \lambda) = F(\lambda) = 1 - e^{-(R_i)\lambda} \tag{1.3.2}$$

Because the baseline individual doesn't carry disease-related variation, the risk is 1. Therefore, the baseline penetrance is $1 - e^{-\lambda}$. When $\lambda = 0.1$, the baseline penetrance is 0.095. Based on Equation 1.3.2, the penetrance of an individual is a function of the combined disease risk of this individual. With λ fixed as 0.1, Table 1.3 lists corresponding penetrances of different risks.

Risk	Penetrance
1.1	0.104165865
1.2	0.113079563
1.5	0.139292024
2.0	0.181269247
3.0	0.259181779
5.0	0.39346934
10.0	0.632120559

Table 1.3: Corresponding penetrances of different disease risks.

Let μ denote the desired baseline penetrance. One can specify $\lambda = -ln(1 - \mu)$ to obtain such baseline penetrance.

1.3.5 Extension of Surrounding Loci

As in previous implementation, *gs* continues to use two methods to populate surrounding loci with diversity. One is the extension method, and the other is the block method. Please refer to Section 1.2 for details.

1.3.6 Alternative Usage

By default, *gs* generates genotypes/haplotypes by perturbing real data or by utilizing haplotype block information. When evaluating power of statistical approaches in detecting disease loci, hundreds or thousands of replicates have to be generated. However, for large scale genome-wide association studies with up to one million SNPs and thousands of individuals, such an approach will take a long time to generate simulated genotype data. Our program provides a simple alternative. Users can choose to generate only genotypes at disease loci, or only SNP markers that are in linkage disequilibrium with disease loci. These genotypes can then be randomly inserted into real data. By iterating this process, users can quickly generate required number of replicates for their experiments. An obvious limitation for such a strategy is that LD patters around disease loci can be arbitrary.

1.3.7 Quantitative Trait

We implemented a simple quantitative trait model. The reciprocal of the combined disease risk for an individual serves as the mean of an exponential distribution, same as in 1.3.1, from which a random variable x will be drawn. The quantitative trait value of this individual will be $-ln(x)$.

1.3.8 Summary

A subject i is simulated as follows. Genetic variations are simulated first. The program will identify all the involving SNP loci from user input, either by absolute positions or by minor allele frequencies. At each SNP position, the program will determine a left and right boundary according to pre-defined threshold lengths and LD values. Within the determined range, the program will randomly pick up a haplotype from the pool. Second, after genetic variations required to determine the affection status are ready, the simulation continues to environmental factors and other covariates. Those independent values are drawn from the specified distributions or provided by the user. Third, all disease risks caused by different factors are collected by checking each locus interaction group and each covariate interaction rule, and the affection status/quantitative trait is determined thereafter. At last, if the affection status satisfies the requirement of the simulation in terms of the desired quantity, or it is quantitative trait, each involving SNP locus is ready to be extended in both directions using either the extension method or the block method.

1.4 Platforms

Executable binaries are available for Windows, Linux and Mac OS X.

Chapter 2

Running the Software

2.1 Obtaining and Installing the Software

The *gs* program and this document can be obtained at http://www.eecs.case.edu/∼jxl175/gs.html. The user may use a tool such as WinZip to unzip the files under Windows and use the command "`unzip`" to unzip the files under Linux or Mac OS X.

2.2 File List

The package contains the executable binary, this manual, a readme file, and several examples organized in the subdirectories. The examples are described briefly in the readme file and in detail in each additional readme file inside each example subdirectory. The necessary input files and parameter files are provided with examples, whose formats are described in the sections that follow.

2.3 Input Files

The extension method requires an input file that contains haplotype samples. Its format is very simple. Each line is a haplotype. For the single-locus model and two-locus model implemented earlier, the allele at each marker is denoted as 0 or 1. The file test_01_data in the subdirectory "example1" is in such format. The haplotypes in this file were downloaded from the HapMap website. Users can use two of them when generating data from a two-locus model. For the use in the extension to simulate genetic and environmental interactions, the allele is denoted as 1 or 2. The file test_data in the subdirectory "example2" and "example3" is in this format.

The block method requires a block structure file that contains the block structure from a region. The file format is the same as the output file from a widely used program Haploview[1]. Briefly speaking, each block

begins with a line that consists of the block index and the marker indices, followed by common haplotypes one line each. At the end of each block there is a line reporting the LD of the block in terms of multi-allelic D'. Here is an example.

```
BLOCK 1.  MARKERS: 1 2 3!  4 5 6 7 8
33121122 (0.825) |0.794 0.206|
11442343 (0.149) |0.310 0.690|
Multiallelic Dprime:  0.785
BLOCK 2.  MARKERS: 10!  11 12 13 14
44123 (0.831)
22211 (0.144)
Multiallelic Dprime:  0.790
```

The symbol "!" after some marker index indicates that the marker is a tag SNP from Haploview. The program *gs* does not use that information when generating samples. The number in the parenthesis is the haplotype frequencies. Transition probabilities are shown as a matrix with this block's haplotypes as the rows and the next block's haplotypes as the columns. In this example, the first block has 8 markers with 2 haplotypes displayed and the second block has 5 markers and 2 haplotypes. The transition matrix can be read as follows: the haplotype 33121122 has a probability of 79.4% connecting 44123 and a probability 20.6% connecting 22211, and so forth.

Because the block structure file usually does not contain all markers, *gs* allows users to provide a supplementary genotype file that contains all markers. In this way, markers not in any block will also be generated. Such genotype file can be in the linkage format or the phased haplotype format same as the phased haplotype format used by Haploview, *i.e.*, one column for family ID, one column for individual ID, followed by haplotypes, and two lines of haplotypes for each individual. The file test_block_geno in the subdirectory "example1" is such a supplementary genotype file in the phased haplotype format. One may generate the block structure file from a genotype file and use the same genotype file as the supplementary genotype file.

To facilitate input file preparation, we have developed a *gs* file conversion tool — *gsconverter* that can directly convert HapMap files and fastPhase output files into *gs* input files. Please see details in Section 2.8.

2.4 Parameter File

The *gs* program relies on the parameter file to specify running mode, SNPs to simulate, disease model, output format and other important parameters. The structure and definitions of a parameter file will be explained in details as follows.

2.4.1 For Previous Single-Locus and Two-Locus Models

The parameter file is organized in the following way. Each parameter consists of a tag and parameter body. The tag occupies one row and has the format "Pi", where i is a number indicator. Each parameter may have multiple values to be specified under the tag. Currently, there are 10 parameters. An example is given below.

```
P1
0
P2
100 100 10
P3
1 0.15
P4
0
0.05 0.075 0.1125
P5
0.0
P6
2 11
P7
0
P8
0 output
P9
0
P10
1
```

P1 indicates *gs* running mode: 0 for case-control data, and 1 for quantitative data.

P2 specifies the number of cases, number of controls and number of replicates one wants to generate for case-control samples. In the case of simulations for quantitative traits, only two values are needed for the second parameter. The first one specifies the number of individuals, and the second one specifies the number of replicates.

P3 is used to specify which marker is disease/trait locus and which allele is the high risk allele (or the allele that increases the value in the case of a quantitative trait). Two values are needed in this line. The first value can be 1 or 0, representing that the allele with greater (or smaller) ID will be the risk allele. To choose a marker, one can either specify a frequency or directly specify a marker position, using the second value. When a frequency f is specified ($0 < f < 1$), the program will automatically pick a marker position with

the frequency of the above specified allele roughly equal to f. If the value is an integer i ($1 \leq i \leq m$, m is the total number of markers), the program will pick the i_{th} marker as the trait locus. For a two-locus model, one can specify the second locus in the same line following the first locus in the same manner.

P4 is used differently for one-locus model and two-locus model.

For case-control samples based on *one-locus* disease model, the fourth parameter has four values to specify. The first is an indicator to use three penetrances directly if set as 0, or to use disease prevalence along with genotype relative risks if set as 1. The following three values are the penetrances of each genotype (in the order of homozygous wild, heterozygous, and homozygous mutant) in the case of directly providing penetrances, or the disease prevalence, genotype relative risk 1 and genotype relative risk 2 in the other case. For the mode of one-locus QTN, users need to specify the π_A followed by the π_D (see Section 1.2).

Regarding a *two-locus* model, a nonnegative integer i ($0 \leq i \leq 9$) is required as an indicator and a built-in model index. If i is zero, users need to provide a 3 by 3 penetrance table like the following example.

```
P4
0
0.025 0.025 0.025
0.025 0.025 0.325
0.025 0.325 0.325
```

If i is greater than zero and less than or equal to 9, the *gs* program will pick the i^{th} built-in two-locus model. Three model parameters are then followed, in the order of population prevalence, genotype odd increments θ_1 and θ_2 for the first and second locus, respectively. If a two-locus model only requires one θ, such as the jointly dominant-dominant model with the same effect at both loci (Table 1.1), θ_2 should have the same value as θ_1. An example is given below.

```
P4
6
0.1 0.25 0.3
```

The complete list of built-in two-locus models and their indices is given in the Table 2.1, followed by their odds tables one by one.

P5 indicates the missing rate.

P6 is to set the values of $min\_overlap$ and $max\_overlap$, indicating the minimum and maximum lengths of the overlapped region when using the option $\in$ (see Section 1.2 for details).

P7 is a binary value which indicates whether *gs* should output the alleles at the disease/trait (when the value is 1), or not (when the value is 0).

P8 has two values. The first one is an integer and can be either 0, 1 or 2, corresponding to one of the three output formats (*i.e.*, phased haplotype format, linkage format, and *gs* own output format). The three formats will be explained in depth in Section 2.6. The second value is a string to specify an output subdirectory under the current running directory. If the subdirectory does not exist, the program will automatically create one.

Index	Model Name
0	(using 3 x 3 penetrance table)
1	Jointly multiplicative-multiplicative model (Table 2.2)
2	Model of two-locus interaction multiplicative effects (Table 2.3)
3	Jointly dominant-dominant model (Table 1.1)
4	Jointly recessive-recessive model (Table 2.4)
5	Jointly recessive-dominant model (Table 2.5)
6	Threshold model (Table 2.6)
7	Additive-additive model (Table 2.7)
8	Exclusive OR model (Table 2.8)
9	A special interaction model (Table 2.9)

Table 2.1: Built-in two-locus model list

	bb	Bb	BB
aa	α	$\alpha(1+\theta_2)$	$\alpha(1+\theta_2)^2$
Aa	$\alpha(1+\theta_1)$	$\alpha(1+\theta_1)(1+\theta_2)$	$\alpha(1+\theta_1)(1+\theta_2)^2$
AA	$\alpha(1+\theta_1)^2$	$\alpha(1+\theta_1)^2(1+\theta_2)$	$\alpha(1+\theta_1)^2(1+\theta_2)^2$

Table 2.2: Jointly multiplicative-multiplicative model

	bb	Bb	BB
aa	α	α	α
Aa	α	$\alpha(1+\theta)$	$\alpha(1+\theta)^2$
AA	α	$\alpha(1+\theta)^2$	$\alpha(1+\theta)^4$

Table 2.3: Model of two-locus interaction multiplicative effects

	bb	Bb	BB
aa	α	α	α
Aa	α	α	α
AA	α	α	$\alpha(1+\theta)$

Table 2.4: Recessive-recessive model

P9 is for users to specify a seed for the random number generator. Users can duplicate their simulation results by assigning a specific seed. More precisely, if the number in this parameter is greater than zero, the random generator of *gs* will take the number as its seed. Otherwise, a time-based seed will be adopted.

	bb	Bb	BB
aa	α	α	α
Aa	α	α	α
AA	α	$\alpha(1+\theta)$	$\alpha(1+\theta)$

Table 2.5: Recessive-dominant model

	bb	Bb	BB
aa	α	α	α
Aa	α	α	$\alpha(1+\theta)$
AA	α	$\alpha(1+\theta)$	$\alpha(1+\theta)$

Table 2.6: Threshold model

	bb	Bb	BB
aa	α	$\alpha(1+\theta_2)$	$2\alpha(1+\theta_2)$
Aa	$\alpha(1+\theta_1)$	$\alpha(2+\theta_1+\theta_2)$	$\alpha(3+\theta_1+2\theta_2)$
AA	$2\alpha(1+\theta_1)$	$\alpha(3+2\theta_1+\theta_2)$	$2\alpha(2+\theta_1+\theta_2)$

Table 2.7: Additive-additive model

	bb	Bb	BB
aa	α	α	$\alpha(1+\theta)$
Aa	α	α	$\alpha(1+\theta)$
AA	$\alpha(1+\theta)$	$\alpha(1+\theta)$	α

Table 2.8: Exclusive OR model

	bb	Bb	BB
aa	α	α	$\alpha(1+\theta)$
Aa	α	α	α
AA	$\alpha(1+\theta)$	α	α

Table 2.9: A special interaction model

P10 specifies starting ID number. By default, the ID number of generated haplotypes/individules in each replicate starts from 1. One can change the initial number by specifying it in this last parameter. By doing so, users can combine outputs generated from tow different inputs. If the two inputs represent two different populations, which might have different allele frequencies and effects at the disease/trait locus, then the

combined output is actually a sample of mixed populations.

2.4.2 For Genetic and Environmental Interactions

Part of the parameter file has changed significantly to meet the new requirements. First of all, the structure of the parameter file is the same. We still use "Pi" tag to separate different parameters. Parameter values must follow the tag without empty lines in between. One can use "//" to write comments at the tail of each line. Comment lines led by "//" and empty lines are allowed after current parameter ends and before next parameter tag.

P1 indicates running mode. 0 is for case-control simulation, and 1 is for quantitative trait simulation. For the mode of case-control simulation, following the 0, additional floating point number is required as the disease affection status threshold mentioned above in formula 1.3.1.

P2 doesn't change, which gives the number of cases, the number of controls and the number of replicates for generating case-control samples, or the number of individuals and the number of replicates in the case of quantitative trait.

P3 accommodates the definitions of Locus Interaction Groups. (1) The Locus Interaction Group definition starts with an indicator "LIG" followed by an ID number. (2) The second line specifies the entry type in this locus interaction group: 1 for genotype, and 2 for haplotype. An entry means a combination of genotype or a haplotype with its disease risk, which is essential to define the disease effect of any variation. (3) The third line indicates how to locate the disease-associated loci. Integer 1 means specifying them by minor allele frequencies, and integer 2 means by absolute marker positions. (4) Depending on the way of specifying disease loci, the fourth line lists the minor alleles with frequencies pair by pair or a series of marker numbers. (5) From the fifth line on, users can specify as many entries as needed. Each entry takes one line and should have identical number of marker genotypes or alleles corresponding to the number of pairs or markers in the fourth line. Meanwhile, there should be no conflicts and redefinitions. In the case of genotype combination, we take the traditional coding. Only number 1, 2, and 3 are allowed and represent homozygous wild-type, heterozygous, and homozygous mutant respectively. *gs* will calculate which allele is the major allele and which one is minor at each locus from the input file. For example, at certain locus, allele 1 is the major allele, and allele 2 is minor. Genotype 3 stands for '22'. In the case of haplotype, a sequence of characters that match the allele characters in the input file at corresponding positions should be given. The entries end by an empty line. More Locus Interaction Groups can be defined one after another.

P4 defines independent covariates and their interactions with other covariates or genetic variations.

We have built in several common distributions that users can select from for a covariate or users can provide a value file that contains a series of numeric values for the specified covariate. (1) the covariate definition starts with the indicator "COV" and the name of this covariate follows. (2) The second line specifies the way of generating values for this covariate: 1 for built-in distribution, and 2 for from file. (3) If the built-in distribution option is selected, the third line is used to specify the distribution and its parameters

given in the parenthesis and separated by commas. Currently, the following distributions are supported:

NORM: Normal distribution with two parameters the mean and the variance.

EXP: Exponential distribution with one parameter the mean.

WEIBULL: Weibull distribution with one scale parameter and one exponent parameter.

If from file option is selected, the third line specifies the file that contains the covariate values. The format of the value file is very simple. It is a text file and each line is a numeric value. The *gs* program will read in all values line by line till the file end. During the simulation procedure, each time one individual being simulated needs a value of this covariate, *gs* will randomly select one from those read in from the value file.

Covariate Interaction Rule(s) should be defined after all involved independent covariates have been defined above. (1) The rule definition starts with "CIR" followed by an ID number. (2) The second line specifies the type of this Covariate Interaction Rule: 1 for genetic-environment interaction, and 2 for environment-environment interaction. (3) In the case of $G\times E$ interaction, the third line to the fifth line are similar to the definition in the Locus Interaction Group, which give the entry type, the way of specifying disease loci, and the minor alleles with frequencies or the marker positions. The sixth line gives the name(s) of involved covariate(s) defined before. In the case of $E\times E$ interaction, the third line gives the name(s) of involved covariate(s) defined before. (4) The following lines are entries defining disease risks. The genetic variations are given similarly as in the Locus Interaction Group definition. Covariate value requirements are specified in the form of a range given in the square brackets. The items in an entry should match the order given in previous lines.

Similarly, more independent covariates and Covariate Interaction Rules can be defined one by one and separated by an empty line. However, to keep the program simple, no duplication of genetic loci in Locus Interaction Group definition and Covariate Interaction Rule definition is allowed. If one locus interacts with several other loci, all of those loci should be defined in one Locus Interaction Group and any one of them should not appear in other Locus Interaction Groups or Covariate Interaction Rules. On the other hand, if several loci interact with certain covariates, they should be defined in a Covariate Interaction Rule and any one of them should not appear in other Locus Interaction Groups or Covariate Interaction Rules.

P5 through P10 remain same as described in Section 2.4.1. A comprehensive pseudo parameter file is given below as an example for illustration purpose only.

```
P1
0 0.1
P2
100 100 10
P3 //locus interaction group
LIG: 1 //group ID
1 //entry type:  1/2 (genotype/haplotype)
1 //specify disease locus:  1/2 (by MAF/marker position)
```

```
2, 0.05; 1, 0.1; 2, 0.25 //minor allele and frequency pair(s)
2 1 3 1.1 //genotype combinations with risks
1 2 2 1.5
3 1 1 1.2

LIG: 2
2 //haplotype
2 //by marker position
3, 4, 5 //marker position(s)
1 2 1 1.2 //haplotype risks
2 2 2 1.8

P4 //covariates and G×E, E×E interaction rules
COV: ENV1 //covariate name
1 //generating way:  1/2 (built-in distribution/outside file)
NORM(0, 1) //distribution name with parameter(s)

COV: ENV2
2 //outside file
env2_values //value filename

CIR: 4 //covariate interaction rule ID
2 //type:  1/2 (G×E/E×E)
ENV1, ENV2 //involved covariate(s)
[1.0, 2.0] [1, 1] 1.8 //covariate condition(s) with risk(s)

CIR: 5
1 //G×E
1 //genotype
1 //by MAF
2, 0.05; 1, 0.1; 2, 0.25
ENV1, ENV2 //involved covariate(s)
2 1 3 [1.0, 2.0] [1.0, 2.0] 1.2 //genotype and covariate condition(s) with risk(s)
3 3 3 [1.0, 2.0] [1.0, 2.0] 1.2

P5 //missing rate
0.0
```

```
P6 //min overlapping length, max overlapping length for extension method
2 11
P7 //output disease locus:  1/0 (yes/no)
0
P8 //output format:  0/1/2 (phased, linkage, gs own), output directory
0 output
P9 //random seed:  0/(an integer) (time-based/user-specified)
0
P10 //starting ID number
1
```

2.5 Running the Software

One can launch *gs* on a Command Prompt (DOS) window on Windows by typing the following command:

```
gs -option infile1 [infile2] parafile
```

Similarly, one can launch *gs* by typing the following command in a Linux or Mac OS X terminal:

```
./gs -option infile1 [infile2] parafile
```

The above commands assume that *gs* is in the current directory. Otherwise, appropriate path needs to be included in front of the executable file. The file infile2 is only needed if one needs to generate samples based on a two-locus model.

2.5.1 Options for Previous Single-Locus and Two-Locus Models

There are four options for generating samples under previous single-locus and two-locus models using extension and block methods. They require different inputs. For a single-locus model, one needs to invoke the option `b` with an input containing a block structure in order to use the block method. To use the extension method, one needs to invoke `e` with an input containing phased haplotype pairs. For a two-locus model, one needs to invoke `b2` and `e2` respectively for the two methods.

```
-b:  block-based
-e:  extension-based
-b2:  two-locus, block-based
-e2:  two-locus, extension-based
```

For the block-based method, in addition to the structure file, one can provide a genotype file that contains all the SNPs. The *gs* program provides two additional options `gp` and `gl` to allow users to specify a supplementary genotype file when using a one-locus model. Each of the two options is followed by a genotype filename, while option `gp` indicates that the genotype file is in phased haplotype format, and

option `gl` indicates the linkage format.

```
-gp <filename>:  provide a genotype file in phased haplotype format
-gl <filename>:  provide a genotype file in linkage format
```

For a two-locus model, users can provide two genotype files using option `gp1` and `gp2`, or `gl1` and `gl2`, for the phased haplotype format and the linkage format respectively.

The following are some sample runs. More are provided in the program package.

```
gs -e test_01_data para.txt
gs -e test_01_data para-qtn.txt
gs -b test_block para.txt -gp test_block_geno
gs -e2 test_01_data test_01_data2 para-2loci.txt
gs -b2 test_block test_block2 para-2loci.txt
```

2.5.2 Options for Genetic and Environmental Interactions

Two options are introduced to invoke simulation of genetic and environmental interactions.

```
-bm:  block-based multiple-factor simulation
-em:  extension-based multiple-factor simulation
```

Similarly, "-gp" option and "-gl" option can be used to provide supplementary genotype file in phased haplotype format or linkage format respectively. The command line launch of the program remains the same. The following are examples to run interaction simulation.

```
gs -em test_data para.txt
gs -bm test_block para-3loci.txt -gp test_block_geno
gs -em test_data para-qt.txt
```

2.6 Output Files

For each replicate, there is one output file (this is also true for a two-locus model). The name of the output file is automatically given based on the method used and a serial number for replicate. For example, "ext20" means the 20th replicate using extension-based method. Similarly, "block16" means the 16th replicate using block-based method. For previous single-locus and two-locus models, in case of QTN simulation, the suffix "-qtn" will be appended at the end of filename, and for a two-locus model, "-2loci" will append to the filename. For genetic and environmental interactions, there will be no additional suffix.

The program can generate three different formats, *i.e.*, phased haplotype format, linkage format, and *gs* own output format.

***gs*'s own format**: each individual consists of two lines, one for each haplotype. For each line, the first

column is the index/ID of a haplotype, and the second column is the status or the trait value of an individual. We use 2 denoting affected status and 1 denoting normal status (both haplotypes from one individual have the same status as the individual), followed by the alleles. The haplotypes downloaded from HapMap website using 0 and 1 denoting alleles at each marker position. But zero is also commonly used to denote missing alleles. In this case, the program automatically changes 0 to 1 and 1 to 2 in the output file and uses 0 to represent missing alleles. The block structure file from Haploview uses 1, 2, 3 and 4 denoting alleles and this will not change in the output file.

Phased haplotype format: it is one input format of Haploview. It also uses two lines to represent the two haplotypes for each individual. But in their format, the first column is family ID and the second column is an individual ID. No disease status or trait values will be printed out. Users can use this format to investigate different tag SNP selection algorithms implemented in Haploview.

Linkage format (pre MAKEPED): it is a widely used format in the statistical genetics community. Each line is an individual with columns of family ID, individual ID, father ID, mother ID, gender, affected status and genotypes.

For all the three formats, the simulated cases are output first followed by the simulated controls.

In addition to the data files, there is one more information file (`ext.info` for the extension method and `block.info` for the block method). This information file simple gives the real disease/trait allele frequency(ies) (which might not be exactly the same as the disease allele frequency specified by users), the disease locus position(s) and the number of total replicates.

When the simulation involves environmental factors to determine the affection status of an individual, their values are also output in a separate file for each replicate. The suffix ".env" will be appended to the replicate filename as its environmental value filename, *e.g.*, "ext1.env". The first line is a column name line including the names of all environmental factors. Each individual occupies one line and has the individual ID and all environmental factor values listed.

2.7 Generating Interacting Disease Loci or Markers in LD Only

By default, *gs* generates genotypes/haplotypes by perturbing real data or by utilizing haplotype block information. When evaluating power of statistical approaches in detecting disease loci, hundreds or thousands of replicates have to be generated. However, for large scale genome-wide association studies with up to one million SNPs and thousands of individuals, such an approach will take a long time to generate simulated genotype data. The *gs* program provides a simple alternative. Users can choose to generate only genotypes at disease loci, or only SNP markers that are in linkage disequilibrium with disease loci. These genotypes can then be randomly inserted into real data. By iterating this process, users can quickly generate required number of replicates for their experiments. An obvious limitation for such a strategy is that LD patters around disease loci can be arbitrary.

The following two options are used to generate disease loci only and SNP markers that are in linkage disequilibrium with disease loci respectively.

```
-dlm:  disease loci only
-dlmld:  only SNP markers in LD with disease loci
```

Since no input is needed, only one parameter filename is required following the option. Please see the examples below.

```
gs -dlm para.txt
gs -dlmld para.txt
```

For the option of generating disease loci only, the parameter file is in the same format as defined before in Section 2.4.2. However, when specifying how to locate the disease-associated loci in the definition of Locus Interaction Group or Covariate Interaction Rule, one can only choose the way of minor allele frequencies.

For the option of SNP markers in LD only, the parameter file requires one additional line following the definition of how to locate the disease loci in each Locus Interaction Group or Covariate Interaction Rule. An example is given below. Each SNP marker in LD with the disease locus is specified by an ordered triple. The first element gives the minor allele of the marker, the second specifies its frequency, and the third gives the r^2 value between the disease locus and the marker. The number of LD marker definitions must match the number of disease loci specified just above them.

```
LIG: 1 //group ID
1 //entry type:  1/2 (genotype/haplotype)
1 //specify disease locus:  only 1 (by MAF) allowed for LD markers
2, 0.25; 2, 0.25; 2, 0.25 //minor allele and frequency pair(s)
2, 0.25, 0.8; 2, 0.25, 0.7; 2, 0.25, 0.9 //matched LD markers
2 1 3 1.1 //genotype combinations with risks
1 2 2 1.5
3 1 1 1.2
```

The output can also select the three formats described in Section 2.6. The prefix of the output files is "dl", *e.g.*, "dl.info", "dl1", "dl1.env", etc.

2.8 *gsconverter*

To facilitate input file preparation, we have developed a *gs* file conversion tool — *gsconverter* that can directly convert HapMap phased haplotype files and fastPhase output files into *gs* input files. The executable is included in the software package.

Similarly as *gs*, to run *gsconverter*, one can type as follows on Windows and Linux/Mac OS X respectively: `gsconverter -option source_file`

```
./gsconverter -option source_file
```

Five options are available:

-01to12 converts 0,1-represented haplotypes to 1,2-represented ones.

This is a simple replacement of 0 by 1 and 1 by 2. The source file contains haplotype samples. Each line is a haplotype. The allele is denoted as 0 or 1, separated by a space. The name of the converted file is the source file name plus a suffix ".gs12".

-12to01 converts 1,2-represented haplotypes to 0,1-represented ones.

This is a reverse replacement of 1 by 0 and 2 by 1. The name of the converted file is the source file name plus a suffix ".gs01".

-12toph converts haplotype samples to phased haplotype format.

The source file contains 1,2-represented haplotypes line by line without leading columns. HaploView cannot directly take such input but the phased haplotype format with two leading columns — Family ID and Individual ID. This option adds two pseudo leading columns so that HaploView can read. This is useful when one wants to generate block structure from HaploView. Moreover, the converted file in phased haplotype format can also serve as the supplementary genotype file when using the block method. The name of the converted file is the source file name plus a suffix ".gsph".

-hm3to12 converts HapMap Phase III phasing file to *gs* input file.

For HapMap Phase III, files are organized in a SNPs$\times$haplotypes format with a header row. Every row that follows represents a SNP and every column a haplotype. This option reads in such file, counts the major and minor alleles, and outputs haplotypes line by line using 1 for major allele and 2 for minor allele. The name of the converted file is the source file name plus a suffix ".gs12".

-fpto12 converts fastPhase output file to *gs* input file.

This option converts the standard fastPhase output file to *gs* 1,2-represented line-haplotype input file. The fastPhase output file should be the standard one with descriptions ahead and inferred haplotypes between labels "BEGIN GENOTYPES" and "END GENOTYPES", not the simplified output generated using "-Z" option. The name of the converted file is the source file name plus a suffix ".gs12".

The following are some sample runs.

```
gsconverter -01to12 test_01_data
gsconverter -12to01 test_data
gsconverter -12toph test_data
gsconverter -hm3to12 hapmap3_r2_b36_chr22_ceu.unr.phased
gsconverter -fpto12 test_hapguess_switch.out
```

2.9 Citation

(i) J Li and Y Chen. Generating Samples for Association Studies Based on HapMap data. BMC Bioinformatics 9:44, 2008.
(ii) Y Chen and J Li. *gs*2.0: a Simulation Tool for Generating Samples Involving Genetic and Environmental Interactions for Association Studies. *Submitted*.

Bibliography

[1] Barrett, J.C., Fry, B., Maller, J. and Daly, M.J. (2005) Haploview: analysis and visualization of LD and haplotype maps. *Bioinformatics*, **21**, 263-5.

[2] Durrant, C., Zondervan, K.T., Cardon, L.R., Hunt, S., Deloukas, P., and Morris, A.P. (2004) Linkage disequilibrium mapping via cladistic analysis of single-nucleotide polymorphism haplotypes. *Am. J. Hum. Genet.*, **75**, 35-43.

[3] Li, J., Y. Zhou and R. C. Elston (2006) Haplotype-based Quantitative Trait Mapping Using a Clustering Algorithm. *BMC Bioinformatics*, **7**:258.

[4] Li, W. and Reich, J. (2000) A Complete Enumeration and Classification of Two-Locus Disease Models. *Hum Hered* **50**:334-349.

[5] Schaid DJ, and Sommer SS. (1993) Genotype relative risks: methods for design and analysis of candidate-gene association studies. *Am J Hum Genet* **53**(5):1114-1126.

[6] Terwilliger JD, and Ott J. (1994) Handbook of Human Genetic Linkage. Baltimore: Johns Hopkins.

[7] Lynch, M and Walsh, B. (1998) Genetics and analysis of quantitative traits. Sinauer Associates, MA, USA.