ANALYSIS OF A HIERARCHICAL BAYESIAN METHOD FOR QUANTITATIVE TRAIT LOCI

Caroline Pearson

A Thesis Submitted to the
University of North Carolina Wilmington in Partial Fulfillment
Of the Requirements for the Degree of
Master of Science

Department of Mathematics and Statistics

University of North Carolina Wilmington

2007

Approved by

Advisory Committee

Chair

Accepted by

Dean, Graduate School

TABLE OF CONTENTS

ABSTRACT

Simulations were performed to compare two methods that detect quantitative trait loci on plant data. Karl Broman's interval mapping algorithm which uses only one observation value per plant line was compared to a hierarchical Bayesian model that allows replicates into the analysis and takes into account the variability within each plant line. The simulation study utilized the genetic map of Bay-0 X Shahdara plant with 38 genetic markers on 5 chromosomes. It is shown through these simulations that the hierarchical Bayesian model and Broman's interval mapping algorithm are able to detect quantitative trait loci (QTL) when only a single location was chosen, but the hierarchical model was more powerful when two locations were chosen. This work shows that when analyzing plant replicates the variability within each line has a strong impact on the success of the overall analyses.

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Susan Simmons for her continued support, guidance, patience, and encouragement. This work would not have been possible without her understanding and support through my entire graduate studies.

Also, I would like to thank the entire faculty and staff of the Department of Mathematics and Statistics for sharing with me their knowledge and appreciation for mathematics and statistics.

DEDICATION

I would like to dedicate this thesis to my husband, Josh, my son, Auston, and my mother, Norma, whose continued support and encouragement have enabled me to do more than I ever thought I could.

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

The growing interest in the genetics field has given way for much research on identifying locations on a genome responsible for a quantitative trait, which is referred to as quantitative trait loci (QTL). Alfred Henry Sturtevant constructed the first genetic map in 1913 [1] which depicts the relative distance between known markers or genes on an organism's genome. The first analysis relating genes to quantitative traits was done in 1923 by Sax [2]. Identifying the genetic loci responsible for the different attributes of traits has been researched for decades and a number of novel methods have evolved. However, which method is the best and most appropriate is still under some debate.

A single trait is usually determined by many genes; as a result, many QTLs are associated with a single trait. The number of QTLs associated with each phenotypic trait tells us the genetic makeup and the variation of this trait. For instance, a small effect can be determined if there are many QTLs correlated with a single trait and a large effect can be determined if there are only a few QTLs correlated with a single trait. The information gleamed from the QTL can help us better understand the chemical structure of these traits, help us better understand the evolution of these traits over a period of time, and eventually enable us to alter the chemical structure of these traits. One potential benefit of understanding plant QTLs is the ability to alter the chemical structure of a plant to make it more tolerant of ultraviolet (UV) radiation, which may help agriculturalists deal with the depleting ozone layer; this layer filters much of the UV radiation before it can enter the atmosphere and ultimately the terra firma.

One popular method for detecting QTLs is the interval mapping algorithm. This method places pseudo-markers in the interval to evaluate the possibility of a QTL in the interval. There are many different variations of interval mapping that have been predominately used today, however, the software packages available for interval mapping only utilize one observation per genotype or line. Experiments involving plant QTLs involve multiple observations per genotype (or plant line); therefore, they require methods that can incorporate not only the mean value but the variance of the genotype (or plant line). In these experiments not only is the mean value an important part of the data but since plants used in these experiments have identical genetic composition the variance also provides relevant information. For instance, if there are 5 cloned plants all consisting of identical genetic makeup with the height of each plant being 15.2 in, 15.6 in, 14.9 in, 15.8 in, 15.5 in, then the mean height of these plants is 15.4 in and the variance is 0.125. However, if there are 5 cloned plants all consisting of identical genetic makeup with the height of each plant being 32.7 in, 7.3 in, 24.8 in, 10.4 in, 1.8 in, then the mean height of these plants is 15.4 in but the variance is 165.805. It can be seen that we can have the exact same mean in this example but these values are from very different populations as evident from the variance. Therefore, it has raised the question, are the methods developed for animal and human QTL analyses appropriate for plant QTL analyses?

A hierarchical Bayesian model [3] has been developed to incorporate this type of information into the detection of QTLs which is explained in more detail in methods section. This thesis compares the performance of the hierarchical Bayesian model in [3] to an interval mapping algorithm of Broman [4] in a simulation study. The study investigates models with one QTL and two QTLs, and low and high effect sizes.

METHODS

The observed quantitative trait in the plant QTL experiment will be represented as y_{ij}, with $i = 1,\ldots, L$ (L = number of plant lines) and $j = 1,\ldots, n_i$ (n_i = the number of replicates). The true mean of y_{ij} will be represented as θ_i for line i, and we assume $y_{ij} \sim \mathrm{N}(\theta_i, \sigma_i^2)$. Each θ_i is assumed to be linearly dependent on the genetic composition of the plant which can be expressed as,

$$\theta_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots \beta_M x_{iM} \qquad (1)$$

where $x_{im} = 1$ if the marker is from parent A and $x_{im} = 0$ if the marker is from parent B. M represents the number of markers.

Broman's Interval Mapping

Interval mapping is a well known method for detecting QTLs today. Karl Broman [4] wrote an interval mapping algorithm freely available in R language which is widely known and well respected for analysis on animal data. Broman's method uses one response value per animal genotype. Suppose there is data on L animal lines derived from an inbred line cross. Quantitative trait measurements are denoted by y_i with $i = 1,\ldots, L$ and the genotyping data are denoted by the x_{im} with $m = 1,\ldots, M$. Information from the known marker genotypes are used to estimate unknown genotypes within an interval. With μ = model parameters and γ = QTL locations, given the observed data, multiple imputed versions of the QTL genotypes are then used to compute approximations to the posterior densities of interest $p(\gamma|y, x)$ and $p(\mu|y, x)$. The posterior density, $p(\gamma|y, x)$, is the probability that location γ is the QTL given the quantitative trait and genotypes. The posterior density, $p(\mu|y, x)$, measures how well the QTL genotypes are matched to the observed marker data. The QTL genotype matrix is denoted as $g = (g_{ij})$ where rows i

correspond to individuals and columns *j* correspond to the location. The posterior distribution is given as,

$$p(g|y, x) \propto p(y|g)p(g|x) \qquad (2)$$

since this distribution is conditioning on the unobserved QTL genotypes.

Assume there are *q* QTLs in the model, then the location γ will have *q* components. Since the locations are not known, all possible locations are scanned to search for the QTL. So simulations are done by simulating genotypes at all locations in the genome from their joint distributions given the known marker data. A discrete grid known as the pseudomarker grid is created for locations spanning the genome. For every multiple pseudomarker locations $u = (u_1, \ldots., u_q)$, the i^{th} realization of genotypes is an $l \ x \ q$ matrix denoted as $r_i(u)$. Weighted sample of QTL genotypes is generated by,

$$W_H(r_i(u)) = p(y|g = r_i(u))p(\gamma = u) \qquad (3)$$

where W_H is the assumed generic model *H* which is a description of the distribution of phenotypes given the QTL genotypes. The posterior distribution of the QTL location has been shown to be proportional to the average weight of all pseudomarker realizations at that location.

$$p(\gamma = u|y, x) \propto p(y|g)p(g|x, \gamma = u)p(\gamma = u)dg \ W_H(r_i(u)) \qquad (4)$$

The idea behind interval mapping is to utilize likelihood ratio test, which can be expressed on the scale base 10 logarithm which is called LOD score shown below in equation 5.

$$\mathrm{LOD} \propto -2\ln\left[\frac{\max(\text{likelihood assuming no QTLs})}{\max(\text{likelihood assuming QTL at location})}\right] \qquad (5)$$

For Broman's algorithm, the LOD score at location *γ* is expressed as,

$$\text{LOD}(\gamma) = \text{constant} + \log_{10}[\sup_{\mu} p(y, x|\, \mu, \gamma)]. \qquad (6)$$

Broman's interval mapping algorithm is a well known and utilized method for QTL detection. However, this method assumes one observation per genotype, or per line, like most QTL models which assume that the variance is the same within each line (Broman and Speed [5]; Lander and Botsterin [6]). The hierarchical model does not make this assumption of homogeneity of variance and is able to incorporate the replicate information within each line.

Hierarchical Bayesian Model

Hierarchical models have proven to be invaluable in many instances (Boone *et al.* [7]; Simmons *et al.* [8]). The hierarchical Bayesian model has more flexibility to adequately analyze plant replicates in a QTL experiment. As described before, the true mean of y_{ij} will be represented as θ_i and the variance as σ_i^2 within line i as using equation (10 to model the true mean θ_i, we assume that $\theta_i \sim \text{N}(X\beta, \tau^2)$. The structure of the data for the hierarchical model is depicted in on the following page. In addition, the quantitative trait within each line is assumed to follow a normal distribution, or in other words $y_{ij} \sim\text{N}(\theta_i, \sigma_i^2)$. The true mean θ_i is assumed to be dependent on the genetic composition via equation (10) and $\theta_i \sim \text{N}(X\beta, \tau^2)$.

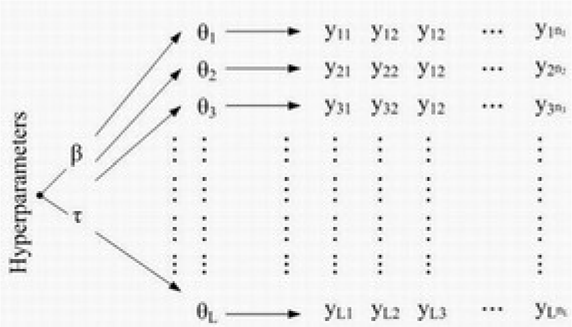
Hyperparameters
β
τ
θ_1 y_{11} y_{12} y_{12} $\cdots$ y_{1n_1}
θ_2 y_{21} y_{22} y_{12} $\cdots$ y_{2n_2}
θ_3 y_{31} y_{32} y_{12} $\cdots$ y_{3n_3}
θ_L y_{L1} y_{L2} y_{L3} $\cdots$ y_{Ln_L}

Figure 1. Structure of Data

The following assumptions are made in regards to the prior distributions,

$$\beta_m \sim \mathrm{N}(0,100) \qquad (7)$$

$$\tau^2 \sim \text{Inverse-}\chi^2\ (2) \qquad (8)$$

$$\sigma_i^2 \sim \text{Inverse-}\chi^2\ (2). \qquad (9)$$

The Inverse-χ^2 (2) has infinite variances (Boone *et al*. [7]) and the posterior distribution for β's assume that no markers have an effect on the quantitative trait. Therefore, this forces the data to dictate which markers are most important with respect to the quantitative trait. Combining this information into a hierarchical model creates a full joint posterior distribution of the form,

$$p(\theta,\beta,\sigma^2,\tau^2 \mid y) \propto (\tau^{\tau_0+2+L}\prod_i (\sigma_i^{n_i+\sigma_{0i}+2}))^{-1}\cdot$$
$$\exp\left[-\sum_i \frac{1}{2\sigma_i^2} - \frac{1}{2\tau^2} - \frac{1}{200}\beta'\beta - \frac{1}{2\tau^2}(\theta - X\beta)'(\theta - X\beta) - \sum_i\sum_j \frac{1}{2\sigma_i^2}(y_{ij}-\theta_i)^2\right]. \qquad (10)$$

Where $i = 1, \ldots, L$, and $j = 1,\ldots, n_i$.

The Gibbs Sampler, a Markov Chain Monte Carlo technique, can generate samples from the full joint posterior distribution in (10) by using the following conditional posterior distributions,

$$p(\tau^2 \mid \theta,\beta,\sigma^2,y) \propto Inv-Gamma\left(\frac{L+2}{2},\frac{(\theta - X\beta)'(\theta - X\beta)+1}{2}\right) \qquad (11)$$

$$p(\sigma_i^2 \mid \theta,\beta,\tau^2,y) \propto Inv-Gamma\left(\frac{n_i+2}{2},\frac{\sum_j (y_{ij}-\theta_i)^2+1}{2}\right) \qquad (12)$$

$$p(\beta \mid \tau^2,\theta,\sigma^2,y) \propto N\left(\left(\frac{I}{100}+\frac{X'X}{\tau^2}\right)\frac{X'\theta}{\tau^2},\left(\frac{I}{100}+\frac{X'X}{\tau^2}\right)^{-1}\right) \qquad (13)$$

$$p(\theta \mid \tau^2, \beta, \sigma^2, y) \propto N\left(\frac{\frac{X_i\beta}{\tau^2} + \frac{n_i\bar{y}_i}{\sigma_i^2}}{\frac{1}{\tau^2} + \frac{n_i}{\sigma_i^2}}, \frac{1}{\frac{1}{\tau^2} + \frac{n_i}{\sigma_i^2}} \right). \tag{14}$$

This information can be used to find posterior probabilities and ultimately determine which markers are important in controlling the quantitative trait. The set of all possible models is denoted as Λ and the K^{th} model by λ_K. The cardinality or size of Λ is denoted by |Λ|. The vector of unknown parameters for model K will be denoted by δ_K.

The probability of model K given the data using Bayes Rule,

$$p(\lambda_K \mid D) = \frac{p(D \mid \lambda_K)p(\lambda_K)}{\sum_{K=1}^{|\Lambda|} p(D \mid \lambda_K)p(\lambda_K)} \tag{15}$$

Since no prior knowledge of which model is most appropriate, each λ_K are equally likely. Then $p(D| \lambda_K)$ is calculated as,

$$p(D \mid \lambda_K) = \int P(D \mid \lambda_K, \delta_K)p(\delta_K \mid \lambda_K)d\delta_K \tag{16}$$

Since the integral will be computational intensive, it can be estimated by Monte Carlo methods as,

$$p(D \mid \lambda_K) = \int P(D \mid \lambda_K, \delta_K)p(\delta_K \mid \lambda_K)d\delta_K \approx \frac{1}{t}\sum_{i=1}^{t} p(D \mid \delta_K^{(i)}, \lambda_K)p(\delta_K^{(i)} \mid \lambda_K) \tag{17}$$

Since there are many unknown parameters in this model, a large number of samples from the posterior are recommended. In this research, t was chosen to be 100,000 with a burn-in period of

2,000. The posterior probability of the model given the data can be used to find the activation probability of a marker, $P(\beta_j \neq 0 \mid D)$. The activation probability is defined as,

$$p(\beta_j \neq 0 \mid D) = \sum_{K=1}^{|\Lambda|} p(\beta_j \neq 0 \mid \lambda_K, D) p(\lambda_K \mid D) \qquad (18)$$

Since this can become computationally intensive given the total number of models is 2^M and most genetic maps have more than 100 markers, a searching technique is utilized that that breaks down the genome into smaller regions by conditioning on the regions of importance. The searching technique first breaks the genome into N chromosomes, yielding 2^N number of models that need to be evaluated. Then it identifies the chromosomes of importance and divides those regions in half. This continues until the important marker(s) are identified. The activation probability for each region R_j is evaluated by,

$$p(R_j \neq 0 \mid D) = \sum_{K=1}^{|\Lambda|} p(R_j \neq 0 \mid \lambda_K, D) p(\lambda_K \mid D) \qquad (19)$$

Regions with posterior probability larger than 0.5 are regarded as potential QTLs and retained in the model. Once all potential regions are identified, those regions retained are divided in half. For example, in a hypothetical example with 7 chromosomes, the search algorithm would first find which chromosomes make a significant contribution to the QTL by searching through all $2^7 = 128$ possible models and calculating the activation probability for each chromosome. For this example, the following activation probabilities were obtained $C_1 = 0.01$, $C_2 = 0.03$, $C_3 = 0.67$, $C_4 = 0.33$, $C_5 = 0.90$, $C_6 = 0.21$ and $C_7 = 0.84$. Chromosomes 3, 5 and 7 have activation probabilities higher than 0.5 and are kept for further analysis. Dividing these chromosomes in half, there are now six regions to explore (i.e. $2^6 = 64$ models). These regions are defined as C_{31}, C_{32}, C_{51}, C_{52}, C_{71} and C_{72}. The algorithm is rerun and activation probabilities for each of these six regions are calculated. Only those regions with activation probability higher than 0.5 are retained and then divided in half. This algorithm is repeated until the activation probabilities are calculated on individual markers.

SIMULATIONS

Simulations were conducted to compare the hierarchical Bayesian method against Karl Broman’s interval mapping. The Bay-0 X Shahdara marker structure was used as the X matrix (165 lines x 38 markers) since it is a well known plant structure and consists of a small number of markers. Figure 2 represents the genetic map of the Bay-0 X Shahdara population created by Oliver Loudet and Sylvian Chaillou [9].

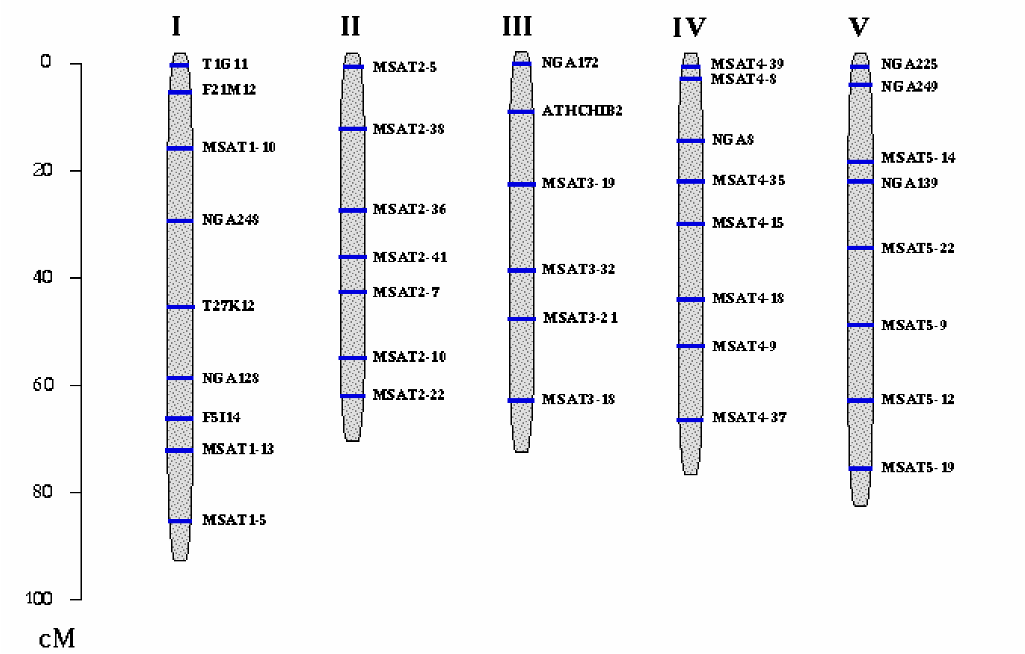

Figure 2. Genetic Map of Bay-0 X Shahdara Population

The Bay-0 X Shahdara population consists of 5 chromosomes which contain 6 to 9 markers each. Marker values (x_i) were set to x_i = 0, 0.5 or 1. The marker value x_i = 0 came from parent A and the marker x_i = 1 came from parent B, whereas, the marker value x_i = 0.5 is a missing or unknown value. Ten response values (y_{ij}) were simulated per line around approximately a 26 unit mean (μ) which depended on the QTL location and genotypic trait.

Simulations were done with one QTL and two QTLs on different chromosomes. The marker values for the one QTL simulations were obtained by

$$y_{ij} = \mu + 2^*a_i\ ^*x_i + \varepsilon_{ij} \tag{20}$$

where, μ = the underlying true mean and a_i is the QTL effect, x_i = 0, 0.5 or 1 and ε_{ij} is random error noise. The variance for the random error noise was simulated using two different methods. A simplistic approach that gave only two variances, σ_1^2 and σ_2^2, was used. A random draw from a Bernoulli distribution with a probability of success 0.5 was used to determine variance within each line. The second method used a gamma distribution, to simulate different variances for each plant line using two different α shape parameters. For the two QTL simulations the marker values were obtained by

$$y_{ij} = \mu + a_1{}^*x_{1i} + a_2{}^*x_{2i} + \varepsilon_{ij} \tag{21}$$

where, μ = the underlying true mean and a_1 and a_2 are the QTL effects and ε_{ij} is random error noise. Different effect sizes were chosen for the simulations that include the gamma distibutions. Slighly larger effect sizes than the perivious simulations were considered to be better suited for the larger degree of variance values obtained. Therefore, effect sizes 5 and 15 were evaluted for the one and two QTL simulations. The varaince values for these simulations were obtained from two gamma distibutions with $\mu = \alpha\beta = 4$ and $\sigma^2 = \alpha\beta^2 = 4$, and $\mu = \alpha\beta = 8$ and $\sigma^2 = \alpha\beta^2 = 8$. The response values ($y_{ij}$) were simulated in the same manner as the simulations conducted with bimodal standard deviations using equation 20 for the one QTL simulations and equation 21 for the two QTL simulations.

RESULTS

Two different methods were tested using the simulated response values and the Bay-0 X Shahdara genetic map (X). To determine the success of one or both methods, there were separate criteria associated with each method. The Interval mapping method uses LOD scores as described in the Methods section in equation 5. The LOD scores were calculated for separate locations on a chromosome, given a predetermined threshold, it is said any value greater that that threshold could be a potential QTL. In the Broman's interval mapping simulations shown below, the threshold value was set to 11, hence, any LOD score greater then 11 was deemed significant. The hierarchical Bayesian method uses a conditional activation probability to determine the locations of interest. Activation probabilities of 0.50 or greater were deemed significant indicating a potential QTL.

Table 1 and Table 2 summarize the results from the one and two QTL simulations using bimodal standard deviations, respectively. Table 3 and Table 4 summarize the results from the one and two QTL simulations using standard deviations derived from gamma distributions, respectively. Small and large effect sizes were chosen to study the power of both methods under different standard deviation scenarios.

Table 1. One QTL Simulation with Bimodal Standard Deviations

Effects	Standard Deviations	True Locations	Hierarchical Bayesian Method	Broman's Interval Mapping Method
2	$\sigma_1 = 2.0$ $\sigma_2 = 4.5$	Chrom 3 M 18	Chrom 3 M 18 (1.000)[1] M 19 (0.987)[1]	Chrom 3 M 19 (52.49)[2]
	$\sigma_1 = 4.2$ $\sigma_2 = 9.1$	Chrom 5 M 36	Chrom 5 M 35 (0.586)[1] M 36 (1.000)[1]	Chrom 5 M 37 (17.53)[2]
12	$\sigma_1 = 2.0$ $\sigma_2 = 4.5$	Chrom 3 M 15	Chrom 2 M 15 (1.000)[1]	Chrom 2 M 16 (117.84)[2] Chrom 3 Loc 2.5 cM (63.05)[2]
	$\sigma_1 = 4.2$ $\sigma_2 = 9.1$	Chrom 2 M 12	Chrom 2 M 11 (0.989)[1] M 12 (1.000)[1]	Chrom 2 M 13 (106.39)[2]

Chrom = Chromosome; M = Marker; Loc = Location; cM = centiMorgan

[1] Final conditional activation probability

[2] LOD score

Table 2. Two QTL Simulation with Bimodal Standard Deviations

Effects	Standard Deviations	True Locations	Hierarchical Bayesian Method	Broman's Interval Mapping Method
1,2	$\sigma_1 = 1.5$ $\sigma_2 = 2.5$	Chrom 1 M 5 Chrom 4 M 24	Chrom 1 M 5 (1.000)[1] Chrom 4 M 24 (1.000)[1]	Chrom 4 Loc: 17.5 cM (21.73)[2]
	$\sigma_1 = 2.0$ $\sigma_2 = 4.5$	Chrom 2 M 11 Chrom 5 M 36	Chrom 2 M 11 (0.997)[1] Chrom 5 M 36 (1.000)[1]	LOD all < 11
1,2	$\sigma_1 = 1.5$ $\sigma_2 = 2.5$	Chrom 2 M 14 Chrom 3 M 21	Chrom 2 M 14 (0.999)[1] Chrom 3 M 21 (1.000)[1] M 22 (0.543)[1]	Chrom 3 Loc: 65 cM (33.94)[2]
	$\sigma_1 = 2.0$ $\sigma_2 = 4.5$	Chrom 1 M 8 Chrom 3 M 18	Chrom 1 M 8 (0.963)[1] Chrom 3 M 18 (1.000)[1] M 19 (0.535)[1]	Chrom 3 M 19 (16.37)[2]
2,12	$\sigma_1 = 1.5$ $\sigma_2 = 2.5$	Chrom 1 M 5 Chrom 4 M 24	Chrom 1 M 5 (1.000)[1] Chrom 4 M 24 (1.000)[1]	Chrom 4 M 25 (88.08)[2]
	$\sigma_1 = 2.0$ $\sigma_2 = 4.5$	Chrom 2 M 11 Chrom 5 M 36	Chrom 2 M 11 (1.000)[1] Chrom 5 M 36 (1.000)[1]	Chrom 5 M 37 (93.55)[2]

Chrom = Chromosome; M = Marker; Loc = Location; cM = centiMorgan

[1] Final conditional activation probability

[2] LOD score

Table 3. One QTL Simulation with Gamma Distribution Parameters

Effects	Gamma Distribution Parameters	True Locations	Hierarchical Bayesian Method	Broman's Interval Mapping Method
5	$\alpha = 4, \beta = 1$	Chrom 1 M 7	Chrom 1 M 7 (1.000)[1] M 9 (0.999)[1]	Chrom 1 M 8 (82.48)[2]
	$\alpha = 8, \beta = 1$	Chrom 4 M 29	Chrom 4 M 29 (1.000)[1] M 30 (0.999)[1]	Chrom 4 M 30 (49.18)[2]
15	$\alpha = 4, \beta = 1$	Chrom 2 M 16	Chrom 2 M 16 (1.000)[1]	Chrom 2 M 16 (19.37)[2] Chrom 3 M 17 (125.38)[2]
	$\alpha = 8, \beta = 1$	Chrom 5 M 33	Chrom 5 M 33 (1.000)[1] M 34 (0.936)[1]	Chrom 5 M 34 (90.23)[2]

Chrom = Chromosome; M = Marker

[1] Final conditional activation probability

[2] LOD score

Table 4. Two QTL Simulation with Gamma Distribution Parameters

Effects	Gamma Distribution Parameters	True Locations	Hierarchical Bayesian Method	Broman's Interval Mapping Method
5,5	α = 4, β = 1	Chrom 1 M 4 Chrom 2 M 14	Chrom 1 M 1 (0.824)[1] M 3 (1.000)[1] M 4 (0.838)[1] Chrom 2 M 13 (0.822)[1] M 14 (1.000)[1]	Chrom 1 M 5 (23.91)[2] Chrom 2 M 15 (22.06)[2]
	α = 8, β = 1	Chrom 2 M 11 Chrom 5 M 32	Chrom 2 M 11 (1.000)[1] M 12 (0.793)[1] Chrom 5 M 31 (0.795)[1] M 32 (1.000)[1]	Chrom 2 Loc: 25 cM (12.53)[2] Chrom 5 Loc: 15 cM (14.31)[2]
5,15	α = 4, β = 1	Chrom 3 M 18 Chrom 4 M 27	Chrom 3 M 18 (1.000)[1] Chrom 4 M 27 (1.000)[1] M 28 (0.997)[1]	Chrom 4 M 28 (71.47)[2]
	α = 8, β = 1	Chrom 1 M 7 Chrom 3 M 21	Chrom 1 M7 (1.000)[1] Chrom 3 M 20 (0.998)[1] M 21 (1.000)[1] M 22 (0.999)[1]	Chrom 3 M 22 (57.95)[2]

Chrom = Chromosome; M = Marker; Loc = Location; cM = centiMorgan

[1] Final conditional activation probability

[2] LOD score

The bimodal standard deviation simulations in Table 1 and in Table 2 clearly indicate that the hierarchical Bayesian method is able to detect the simulated QTLs in both one and two QTL scenarios. In many instances, the hierarchical Bayesian method detects adjoining markers which is not uncommon in QTL analysis and could indicate some underlying correlation between the markers. Broman's interval mapping algorithm, in Table 1, for the one QTL simulation is also able to detect the approximate location of the quantitative trait, but will only detect the region around the QTL by selecting the marker immediately after the QTL. In Table 2, Broman's interval mapping algorithm for the two QTLs scenario is only able to detect the QTL with the larger effect size. For the instance of effect sizes 1 and 2, Broman's method was not able to detect either of the QTLs.

The simulations with the standard deviations derived from gamma distributions in Table 3 and Table 4, the hierarchical Bayesian method performed just as well as the bimodal standard deviation simulations. The hierarchical Bayesian method detected every QTL in the one and two QTL scenarios along with some possibly correlated markers. In the instance of two QTL simulation with two small effect sizes of 5 and 5 with gamma distribution parameter $\alpha = 4$, the hierarchical Bayesian method obtains a false positive value at marker 1 on Chromosome 1 (activation probability = 0.824). However, the hierarchical Bayesian method was still able to detect the correct markers 4 and 14 on Chromosomes 1 and 2, respectively. Broman's interval mapping algorithm also performed similarly, as it did in the bimodal standard deviation simulations. However, Broman's algorithm only identified the markers immediately after the QTL in the one QTL scenarios and in the two QTL scenarios it only identified the markers immediately after the QTL with the larger effect size. For the instance of effect size 5 and 5 in the two QTL simulations, Broman’s method was able to identify the correct chromosomes; however, it was not able to detect either of the QTLs.

Broman’s method never actually identifies the QTLs exactly for any of the scenarios presented above, only the approximate area in some instances. Whereas, the hierarchical Bayesian method was always able to detect the QTLs for each scenario even though the method did detect other markers of importance as well that could possibly be due to these markers being correlated.

CONCLUSIONS

Methods utilized today for QTL analysis are mainly developed for one genotype per line. However, plant biologists can clone plants and create replicates within each line, thus, making the methods and software available today lose valuable information. These methods only utilize one response value per line so they summarize the response values into one, usually into a mean or median. In the simulations shown above and in Pearson *et al.* [10] it is clear that the inclusion of the variability within each line does have a significant impact on the success of QTL analysis. Given smaller effect sizes, it can sometimes be a limiting factor to summarize information and completely disregard the variability. This can possibly lead to false positives or missing the QTL altogether, where in the hierarchical Bayesian model proposed, small effect sizes do not hinder the success of the method. However, more research on this method and comparisons to other software packages available is needed.

REFERENCES

1. Sturtevant, A. H.: The Linear Arrangement of Six Sex-linked Factors in *Drosophila*, as Shown by their Mode of Association. Journal of Experimental Zoology 14 (1913) 43-59.

2. Sax, K.: The association of size differences with seed-coat pattern and pigmentation in Phaseolus vulgaris. Genetics 8 (1923) 552-560.

3. Boone, E.L., Simmons, S.J., Bao, H., Stapleton, A.E.: Bayesian Hierarchical Regression Models for detecting QTLs in plant experiments. Submitted to Journal of Applied Statistics.

4. Broman, K.W., Wu, H., Sen Ś, Churchill G.A. R/qtl: QTL mapping in experimental crosses. Bioinformatics 19 (2003) 889-890.

5. Broman, K.W., Speed, T.: A Model Selection Approach for the Identification of Quantitative Trait Loci in Experimental Crosses. Journal of the Royal Statistics Society 64 Part 4 (2002) 641-656.

6. Lander, E.S., Botstein, D.: Mapping Mendelian factors underlying traits using RFLP linkage maps. Genetics 121 (1989) 185-199.

7. Boone, E., Ye, K., Smith, E.P.: Assessment of two approximation methods for computing posterior model probabilities. Computational Statistics and Data Analysis 48 (2005) 221-234.

8. Simmons, S.J., Piegorsch, W. W., Nitcheva, D. and Zeiger, E.: Combining environmental information via hierarchical modeling: an example using mutagenic potencies. Environmentrics 14 (2003) 159-168.

9. Loudet, O., Chaillou, S., Camilleri, C., Bouchez, D., Daniel-Vedele, F.: Bay-0 x Shahdara recombinant inbred lines population: a powerful tool for the genetic dissection of complex traits in Arabidopsis. Theoretical and Applied Genetics 104 6-7 (2002) 1173-1184.

10. Pearson, C., Susan, S. J., Ricanek, K., Boone, E. L.: Comparative Analysis of a Hierarchical Bayesian Method for Quantitative Trait Loci Analysis for the Arabidopsis Thaliana. Submitted to Recognition in Bioinformatics (accepted).

APPENDIX A

FORTRAN Code (Full Model)

```
      program gibbsrout
      USE MSIMSL

      PARAMETER (M=39,L=164,taunot=1.d0,sigmanot=1.d0,KK=102000,
      &           kutoff=2000,sigbeta = 100.d0)
!                 M is number of Markers (column) and L is number of lines

      DOUBLE PRECISION  X(L,M),tau2(1), Xnew1(L,M),
     &                    taua,Y(L,12),dni(L),
     &                    sigma2(L),thetas(L),
     &                    ybar(L),sumy(L),
     &                    ybar2(L),
     &                    sigmaa(L),sumy2(L)

      INTEGER ni(L),NOBS,M

         !Setting parameters
      dL=L + 0.d0
      taua = taunot + (dL/2.d0)
      NOBS = 0

      open(10,file='ni.csv',status='old')
         read(10,*) (ni(i), i = 1,L)
      close(10)

      do i = 1,L
         dni(i) = ni(i) + 0.d0
         sigmaa(i)=(dni(i)/2) + sigmanot
```

```
enddo
open(16, file='bayxsha.csv', status='old')
    do i=1,L
        read(16,*) (X(i,j),j=1,M)
    enddo
    close(16)

    open(19, file='ysim.csv', status='old')
    do i=1,L
        read(19,*) (Y(i,j), j=1,ni(i))
    enddo
    close(19)

    do i=1,L
    sumy(i) = 0.d0
    sumy2(i) = 0.d0
    NOBS = NOBS + ni(i)
    end do

    do i=1,L
    do j=1,ni(i)
        sumy(i) =sumy(i) + Y(i,j)        !Create ybar
        sumy2(i) = sumy2(i) + Y(i,j)*Y(i,j)
   enddo
   ybar(i) = sumy(i)/dni(i)
    thetas(i) = ybar(i)
    sigma2(i) = (sumy2(i) - dni(i)*(ybar(i)**2))/(dni(i) - 1.d0)
    if (sigma2(i).eq.0.d0) sigma2(i) = 1.d0
    ybar2(i) = sumy2(i)/dni(i)
    enddo

    do i = 1,L
```

```
          sumtheta = sumtheta + thetas(i)
          sumtheta2 = sumtheta2 + (thetas(i)**2)
     enddo
     thetabar = sumtheta/dL
     tau2 = (sumtheta2 - dL*(thetabar**2))/(dL - 1.d0)

!Different x matrix*************************************
!***************************************************
 do i = 1,L
Xnew1(i,1) = X(i,1)
enddo

do ic1 = 0,1
     do ic2 = 0,1
          do ic3 = 0,1
               do ic4 = 0,1
                    do ic5 = 0,1

open(50,file='Bayesoutput.txt',status='old',access='append')
write(50,*) "Model ", ic1,ic2,ic3,ic4,ic5
close(50)

M1 = 1
if (ic1.eq.1) M1 = M1 + 9
if (ic2.eq.1) M1 = M1 + 7
if (ic3.eq.1) M1 = M1 + 6
if (ic4.eq.1) M1 = M1 + 8
if (ic5.eq.1) M1 = M1 + 8

do i = 1,L
nbegin = 2
     if (ic1.eq.1) then
     k = 2
     do nmark = nbegin, (nbegin + 8)
          Xnew1(i,nmark) =  X(i,k)
          k = k + 1
     enddo
     nbegin = nbegin + 9
endif
if (ic2.eq.1) then
     k = 11
     do nmark = nbegin, (nbegin + 6)
          Xnew1(i,nmark) =  X(i,k)
          k = k + 1
     enddo
     nbegin = nbegin + 7
endif
if (ic3.eq.1) then
     k = 18
     do nmark = nbegin, (nbegin + 5)
          Xnew1(i,nmark) =  X(i,k)
          k = k + 1
     enddo
     nbegin =  nbegin + 6
endif
if (ic4.eq.1) then
     k = 24
     do nmark = nbegin, (nbegin + 7)
          Xnew1(i,nmark) =  X(i,k)
          k = k + 1
     enddo
     nbegin = nbegin + 8
endif
if (ic5.eq.1) then
```

```
            k = 32
            do nmark = nbegin, (nbegin + 7)
                Xnew1(i,nmark) =  X(i,k)
                k = k + 1
            enddo
        endif
    enddo
    CALL Gibbs (Xnew1,Y,ni,L,M1,taua,sigmaa,ybar,
    &           thetas,ybar2,tau2,KK,kutoff,
    &           sigma2,M,NOBS,sigbeta)

    enddo
    enddo
    enddo
    enddo
    enddo

    !**************************************************
    stop
    end

c   ***************************************************************************
c   ***************************************************************************

!   ______________________________________________________________________
!   SUBROUTINES

    SUBROUTINE Gibbs (Xold,Y,ni,L,M1,taua,sigmaa,
    &           ybar,thetasold,ybar2,tau2old,KK,kutoff,
    &           sigma2old,M,NOBS,sigbeta)

    DOUBLE PRECISION Xold(L,M),XB(L),tau2(1),
&                    taua,taub(1),sigmab(L),Y(L,12),betamu(M1),
&                    covarbeta(M1,M1),sigma2old(L),thetamu(L),
&                    thetasold(L),thetasig(L),ybar(L),
&                    stdtau2(1),betasst(M1),stdsig(L),ybar2(L),
&                    stdtheta(L),sigmaa(L),minloglik,SSE,
&                    liktemp(KK),temp4,temp5,maxloglik,XTX(M1,M1),
&                    sumtemp4,bayesfac,RSIG(M1,M1),TOL,betas(M1),
&                    X(L,M1),tau2old(1),DMACH,thetas(L),sigma2(L),
&                    xregress(NOBS,M1),yregress(NOBS),SST,
&                     XTXold(M1,M1),betasst2(M1)

INTEGER ni(L),IRANK,KK,kutoff

TOL = 100.0*DMACH(4)
minloglik = 1.d8
maxloglik =  -1.d8
sumtemp4 = 0.d0
tau2(1) = tau2old(1)
icount = 0

do i =1,L
    do j = 1,M1
        X(i,j) = Xold(i,j)
enddo
enddo

    num = 1
    do i = 1,L
        do j = 1,ni(i)
            yregress(num) = Y(i,j)
            num = num + 1
        enddo
    enddo

    num2 = 1
    do i = 1,L
        do k = 1,ni(i)
            do j = 1,M1
                xregress(num2,j) = X(i,j)
            enddo
            num2 = num2 + 1
        enddo
    enddo

CALL DRLSE (NOBS, yregress, M1, xregress, NOBS, 0, betas,
&       SST, SSE)

CALL DMURRV (L, M1, X, L, M1, betas, 1, L, XB)
                                                    !Mult matrix x vector
do i =1,L
    thetas(i) = thetasold(i)
    sigma2(i) = sigma2old(i)
enddo

CALL DMXTXF (L, M1, X, L, M1, XTX, M1)
                                                    !Calculates XTX
 do i = 1,M1
    do j=1,M1
        XTXold(i,j) = XTX(i,j)
enddo
enddo

    !Gibbs Sampler
    do k=1,KK
    !***** THETAS       ***************************
        CALL thetapar (tau2,sigma2,XB,L,ybar,ni,thetamu,thetasig) !parameter
        CALL DRNNOR (L,stdtheta)
        do i=1,L
        thetas(i) = stdtheta(i)*thetasig(i) + thetamu(i)
        enddo
    !***** TAU          ***************************
        CALL tauparm (thetas,XB,L,taub)
        CALL drngam(1,taua,stdtau2)
        tau2(1) = taub(1)/stdtau2(1)

    !***** BETA     ***************************
    CALL betapar (XTX,M1,tau2,L,thetas,X,betamu,covarbeta,sigbeta)
    CALL DCHFAC (M1, covarbeta, M1, TOL, IRANK, RSIG, M1)
                                            ! Cholesky factor

    CALL DRNNOR(M1,betasst)
    CALL DMURRV(M1,M1,RSIG,M1,M1,betasst,1,M1,betasst2)
        do i=1,M1
        betas(i) = betasst2(i) + betamu(i)
        enddo

    CALL DMURRV (L, M1, X, L, M1, betas, 1, L, XB) !Mult matrix x vector

    do i = 1,M1
        do j=1,M1
            XTX(i,j) = XTXold(i,j)
        enddo
    enddo

!   *****  SIGMA        ***************************
    CALL sigmaparm (ybar,ybar2,ni,thetas,L,sigmab)
```

```
 CALL drngam(L,sigmaa(1),stdsig)
      do i = 1,L
      sigma2(i) = sigmab(i)/stdsig(i)
      enddo

 CALL llike (betas,XB,tau2,Y,sigma2,thetas,
&          L,M1,sigmaa,taua,temp4,temp5,sigbeta,icountup)

   liktemp(k)=temp4
   if ((temp5.ge.maxloglik) .and. (k.ge.kutoff)) maxloglik = temp5
   if ((temp5.le.minloglik) .and. (k.ge.kutoff)) minloglik = temp5
   if (k.ge.kutoff) icount = icount + icountup
   enddo          ! Here ends the simulation for the Gibbs Sampler
 do k=(kutoff+1),KK
      sumtemp4 = sumtemp4 + liktemp(k)
 enddo
  denom = (KK-(kutoff+1.0)+0.d0)
  bayesfac = sumtemp4/denom
 write(*,*) 'bayesfac = ', bayesfac ,maxloglik,minloglik

 open(50,file='Bayesoutput.txt',status='old',access='append')
 write(50,*)  bayesfac
 close(50)

      return
      end

 SUBROUTINE tauparm (thetas,XB,L,taub)
 DOUBLE PRECISION sumTXB,taub(1),thetas(L),XB(L)
 INTEGER L

         sumTXB=0.d0
```

```
        do i=1,L
          sumTXB=sumTXB + (thetas(i) - XB(i))*(thetas(i) - XB(i))
     &    +1.d0

       enddo

      taub(1)=0.5*sumTXB
  return
        end

  SUBROUTINE sigmaparm (ybar,ybar2,ni,thetas,L,sigmab)
  DOUBLEPRECISION ybar(L),thetas(L),sumythetas,sigmab(L),ybar2(L),
  &  dni(L)
  INTEGER ni(L)

      sumythetas=0.d0

      do i=1,L
      dni(i) = ni(i) + 0.0
      sigmab(i) = 0.5*(1+(dni(i)*ybar2(i) - 2*thetas(i)*dni(i)*
  &    ybar(i) + dni(i)*thetas(i)*thetas(i)))
      enddo
  return
      end

  SUBROUTINE betapar (XTX,M1,tau2,L,thetas,X,betamu,covarbeta,
  &  sigbeta)
  DOUBLE PRECISION    XTX(M1,M1),step1(M1,M1),covarbeta(M1,M1),
  &  mupart2(M1),thetas(L),betamu(M1),tau2(1),X(L,M1)
  INTEGER M1,L

      do i=1,M1
      do j=1,M1
      if (i.eq.j) then
      step1(i,j)=(1/sigbeta)+((1/tau2(1))*XTX(i,j))
          else
              step1(i,j) =  ((1/tau2(1))*XTX(i,j))
      endif
      enddo
      enddo

      CALL DLINDS (M1, step1, M1, covarbeta, M1)
      CALL DMURRV (L, M1, X, L, L, thetas, 2, M1, mupart2)

      do i = 1,M1
          mupart2(i) = mupart2(i)/tau2(1)
      enddo

      CALL DMURRV (M1, M1, covarbeta, M1, M1, mupart2, 1, M1, betamu)

      return
      end

      SUBROUTINE thetapar (tau2,sigma2,XB,L,ybar,ni,thetamu,thetasig)
      DOUBLE PRECISION tau2(1),sigma2(L),XB(L),ybar(L),thetamu(L),
     &    thetasig(L),dni(L)
      INTEGER L ,ni(L)

      do i=1,L
      dni(i)=ni(i) + 0.0

      thetamu(i) = (1/tau2(1))*(tau2(1)*sigma2(i)/(dni(i)*tau2(1)
     & +sigma2(i)))*XB(i) +(1/sigma2(i))
     & *(tau2(1)*sigma2(i)/(dni(i)*tau2(1)+sigma2(i)))*
```

```
 &   dni(i)*ybar(i)
     enddo

     do i=1,L
         thetasig(i) = sqrt(tau2(1)*sigma2(i)/(dni(i)*tau2(1)
 &   +sigma2(i)))
     enddo

     return
     end

   SUBROUTINE llike (betas,XB,tau2,Y,sigma2,thetas,
&        L,M1,sigmaa,taua,flik,likehood2,sigbeta,icountup)

   DOUBLE PRECISION  betas(M1),XB(L),tau2(1),
 &                    taua,Y(L,10),btb,thetas(L),
 &                    sigma2(L),sigmaa(L),lik1,lik2,likehood,flik,
 &                    likehood2
  INTEGER M1,L

   lik1=0.d0
   lik2=0.d0
   btb=0.d0
   icountup = 0

 do i=1,L
     lik1= lik1 - (sigmaa(i))*dlog(sigma2(i)) -
&    (1/(2.d0*sigma2(i))) -
&    (1/(2.d0*tau2(1)))*
&    (thetas(i) - XB(i))*
&    (thetas(i) - XB(i))
```

```
      end do

 do i=1,L
       do j=1,10
             lik2 = lik2 -(1/(2.d0*sigma2(i)))*(Y(i,j)-thetas(i))*
 &           (Y(i,j)-thetas(i))
       end do
     end do

     do i = 1,M1
       btb=btb + betas(i)*betas(i)
       end do

     likehood = lik1 + lik2 - (taua)*dlog(tau2(1))
&      - (1/(2.d0*tau2(1))) - (1/(2.d0*sigbeta)) * btb
       likehood2=likehood + 3000  !Adjusting likelihood
       if (likehood2.gt.10.d0) icountup = 1
       if (likehood2.gt.10.d0) likehood2 = -9999.d200
     flik = dexp(likehood2)
       return
       end
```

APPENDIX B

R Code for One QTL Simulation with Bimodal Standard Deviations

```
Xold<-read.table('bayxsha.csv',sep=',')
 X<-Xold[,-1]
y<-matrix(nrow=length(X[,1]),ncol=10)
meany<-rep(0,length(X[,1]))
mu<-26
effect<-2
QTL<-12
for (i in 1:length(X[,1]))
{if (X[i,QTL]==1) meany[i]<-mu + 2*effect
if (X[i,QTL]==0.5)  meany[i]<-mu + effect
if (X[i,QTL]==0) meany[i]<-mu}

for (i in 1:length(X[,1]))
{poin<-sample(c(0,1),1,replace=TRUE)
if (poin==0)
{for (j in 1:10)
   {y[i,j]<-rnorm(1,meany[i],4.2)}}
if (poin==1)
   {for (j in 1:10)
{y[i,j]<-rnorm(1,meany[i],9.1)}}
}
write.table(y,'ysim.csv',sep=',',row.names=FALSE,col.names=FALSE)
```

APPENDIX C

R Code for Two QTL Simulation with Bimodal Standard Deviations

```
Xold<-read.table('bayxsha.csv',sep=',')
 X<-Xold[,-1]
y<-matrix(nrow=length(X[,1]),ncol=10)
meany<-rep(0,length(X[,1]))
mu<-26
effect1<-2
effect2<-12
QTL1<-11
QTL2<-36
meany<-vector(length=length(X[,1]))
for (i in 1:length(X[,1]))
{meany[i]<-mu+effect1*(X[i,QTL1])+effect2*(X[i,QTL2])}

for (i in 1:length(X[,1]))
{poin<-sample(c(0,1),1,replace=TRUE)
if (poin==0)
{for (j in 1:10)
   {y[i,j]<-rnorm(1,meany[i],2.0)}}
if (poin==1)
   {for (j in 1:10)
{y[i,j]<-rnorm(1,meany[i],4.5)}}
}
write.table(y,'ysim.csv',sep=',',row.names=FALSE,col.names=FALSE)
```

APPENDIX D

R Code for One QTL Simulation with Gamma Distribution Parameters

```
Xold<-read.table('bayxsha.csv',sep=',')
 X<-Xold[,-1]
y<-matrix(nrow=length(X[,1]),ncol=10)
meany<-rep(0,length(X[,1]))
mu<-26
effect<-15
QTL<-16
for (i in 1:length(X[,1]))
{if (X[i,QTL]==1) meany[i]<-mu + 2*effect
if (X[i,QTL]==0.5)  meany[i]<-mu + effect
if (X[i,QTL]==0) meany[i]<-mu}

sigvec<-rep(0,length(X[,1]))
sigvec<-rgamma(length(X[,1]),shape=4,scale=1)

for (i in 1:length(X[,1]))
{for (j in 1:10)
   {y[i,j]<-rnorm(1,meany[i],sigvec[i])}
}
write.table(y,'ysim.csv',sep=',',row.names=FALSE,col.names=FALSE)
```

APPENDIX E

R Code for Two QTL Simulation with Gamma Distribution Parameters

```
Xold<-read.table('bayxsha.csv',sep=',')
 X<-Xold[,-1]
y<-matrix(nrow=length(X[,1]),ncol=10)
meany<-rep(0,length(X[,1]))
mu<-26
effect1<-5
effect2<-5
QTL1<-11
QTL2<-32
meany<-vector(length=length(X[,1]))
for (i in 1:length(X[,1]))
{meany[i]<-mu+effect1*(X[i,QTL1])+effect2*(X[i,QTL2])}

sigvec<-rep(0,length(X[,1]))
sigvec<-rgamma(length(X[,1]),shape=8,scale=1)

for (i in 1:length(X[,1]))
{for (j in 1:10)
  {y[i,j]<-rnorm(1,meany[i],sigvec[i])}
}
write.table(y,'ysim.csv',sep=',',row.names=FALSE,col.names=FALSE)
```