EFFICIENT KERNEL ALGORITHMS FOR QTL MAPPING PROBLEMS

KAJSA LJUNGBERG*, SVERKER HOLMGREN*, ÖRJAN CARLBORG†

Abstract. The advent of sophisticated and powerful methods for molecular genetics pushes the need for efficient methods for data analysis. Advanced algorithms are necessary for extracting all possible information from laboriously obtained data sets. We present a general linear algebra framework for QTL mapping, applicable to many commonly used methods, using both linear regression and maximum likelihood estimation. The formulation simplifies future comparisons between and analyses of the methods. We show how the common structure of QTL analysis models can be used to improve the kernel algorithms, drastically reducing the computational effort while retaining the original analysis results. We have evaluated our new algorithms on data sets originating from two large F_2 populations of domestic animals. Using an updating approach, we show that 1-3 orders of magnitude reduction in computational demand can be achieved for matrix factorizations. For interval mapping/composite interval mapping settings using a maximum likelihood model, we also show how to use the original EM algorithm instead of the ECM approximation, significantly improving the convergence and introducing an additional reduction in the computational time. The algorithmic improvements makes it feasible to perform analyses previously deemed impractical or even impossible. For example, using the new algorithms it is reasonable to perform permutation testing using exhaustive search on populations of 200 individuals for fully epistatic two-QTL models with a large number of parameters.

1. Introduction. Molecular dissection of complex traits is currently an important topic in genome research, and the number of quantitative trait loci (QTL) mapping studies performed is constantly growing. The data sets become larger and more complex, implying that the computational requirements increase. Furthermore, epistasis between multiple QTL is often important for quantitative traits. However, most analyses consider only a single QTL, since searching for multiple and interacting QTL leads to a dramatic increase in the computational time compared to single QTL scans. For the general applicability of accurate multidimensional QTL analyses, extracting the most from the results of the toilsome data collection, there is a need for faster numerical algorithms.

A review of QTL mapping techniques for designed crosses is given in [Doerge, 2002]. QTL analysis experiments for livestock is reviewed in [Andersson, 2001], where several QTL with large and moderate effects are reported. QTL analysis applied to Drosophila is reviewed in [Mackay, 2001].

Most standard techniques for QTL analysis in experimental crosses are based on Interval Mapping (IM) [Lander and Botstein, 1989], which is further developed by the introduction of Composite Interval Mapping (CIM) [Zeng, 1993] and Multiple QTL Mapping (MQM) [Jansen, 1993]. These methods use a grid based on a genetic map to give estimates of both the QTL effect and the QTL location in the genome. All three techniques may be combined with either a linear regression (LR) [Haley and Knott, 1992, Martinez and Curnow, 1992, Haley et al., 1994] or maximum-likelihood (ML) [Lander and Botstein, 1989, Jansen, 1993, Zeng, 1994, Kao et al., 1999] statistical model. At genomic locations corresponding to fully informative markers, the LR and ML models yield identical results. However, for a putative QTL located between markers, the genotype is unknown and must be estimated in some way. Here, the LR and ML models are based on different approaches, and also give different results.

*Department of Scientific Computing, Information Technology, Uppsala University, Box 120, SE-751 04 Uppsala, Sweden (kl@tdb.uu.se, sverker@tdb.uu.se).

†Department of Animal Breeding and Genetics, Swedish University of Agricultural Sciences, Box 7023, SE-750 07 Uppsala, Sweden (Orjan.Carlborg@hgen.slu.se).

In a search for single QTL, the parameters in the LR or ML model are computed at loci defined by a uniform grid covering the genome. For each considered location in the grid, a statistical test quantity is computed. The search for the most likely position of a QTL corresponds to a one-dimensional optimization problem, which is presently often solved by performing an exhaustive grid search. In a search for d QTL, the search space is d-dimensional. If the grid covering the genetic map has k grid points, the statistical test quantity must be computed at $\sim k^d$ locations when performing an exhaustive search. This often results in a prohibitive amount of computations, and better optimization schemes will be needed. However, for both current and future optimization schemes, an increase in performance of the method used for computing the statistical test quantity can dramatically increase the performance and practical applicability of QTL analysis software.

In the optimization problem, the evaluation of the object function consists of the solution of one LR or ML problem. Here, the computational demand for performing an LR estimation is often considered to be rather small, since it consists of a single, linear least-squares problem. ML estimation is more demanding, since the computational problem is non-linear. Often, the iterative expectation-maximization (EM) algorithm, or the closely related ECM scheme [Meng and Rubin, 1993, Zeng, 1994], is applied for maximizing the likelihood.

For both LR and ML models, the computational complexity increases with growing population sizes. Also, to determine if the solution to the optimization problem corresponds to a statistically significant QTL, the statistical test quantity must be compared to a significance threshold. A robust approach for determining this is by permutation tests [Churchill and Doerge, 1994], where a large number of QTL scans have to be performed in permuted data sets. Of course, this drastically increases the computational time for the complete QTL analysis. Multidimensional permutation tests are in general avoided because of the computational cost, and theoretically derived significance thresholds are used instead. These are however less robust as they rely on a number of assumptions which may be questionable.

The purpose of this paper is to, by employing a linear algebra framework for the QTL analysis techniques mentioned above, present several alternate numerical methods for the kernel parameter estimation problems in the QTL scans. In the new numerical methods, special features present in many practical QTL studies are utilized to reduce the computational complexity. Note that changing the algorithms for the kernel operations will not change the results of the QTL analyses. However, the computational times are dramatically reduced, enabling new types of QTL scans.

An introduction to the kernel problems in QTL mapping is given in Section 2, and the matrix framework is presented in Section 3. Algorithms based on updating matrix factorizations are discussed in Sections 4, 5 and 6. In Section 7, the LR model is considered, while the new version of the EM algorithm for the ML model is described in Section 8. Finally, the performance of the new algorithms is analyzed in Section 9, where QTL scans for experimental data sets are performed. Here, the computational times are compared to those of standard methods used in QTL mapping software.

2. Kernel problems in QTL mapping. In standard QTL mapping software, the LR and ML estimation problems are solved using numerical library routines, e.g., from NAG, LINPACK or LAPACK. These routines are of excellent quality, and are normally highly efficient for general LR and ML problems. However, such routines may perform unnecessary computations and do not employ special features of the data sets and algorithms, present in many QTL mapping problems for experimental populations. There is a significant potential for deriving more efficient kernel routines, specifically aimed at QTL analysis problems.

A major potential improvement of the standard algorithms is to efficiently employ similarities between the different parameter estimation problems solved during the QTL scan. When using standard library routines, each subsequent LR or ML problem is solved as being independent. However, in many QTL mapping studies, environmental and genetic effects, for example determined by sex, feeding, season, and family (in the sequel denoted *fixed effects*), are included in the model together with the genetic effects of the putative QTL. All parameters are then jointly estimated at each genomic location. Here, the genetic QTL effects vary over the genome whereas the fixed effects are constant. Also, to increase the power of the statistical test, markers can be included in the model as genetic *cofactors* [Jansen, 1993, Zeng, 1993]. Mathematically, the cofactors essentially behave as additional fixed effects. Moreover, in many QTL analyses, forward selection is used in the way that when one QTL has been found, it is included in the model when continuing the search for additional QTL. In practice, this increases the number of fixed effect parameters. Finally, when using an ML model parts of the data remain fixed over the iterations in the EM algorithm. This is exploited in [Zeng, 1994]. There the EM algorithm is replaced by the ECM algorithm [Meng and Rubin, 1993], which is obtained by introducing an approximation in the EM algorithm, resulting in that the each iteration requires fewer arithmetic operations than in the original version. However, the approximation affects the speed of convergence of the iterative process.

In the MQM approach [Jansen, 1992, 1993], an augmented design matrix is formed. This matrix is constant over the entire genome, provided that marker cofactors are included only as fixed effects. Using an ML model combined with the EM algorithm, the M-step in the iteration reduces to a standard weighted least-squares problems, where the weights change between the iterations. Efficient routines for such problems may be available in libraries used for statistical computing. However, since the weights change, a refactorization procedure must be included in computations.

For experimental data including fixed effects and techniques including cofactors and a window of analysis, we below present numerical methods based on only updating part of the factorization of the design matrix. This is achieved without increasing the size of the matrix. Also, we propose a new formulation of the EM algorithm, which has the same computational complexity advantages as the ECM scheme, but without the risk of introducing slower convergence. The algorithms significantly reduces the computational complexity for a variety of methods and QTL scans.

3. A framework for QTL models. Using matrix notation, it is possible to form a unified framework for the standard methods for QTL analysis. We first form a QTL model including fixed effects and cofactors, covering different experimental population types and multidimensional QTL scans. Let $X \in \mathbb{R}^{m\times n}$ be the design matrix, where m is the number of observations. The total number of parameters in the model is $n \equiv f + c + g$, where f is the number of fixed effects including the mean, c is the number of marker cofactors and g is the number of genetic QTL effects. Columns 1 to f are constant at all locations in the genome. If cofactors are included, $c > 0$. Using the CIM method, columns $f+1$ to $f+c$ are constant, but when the marker represented in a cofactor column is too close to the current location in the genome, it must be removed from the model in order not to absorb all the variance and/or make the design matrix singular. The columns $f+c+1$ to n correspond to genetic effects of the putative QTL, which are different at each point in the genome grid.

By defining a partitioning of the design matrix X, a general QTL model is given by

$$y = Xb + r \equiv [X_f X_c X_g]b + r, \tag{3.1}$$

$$X_f \in \mathbb{R}^{m\times f}, X_c \in \mathbb{R}^{m\times c}, X_g \in \mathbb{R}^{m\times g}, b \in \mathbb{R}^{n\times 1}, y \in \mathbb{R}^{m\times 1}, r \in \mathbb{R}^{m\times 1}.$$

Here, the vector y contains the observations of phenotypic values, and r is the corresponding residual vector. X_f contains indicator variables for fixed effects, X_c contains indicator variables for marker cofactors, and X_g contains indicator variables for the genotype. In the QTL scan, the aim is to find b such that $\|r\|_2$ is minimized, and to compare min $\|r\|_2$ to, e.g., the residual r_{null} of the null hypothesis,

$$y = [X_{fc}]b_{fc} + r_{null}, \tag{3.2}$$

where $X_{fc} = [X_f X_c]$ and b_{fc} is the $(f + c)$-vector of partial regression coefficients for the fixed effects and the marker cofactors.

As an example, consider searching for single QTL in an F_2-population. Let α and β denote homozygous maternal and paternal genotypes, and let γ denote the heterozygote. In an F_2 crossing, two genetic effects are considered, implying that $g = 2$. Hence, the X_g matrix consists of two column vectors. For individual i, $x_{i,f+c+1}$ takes the values -1, 0 and 1, and $x_{i,f+c+2}$ takes the values 0, 1 and 0 for the α, β and γ genotypes, respectively. The parameters b_{f+c+1} and b_{f+c+2} are the additive and dominance genetic parameters, normally denoted by a and d in QTL models. Correspondingly, when searching for single QTL in a backcross population, $g = 1$ and X_g is a single column vector. Searching for multiple QTL results in that g is larger. For example, for an F_2 population, a two-QTL model without epistatic interaction yields $g = 4$, while a model including epistasis results in $g = 8$.

4. QR factorization. Methods based on both linear regression and maximum likelihood models normally involve a factorization of (parts of) the design matrix. The standard method is QR factorization, for example employed in the NAG routine G02DAF and the LINPACK routines SQRDC/SQRSL, which are used in standard QTL mapping software [Haley et al., 1994, Basten et al., 2001].

The QR factorization of X is given by

$$X = [X_f X_c X_g] = QR,$$

where $Q \in \mathbb{R}^{m\times m}$ is an orthogonal matrix and $R \in \mathbb{R}^{m\times n}$ is upper triangular. The standard method for computing the QR factorization is to multiply X with a sequence of n Householder matrices H_i, such that $H_n H_{n-1} \dots H_1 X \equiv Q^T X = R$. By construction $H_i^T = H_i$ and $H_i H_i = I$.

If X is (almost) rank deficient, i.e. there are columns in X which are (almost) linearly dependent, the standard QR factorization procedure breaks down. Linearly dependent columns in X imply that two or more regression variables describe the same effect, normally corresponding to an improperly chosen model. There are several potential reasons for such problems to occur. The problem may be global, e.g., if completely correlated fixed effects are included. An example of local rank deficiency could occur in multi-QTL models, where the genomic positions considered in X_g are located too close together. Then, there will be no (or very few) recombination events between the putative QTL positions, and the corresponding columns in X_g will be identical (or very similar). In this case there is not enough information in the data to accurately model the effects of two putative QTL. In summary, if X is (almost) rank deficient, the analysis software should notify the user of this, and the selection of parameters should possibly be modified.

5. Updating the QR factorization. QR factorization is computationally demanding, and for linear regression it constitutes nearly all of the computational work required to solve the problem. However, as a consequence of how the Householder matrices are constructed, the product $H_i H_{i-1} \dots H_1$ and columns 1 to i of R only depend on columns 1 to i of X. This may be exploited to construct schemes for updating the QR factorization of X, drastically reducing the computational effort. Such algorithms are described e.g. in [Björck, 1996].

As X_f is fixed for all positions in the genome, so is $Q_f \equiv H_1 H_2 \dots H_f$ and R_f, containing the first f columns of R. We get $Q_f^T X_f = R_f$. The matrix X_c consists of c columns, one for each marker cofactor. Using the factorization of X_f, we can compute the partial factorization $Q_f^T X_{fc} = Q_f^T [X_f X_c] = [R_f (Q^T X_c)]$. Continuing the QR-factorization gives $Q_{fc}^T X_{fc} \equiv H_{f+c} \dots H_{f+1} Q_f^T X_{fc} = R_{fc}$. When searching for one or more QTL, any marker cofactor within the window of analysis must be temporarily excluded from X_c. Excluding cofactor j, $1 \leq j \leq c$, amounts to removing column j from X_c, giving the matrix $X_c^{(j)}$. It is not necessary to start the factorization procedure over again with $[X_f X_c^{(j)}]$, but the factorization $Q_{fc}^T X_{fc} = R_{fc}$ which is already obtained can be used as a starting point. When excluding cofactor j the corresponding column in R_{fc} is removed, while Q_{fc}^T is not changed. Removing a column from R_{fc} results in non-zero subdiagonal entries in columns $f + j$ to $f + c$ of the corresponding reduced matrix. By applying $c - j$ Givens rotations, $G_{j+1} \dots G_c$, these entries are zeroed, resulting in the factorization $G_c \dots G_{j+1} Q_{fc}^T X_{fc}^{(j)} = R_{fc}^{(j)}$, where $R_{fc}^{(j)}$ is the triangular factor of $X_{fc}^{(j)}$. If more than one cofactor is to be excluded, the procedure is repeated.

Using the ML model, the QR factorization of X_{fc} (the part of the design matrix that is treated linearly) is required. Here, $X_{fc} = Q_{fc} R_{fc}$, including all cofactors, need only be computed once. Then, the updating procedure presented above is used to update the factorization whenever a marker cofactor enters the window of analysis, and when the cofactor leaves the window, the original factorization is used again.

When employing the LR model, the QR factorization of $X = [X_{fc} X_g]$ is needed, where X_g is different at every genome position. The factorization $Q_{fc}^T X_{fc} = R_{fc}$ is again computed once. To complete the factorization of X, the partial factorization $Q_{fc}^T X = [R_{fc} (Q_{fc}^T X_g)]$ is determined. By applying g additional Householder transformations we obtain $H_{f+c+g} \dots H_{f+c+1} Q_{fc}^T X = Q^T X = [R_{fc} R_g] = R$. Whenever a cofactor enters the window of analysis $Q_{fc}^T X_{fc} = R_{fc}$ is replaced by $G_c \dots G_{j+1} Q_{fc}^T X_{fc}^{(j)} = R_{fc}^{(j)}$, and the final g Householder transformations are computed as before. When solving the least squares problem, $Q^T y$ is needed. As Q_{fc}^T is fixed, $Q_{fc}^T y$ need only be computed once, and $Q^T y$ is updated along with $Q^T X$.

6. Arithmetic complexities. By deriving approximative formulas for the arithmetic complexities of different kernel algorithms, it is possible to perform preliminary studies of the performance for different QTL analysis settings without actually performing the computations.

The QR factorization of X_{fc} requires $\sim 2(f + c)^2 (m - (f + c)/3)$ arithmetic operations. Updating the factorization when excluding cofactor j requires $\sim 5(c-j) + 3(c - j)^2$ arithmetic operations, to be compared with the $\sim 2(f + c - 1)^2 \cdot (m - (f + c - 1)/3)$ arithmetic operations needed for a complete refactorization of $X_{fc}^{(j)}$. For a model with 200 individuals, 2 fixed effects (e.g. mean and sex), and 20 marker cofactors, updating saves more than 99% of the computations needed for the QR factorization. The gain is large since the cost of updating does not depend on m.

The complete factorization of X without updating requires $\sim 2n^2(m-n/3)$ arithmetic operations. Once the QR factorization is known, the work required for computing the residual and/or the regression parameters is insignificant. Determining the QR factorization of X using updating requires $\sim 4g(f+c)(m-(f+c-1)/2)+2g^2(m-f-c-g/3)$ arithmetic operations. The number of extra operations needed when a cofactor is excluded is negligible provided that $m \gg n$ and $(f+c) > g$, which is normally the case. Then, updating saves $\sim 2(f+c)^2 m$ arithmetic operations for each locus. In addition, updating $Q^T y$ saves $\sim 4(f+c)(m-(f+c-1)/2)$ arithmetic operations. For example, for $m = 191$, $f = 31$, $c = 0$ and $g = 2$, as in one of the real data sets we have used, introducing algorithms based on updating decreases the number of arithmetic operations by $\sim 90\%$. Replacing the fixed effects with marker cofactors only very slightly reduces this gain. The relative gain introduced by employing the updating algorithms depends on the ratio g/n. When using $g = 8$ in the example above the number of arithmetic operations is reduced by $\sim 65\%$.

7. Linear regression (LR). In the LR model, the entries in X_g are computed at each genomic location based on the genotypes of the markers flanking the putative QTL, and the recombination frequencies between those markers. The indicator variables are defined by computing the a priori probabilities conditional on the marker genotypes, using a simple recombination model. The estimates do not depend on the solution to the regression problem, and the parameters are found by solving the standard least squares problem,

$$\min_b (y - Xb)^T (y - Xb). \tag{7.1}$$

In principle, solving (7.1) gives the optimal parameter vector b^*, which can then be used for calculating the norm of the residual, $\|r\|_2^2 = (y - Xb^*)^T(y - Xb^*)$, used in the test statistic computation. However, it is more efficient to compute this quantity using the formula $\|r\|_2^2 = z^T z$ where z contains the last $(m-n)$ entries of $Q^T y$.

8. Maximum Likelihood (ML). If a ML model is used, maximum likelihood estimates of the indicator variables in X_g are needed. These depend on the marker genotypes as well as density functions of the phenotypes. The parameter values from the solution to the regression problem are needed for determining the variables themselves, leading to a non-linear computational problem.

8.1. Composite interval mapping (CIM). Using the CIM method [Zeng, 1993, 1994], the genetic paramaters are determined with ML estimation using marker genotypes and phenotypes, and assumptions of normally distributed phenotypes and equal variances. The likelihoods are maximized iteratively via the ECM algorithm. Below, we consider the search for a single QTL in an F_2 population. However, the procedure of improving the computational algorithm is easily adapted to other types of experimental crosses and QTL scans.

For ease of notation, we will use $b_a \equiv b_{f+c+1}$ and $b_d \equiv b_{f+c+2}$ for the additive and dominance genetic parameters. Assume that the residuals r_i are normally distributed with mean zero and equal variance σ^2. For each considered location in the genomic grid, the goal is to maximize the likelihood function

$$\prod_{i=1}^{m} [p'_{i,\alpha} f_i(\alpha) + p'_{i,\gamma} f_i(\gamma) + p'_{i,\beta} f_i(\beta)], \tag{8.1}$$

where $p'_{i,\alpha}$, $p'_{i,\gamma}$ and $p'_{i,\beta}$ are the a priori probabilities of individual i having genotype α, γ and β, given the marker genotypes. The normal density functions $f_i(\alpha)$, $f_i(\gamma)$ and $f_i(\beta)$ with variance σ^2 and means $\sum_{j=1}^{f+c} b_j x_{ij} - b_a$, $\sum_{j=1}^{f+c} b_j x_{ij} + b_d$ and $\sum_{j=1}^{f+c} b_j x_{ij} + b_a$, respectively, specify the densities for the random variables y_i.

The m-vector p'_γ of a priori probabilities of genotype γ is equal to the second column of X_g used in the LR setting. Let p_γ be the corresponding vector of a posteriori probabilities. Let p_β be the vector of a posteriori probabilities of genotype β, and p_α the vector of a posteriori probabilities of genotype α. Finally, let 1 denote a vector of all ones, and $X_{fc} \equiv [X_f X_c]$.

Differentiating (8.1), and setting the derivatives to zero gives

$$b_a = (p_\beta - p_\alpha)^T (y - X_{fc} b_{fc}) / (p_\beta + p_\alpha)^T 1, \tag{8.2}$$
$$b_d = p_\gamma^T (y - X_{fc} b_{fc}) / p_\gamma^T 1, \tag{8.3}$$
$$0 = X_{fc}^T (y - X_{fc} b_{fc} - (p_\beta - p_\alpha) b_a - p_\gamma b_d), \tag{8.4}$$
$$m\sigma^2 = (y - X_{fc} b_{fc})^T (y - X_{fc} b_{fc}) - b_a^2 (p_\beta + p_\alpha)^T 1 - b_d^2 p_\gamma^T 1. \tag{8.5}$$

In [Zeng, 1994], the ECM algorithm is used, where b_a, b_d, b_{fc} and σ^2 are sequentially determined using the most current estimates of the solution variables. This corresponds to a Gauss-Seidel approach for computing the new iterands, which reduces the computational complexity since the same QR-factorization of X_{fc} can be used in (8.4) in all iterations. However, since the equations are not solved simultaneously, the derivatives are not set to zero exactly. As shown in the experiments in Section 9, this results in slower convergence.

In the following, we show that it is possible to solve all the equations simultaneously and still keep the computations to a minimum. Thus we can use the standard EM algorithm with its faster convergence, giving an improvement of the overall efficiency. Inserting (8.2) and (8.3) into (8.4) gives

$$X_{fc}^T (I - P_g N^{-1} P_g^T) X_{fc} b_{fc} = X_{fc}^T (I - P_g N^{-1} P_g^T) y, \tag{8.6}$$

where $P_g = [(p_\beta - p_\alpha) p_\gamma]$, a g-column matrix, and $N = diag((p_\beta + p_\alpha)^T 1, p_\gamma^T 1)$, a diagonal $g \times g$ matrix. The entries in P_g are the a posteriori correspondents to the a priori probabilities in X_g. Equation (8.6) is exactly the normal equations resulting from the least squares problem

$$\min_{b_{fc}} (y - X_{fc} b_{fc})^T V (y - X_{fc} b_{fc}), \tag{8.7}$$

where $V = (I - P_g N^{-1} P_g^T)$. Solving (8.7), given V, represents the M-step in the EM-algorithm.

Inserting (8.2) and (8.3) into (8.5) gives

$$m\sigma^2 = (y - X_{fc} b_{fc})^T V (y - X_{fc} b_{fc}),$$

which equals the sum of squared residuals of the LS problem, i.e. the optimal (minimal) value of (8.7). For other types of crossings, the derivation of the M-step is exactly the same. For example, for a backcross, p_α and p_γ are set to zero, P_g reduces to a column vector, and N is a scalar instead of a 2×2 diagonal matrix.

We now turn to the E-step in the EM algorithm, which amounts to obtaining a new P_g. For this we need $(y - X_{fc} b_{fc})$. It is helpful to compute σ^2, b_a and b_d in intermediate steps, but b_{fc} is never needed. This resembles the situation for the LR model, discussed in Section 7, where only the norm of the residual, not the parameter estimates themselves, is needed for the hypothesis test.

Let $Q_1 R_1 = X_{fc}$ be the thin QR factorization of X_{fc}, where $Q_1 \in \mathbb{R}^{m\times(f+c)}$ are the first $(f+c)$ columns of Q_{fc}, which span the same subspace as X_{fc}, and $R_1 \in \mathbb{R}^{(f+c)\times(f+c)}$ is the first $(f+c)$ rows of R_{fc}. Given a nonsingular R_1, the matrix inversion lemma and some algebraic manipulation gives

$$R_1 b_{fc} = [I + Q_1^T P_g (N + P_g^T Q_1 Q_1^T P_g)^{-1} P_g^T Q_1] Q_1^T (I - P_g N^{-1} P_g^T) y. \tag{8.8}$$

Note that $(N + P_g^T Q_1 Q_1^T P_g)^{-1}$ is easily computed, since it is just a $g \times g$ matrix (E.g. a 2×2 matrix for an F_2 population). Only the residual is required for the statistical test, and the system of equations for b_{fc} need not be solved. Some algebra yields that the residual is given by

$$r = y - Q_1[I + Q_1^T P_g (N + P_g^T Q_1 Q_1^T P_g)^{-1} P_g^T Q_1] Q_1^T (I - P_g N^{-1} P_g^T) y. \tag{8.9}$$

If the QR-factorization of X_{fc} is available, the number of arithmetic operations needed for computing r is small. Once r has been calculated, the formulas (8.2), (8.3), and (8.5) are employed to compute b_a, b_d, and σ^2, from which P_g for the next iteration is determined.

8.2. Multiple QTL mapping (MQM). The MQM method combined with the EM algorithm for solving the ML problem can be included in the linear algebra framework presented in Section 3. In MQM, the augmented design matrix X_{aug} is constructed by duplicating observations, introducing one row in the design matrix for each possible genotype. Hence, X_{aug} may be much larger than the corresponding X in the IM/CIM methods. For example, for an F_2 experiment, $m_{aug} \geq 3m$. When maximizing the likelihoods using the EM algorithm, the observations are assigned weights depending on the a posteriori probabilities of the respective genotypes. The basic computational operation consists of solving the weighted least squares problem

$$\min_b (y_{aug} - X_{aug} b)^T W (y_{aug} - X_{aug} b), \tag{8.10}$$

where the diagonal matrix of weights W changes between the iterations, note the similarity in structure with (7.1) and (8.7). Here, solving (8.10), given W, forms the M-step in the EM-algorithm. For this setting, the computational efficiency can not easily be improved by the type of updating procedures that we propose in earlier sections. The weight matrix W represents a full rank modification of the identity, as opposed to the rank-g modification in the CIM method. An updating scheme would require the inverse of an $m_{aug} \times m_{aug}$ matrix, which is not feasible. Hence, a complete refactorization of X_{aug} probably has to be performed in each iteration. A potential approach for reducing the arithmetic work, which is not pursued here, is described in [O'Leary, 1990]. There, the changes in the matrix W are approximated by small rank matrices, which are then employed in an updating procedure similar to the one described in Section 8.1. Note that the resulting iteration is an approximation of the EM algorithm (c.f. the ECM scheme), probably introducing additional iterations.

9. Results. We first study the computational times for a complete genomic scan using our routine UQRLS based on updating the QR factorization. In the experiments we use the LR model, but the results are equally applicable to an ML setting where updating combined with the new verison of the EM algorithm, described in section 8.1, is employed. The UQRLS timings are compared to those of two standard library routines; G02DAF from the NAG Fortran library, used in [Haley et al., 1994], and SQRDC/SQRSL from the LINPACK library. In [Basten et al., 2001], C-versions of SQRDC/SQRSL are used for computing the QR factorization. Like our UQRLS routine, G02DAF and SQRDC/SQRSL solve least squares problems by QR factorization of X.

The NAG routine G02DAF provides $\|r\|_2^2$ as part of the output, while a separate computation with small arithmetic complexity is required if SQRDC/SQRSL is used. UQRLS returns $\|r\|_2^2$.

As discussed in Section 4, a singular or close to singular design matrix indicates that the model might be dubious, and a computational routine used in practical analyses should give notice when the problem is close to rank deficient. G02DAF can handle rank deficient matrices by using singular value decomposition instead of QR factorization when needed. G02DAF also provides a detailed analysis of the problem, which is not used in the QTL analysis context. SQRDC/SQRSL assumes a non-singular matrix and can not be used if X is rank deficient. UQRLS also assumes a non-singular matrix. Both routines can easily be modified to detect rank deficiency at low computational cost.

We have tested the routines on two data sets. The first data set is obtained from an F_2 intercross between the European Wild Boar and Large White breed, used in e.g. ([Andersson et al., 1994]), with $m = 191$ individuals and $f = 31$ fixed effects. Here, we have performed analyses both for a one-QTL model ($g = 2$) and a two-QTL model including epistasis ($g = 8$). The second set consists of simulated data, with dimensions similar to those in a Red Junglefowl and White Leghorn F_2 intercross. Here, $m = 900$ and $f = 46$, and again both one-QTL and complete two-QTL analyses are performed. No marker cofactors are included in the analyses, since this will only marginally affect the computational performance.

The size of the pig and chicken genome maps are approximately 2500 and 2750 cM, respectively. An exhaustive search in steps of 2 cM thus requires 1250 and 1375 evaluations of the kernel problem. An exhaustive two-QTL search with 2 cM step size in each dimension corresponds to approximately $7.8 \cdot 10^5$ evaluations of the kernel problem for the pig data set, and $9.5 \cdot 10^5$ evaluations for the chicken data set. The timings do not include the computation of the entries in the design matrix X. However, using a reasonably efficient implementation, the time required for this should be insignificant when compared to the genome scan. The computations were performed on a server with modern 900 MHz processors. Using another computer could change the CPU times, but the relative differences between the results would be approximately the same.

Figure (9.1) shows the total CPU time required for solving the least squares kernel problem in a full genomic scan for two one-QTL and two two-QTL models. The gain in using UQRLS instead of G02DAF is dramatic. The difference is $2-3$ orders of magnitude for all problem sizes tested. UQRLS requires 5 and 55 minutes respectively for the two-dimensional scans, while G02DAF would need 11 hours and 13 days, respectively. The differences depend both on the reduction of work introduced by the updating procedure and the extra calculations performed by G02DAF. Comparing SQRDC/SQRSL and G02DAF shows that the unnecessarily detailed analysis and robustness of G02DAF are responsible for a $1-2$ orders of magnitude increase in computational effort, compared to a routine that computes the QR factorization using the standard algorithm. Hence, a comparison of SQRDC/SQRSL and UQRLS demonstrates the gain of our updating algorithm. Updating reduces the CPU time by approximately one order of magnitude for one-QTL models and slightly less for two-QTL models. Note that this corresponds well to the gain in

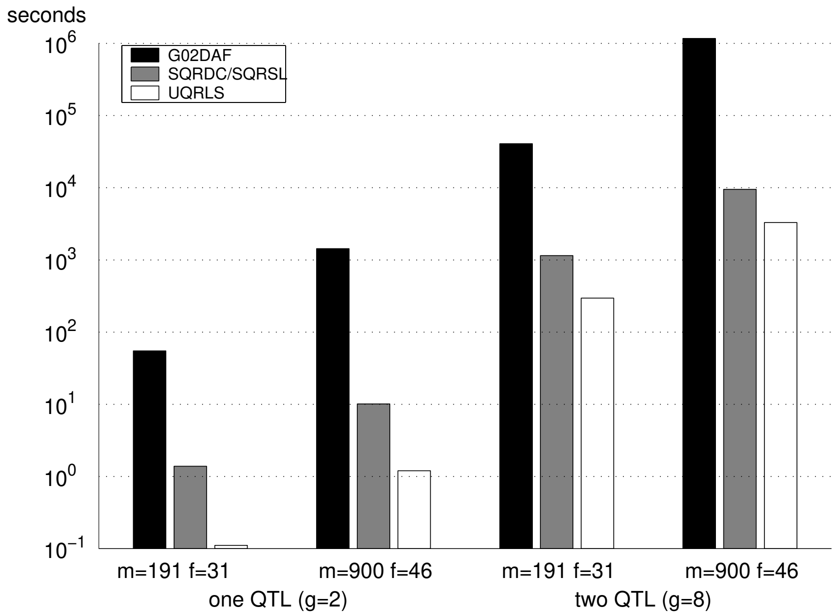

FIG. 9.1. *CPU times for the kernel problem*

arithmetic operations derived in Section 6.

We now turn to evaluating the convergence properties of the new version of the EM algorithm, presented in Section 8.1. Also in these experiments, we for simplicity refrain from using cofactors, resulting in a standard IM model using ML estimation. The aim of the experiments is to compare the convergence rates of the ECM and EM algorithms In CIM fixed effects and marker cofactors are mathematically treated in almost exactly the same way, and the results concerning computational demand can equally well be applied to a CIM model.

We present results for complete genome scans on the pig data set, calculating the ML estimates and test statistics at 1 cM intervals at 2259 genomic positions in total. There were $f = 31$ fixed effects and $g = 2$ genetic effects included in the model. We performed the computations using five versions of the ML solver:

- ECM-null: The ECM algorithm using the null hypothesis parameter estimates as initial values, as is the default in [Basten et al., 2001].
- EM-null: The EM algorithm with the null hypothesis estimates as initial values.
- ECM-prev: The ECM algorithm using the parameter estimates from the previous location as initial values.
- EM-prev: The EM algorithm with the previous estimates as initial values.
- EM-LS: The EM algorithm using parameters from the least squares solution as initial values (The LS solution was obtained using UQRLS, and the work was counted as one EM iteration).

Using ECM-prev and EM-prev, the null hypothesis estimates were used as initial values for the first position on each chromosome, since there the test statistic function is discontinuous.

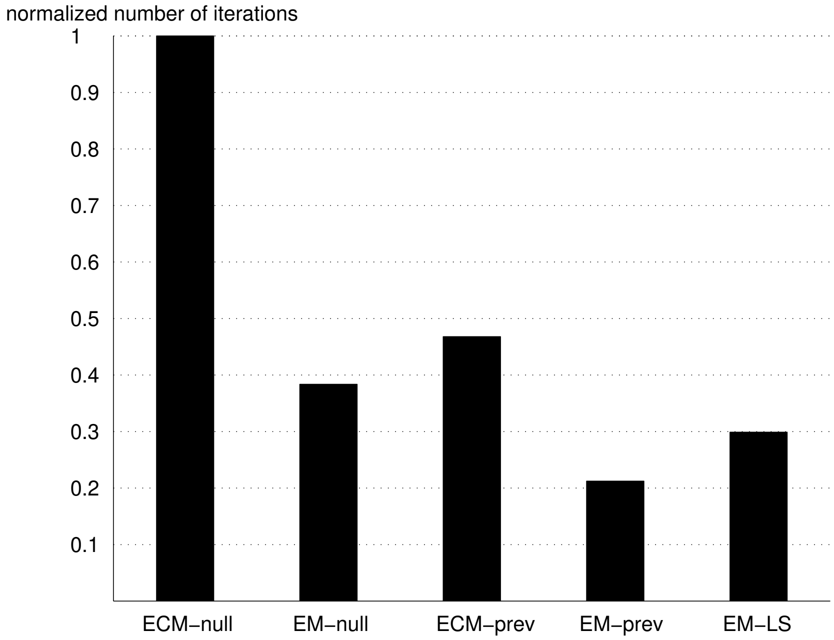

FIG. 9.2. *Normalized numbers of iterations needed for a full genomic scan*

Figure 9.2 shows the normalized numbers of iterations for the different ML solver versions. The computational work needed for one iteration is very similar for the two algorithms, the EM algorithm requiring slightly fewer arithmetic operations per iteration than the ECM algorithm. From the results it is clear that the convergence of the EM algorithm is more than twice as fast as the convergence of the ECM version. This is not surprising, since the ECM algorithm is obtained by introducing approximations in the EM algorithm. Also, the results show that much is gained by using the previous parameter estimates as initial values in the next iteration. This was also expected, since the test statistic is mostly a continuous function. There is only need for a restart at discontinuities, i.e. at new chromosomes and when a marker cofactor enters or leaves the model. In this particular experiment there were no cofactors included, so in a general case there would have to be a restart more often. Then the LS solution should be used rather than the null hypothesis, as the LS solution is inexpensive to obtain if using UQRLS, and is closer to the true ML solution.

The experiment above only counts the number of iterations and approximate number of floating number operations needed for them, assuming matrix factorizations are already available. For this problem both iterative methods only requires a single factorization, since there are no cofactors included in the model and they all exploit the constant X_f. If marker cofactors are included in the model, the comparison will be even more in favor of our new formulation of the EM algorithm, since we use updating instead of full refactorization of X_{fc}. The overhead in checking when an update is required is small compared to the gain in computational effort.

10. Discussion. Using our new algorithms, the gain in computational efficiency is enough to make previously intractable QTL analyses realistic, for example permutation testing for multiple QTL scans. A typical permutation test includes 2000 exhaustive two-dimensional searches.

In Table 10.1 the timings for permutation tests are presented. Solving the LR

Routine	pig, $g=2$	chicken, $g=2$	pig, $g=8$	chicken, $g=8$
G02DAF	1.2 days	33 days	2.6 years	74 years
SQRDC/SQRSL	46 min	5.6h	27 days	220 days
UQRLS	3.7 min	40 min	6.8 days	76 days

FIG. 10.1. *Approximate CPU times for kernel problem in 2000 exhaustive searches using the LR model*

kernel problems corresponding to 2000 exhaustive searches in two dimensions for the real pig data set would take just under 7 days with UQRLS, 27 days with SQRDC/SQRSL and 2.6 years with G02DAF. Hence, the amount of time needed to solve the kernel problem makes a full permutation test with exhaustive search in two dimensions impossible if using G02DAF and unattractive with SQRDC/SQRSL. With UQRLS however, the CPU time needed for the kernel problem is reasonable. Since the 2000 searches are completely independent, the problem is also very easily parallelized. With a careful implementation of the analysis program and possibly using parallelization, it is realistic to routinely perform two-dimensional permutation testing.

For accurate detection and location of multiple interacting QTL in large data sets, using IM-based multidimensional search schemes, more elaborate optimization methods will eventually be required. Some work has recently been performed in this area. In [Carlborg et al., 2000] a genetic optimization algorithm is used for performing a true multidimensional search. In [Kao and Zeng, 1997, Kao et al., 1999, Zeng et al., 1999], the Multiple Interval Mapping (MIM) technique is developed, where the optimization is performed over selected regions in the search space. Also, methods based on sequences of one-dimensional QTL scans have been developed [Jansen and Stam, 1994, Broman, 1997], but such schemes may have problems in detecting interacting QTL where the individual QTLs have small independent effects. With a better optimization algorithm than exhaustive search, models with genome scans in three or more dimensions could probably be used without introducing unrealistic computational times, even when combined with permutation testing.

Finally, it should be noted that if the kernel routines are replaced by highly efficient versions, other parts of the QTL analysis program might turn out to be poorly implemented and too time consuming. To construct an efficient program for accurate multiple QTL scans, other routines might also need optimization efforts.

References.

Leif Andersson. Genetic dissection of phenotypic diversity in farm animals. *Nature Reviews Genetics*, 2:130–138, February 2001.

Leif Andersson, Chris Haley, Hans Ellegren, Sarah Knott, M Johansson, Kjell Andersson, Lena Andersson-Eklund, I Edfors-Lilja, M Fredholm, and I Hansson. Genetic mapping of quantitative trait loci for growth and fatness in pigs. *Science*, 263:1771–1774, 1994.

Christopher Basten, Bruce Weir, and Zhao-Bang Zeng. *QTL Cartographer, Version 1.15.* Department of Statistics, North Carolina State University, Raleigh, NC, 2001.

Åke Björck. *Numerical Methods for Least Squares Problems.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 1996.

Karl Broman. *Identifying quantitative trait loci in experimental crosses.* PhD thesis, Deparment of Statistics, University of California, Berkeley, 1997.

Örjan Carlborg, Leif Andersson, and B Kinghorn. The use of a genetic algorithm for simultaneous mapping of multiple interacting quantitative trait loci. *Genetics*, 155:2003–2010, 2000.

Gary Churchill and Rebecca Doerge. Empirical threshold values for quantitative trait mapping. *Genetics*, 138:963–971, 1994.

Rebecca Doerge. Mapping and analysis of quantitative trait loci in experimental populations. *Nature Reviews Genetics*, 3:43–52, January 2002.

Chris Haley and Sarah Knott. A simple regression method for mapping quantitative trait loci in line crosses using flanking markers. *Heredity*, 69:315–324, 1992.

Chris Haley, Sarah Knott, and Jean-Michel Elsen. Mapping quantitative trait loci in crosses between outbred lines using least squares. *Genetics*, 136:1195–1207, March 1994.

Ritsert Jansen. A general mixture model for mapping quantitative trait loci by using molecular markers. *Theoretical and Applied Genetics*, 85:252–260, 1992.

Ritsert Jansen. Interval mapping of multiple quantitative trait loci. *Genetics*, 135: 205–211, September 1993.

Ritsert Jansen and Piet Stam. High resolution of quantitative traits into multiple loci via interval mapping. *Genetics*, 136:1447–1455, April 1994.

Chen-Hung Kao and Zhao-Bang Zeng. General formulae for obtaining the MLEs and the asymptotic variance-covariance matrix in mapping quantitative trait loci when using the EM algorithm. *Biometrics*, 53:653–665, June 1997.

Chen-Hung Kao, Zhao-Bang Zeng, and Robert Teasdale. Multiple interval mapping for quantitative trait loci. *Genetics*, 152:1203–1216, July 1999.

Eric Lander and David Botstein. Mapping mendelian factors underlying quantitative traits using RFLP linkage maps. *Genetics*, 121:185–199, January 1989.

Trudy Mackay. Quantitative trait loci in Drosophila. *Nature Reviews Genetics*, 2: 11–21, January 2001.

Octavio Martinez and Robert Curnow. Estimating the locations and the sizes of effects of quantitative trait loci using flanking markers. *Theoretical and Applied Genetics*, 85:480–488, 1992.

Xiao-Li Meng and Donald Rubin. Maximum likelihood estimation via the ECM algorithm: A general framework. *Biometrika*, 80:267–278, 1993.

Dianne O'Leary. Robust regression computation using iteratively reweighted least squares. *SIAM Journal of Matrix Analysis and Applications*, 11:466–480, July 1990.

Zhao-Bang Zeng. Theoretical basis for separation of multiple linked gene effects in mapping quantitative trait loci. *Proceedings of the National Academy of Science of the USA*, 90:10972–10976, December 1993.

Zhao-Bang Zeng. Precision mapping of quantitative trait loci. *Genetics*, 136:1457–1468, 1994.

Zhao-Bang Zeng, Chen-Hung Kao, and Christopher Basten. Estimating the genetic architecture of quantitative traits. *Genetical Research*, 74:279–289, 1999.