

Fit and exploration of multiple-QTL models

The majority of efforts for QTL mapping have used a hypothesis testing approach. For example, in single-QTL analyses (Chap. 4), one considers each genomic position, one at a time, and asks the question, “Is there a QTL here?” A primary focus is on the adjustment for the number of tests (i.e., for the scan across the genome), to control the rate of false positive declarations of linkage.

A two-dimensional, two-QTL genome scan (Chap. 8) largely gets around the principal weaknesses of single-QTL analysis (of separating linked QTL and identifying interacting QTL), but again this is a hypothesis testing approach. One considers a pair of putative QTL and asks, “Are there QTL here and here?”

These approaches work surprisingly well, largely due to the independent assortment of chromosomes, but they are formally correct only if a phenotype is affected by no more than two QTL. However, we expect complex traits to be affected by multiple loci. The single- and two-QTL analysis methods indicate individual pieces of the complex genetic architecture that underlies the phenotype. To properly weigh the evidence for the QTL, one should ultimately bring these pieces together in a single coherent model.

The primary goal of QTL mapping, to identify the set of loci that contribute to the phenotypic variation, thus does not fit well into the sequence of yes-or-no questions that forms the hypothesis testing framework. Rather, QTL mapping is best viewed as a *model selection* or *variable selection* problem: what set of loci (and QTL $\times$ QTL interactions) are best supported by the data?

In this chapter, we describe the key aspects of the model selection approach to multiple QTL mapping. We focus primarily on classical approaches to the problem, though we briefly describe approaches using Bayesian statistical inference, primarily to emphasize the advantages and disadvantages of the Bayesian approach. We further describe the R/*qtl* functions for fitting and exploring multiple QTL models.

We will focus on the case of a continuously varying quantitative trait with normally distributed residual variation. The ideas may be extended for use with alternate types of phenotypes, including binary traits, censored survival times, and phenotype distributions exhibiting a spike, but we will not pursue such extensions here.

9.1 Model selection

As discussed in Chap. 1, the QTL mapping problem can be split into two parts: the missing data problem and the model selection problem. The missing data problem arises from the fact that, while QTL may reside anywhere in the genome, we observe individuals' genotypes only at discrete landmarks, the genetic markers. We thus use the observed marker genotype data to infer the genotypes at all intervening locations. This problem, while it remains an annoyance, has been well-solved in a number of ways (including maximum likelihood via the EM algorithm, multiple imputation and Haley–Knott regression); it concerns the fit of a QTL model in the case that genotypes at the putative QTL are not observed.

The more important aspect of QTL mapping is the model selection problem. Imagine one could observe complete genotype data on each individual. For example, in a backcross, imagine that we knew, at every polymorphism at which the parental strains differ, which individuals were homozygous and which were heterozygous. We are still confronted with the difficult problem of identifying the subset of loci that truly influence the phenotype, and of how they combine together to produce the phenotype. By *model*, we mean a defined set of genetic loci (and, potentially, QTL $\times$ QTL interactions), and in *model selection*, we seek to identify such a set of QTL that are well supported by the observed data.

As with hypothesis testing, in selecting a set of loci that are viewed to influence the phenotype, one can make two types of errors: we may miss important loci (false negatives), and we may include extraneous loci (false positives). Unlike hypothesis testing, here we can make both errors at the same time.

QTL mapping projects are initiated for a variety of different purposes. In evolutionary studies, one may be particularly interested in the distribution of QTL effects and of the relative contribution of epistatic effects. In agriculture, one may be primarily interested in obtaining information that will guide future selection experiments. In biomedical experiments, the ultimate goal is to identify the gene or genes (and perhaps even the individual mutations) that contribute to phenotypic differences, in order to gain a better understanding of the mechanism of disease and to identify potential targets for therapeutic drugs.

We focus primarily on QTL mapping for biomedical research. In this context, we are particularly concerned with avoiding false positives, so that

downstream experiments to fine-map and ultimately identify the causal gene or genes will not be conducted in vain. We thus view the goal of model selection for QTL mapping to be to control the rate of inclusion of extraneous loci, while identifying as many true QTL as possible.

Knowledge of epistatic interactions can be important, as they may influence the downstream fine-mapping experiments. For example, experiments to dissect a QTL often begin with the construction of a congenic strain, in which a genomic segment from one strain is introgressed into another strain, creating an inbred strain that is homozygous A everywhere except for one genomic segment, where it is homozygous B. Information on epistatic interactions may indicate that one type of congenic (in which a segment from the B strain is isolated within the A background) may have no effect, while the other (in which the A segment is isolated within the B background) may have a large effect.

However, we are generally not so concerned with precisely identifying the interactions between QTL, but rather we seek to identify the major players and want to ensure that the presence of interactions does not prevent us from identifying important loci. False inference of the presence of an interaction that does not exist (or of the absence of an interaction that does exist) is not so bad as missing an important locus or including an extraneous one.

While there is a large literature on the problem of model selection in linear regression, there are some important differences in the QTL mapping problem. First, model selection in regression has often focused on the minimization of prediction error: one expects that all covariates have some effect on the outcome, but by eliminating some covariates from the prediction model, one allows some bias but eliminates a great deal of variance, and so ultimately obtains improved predictions. In QTL mapping, however, we are not interested in prediction, but rather in identifying the important elements of the model. Second, in QTL mapping we are confronted with a continuum of potential covariates (putative QTL locations along the chromosomes), though with a great deal of missing covariate information. Related to this point, we do not expect to identify the exact site of a QTL, but want to pick loci that are not far from the true QTL. Finally, the correlation among the covariates has a special structure in QTL mapping: from chromosome to chromosome, the genotypes are independent. Moreover, along a chromosome, they display a very simple correlation structure. In the case of no crossover interference, genotypes along a chromosome form a Markov chain, so that, given the genotypes at a particular locus, the genotypes at sites to the left of the locus are conditionally independent of the genotypes at the sites to the right of the locus. While this conditional independence does not hold in the presence of crossover interference, the correlation structure remains relatively simple, and in either case it is effectively known. With this simple correlation structure among the potential covariates, procedures that have been found to perform poorly in other contexts may actually work quite well for the QTL mapping problem.

In defining a model selection procedure for QTL mapping, one must make four distinct choices. First, one must choose a *class of models*, such as strictly additive QTL models, models that allow pairwise interactions between QTL, or models that allow interactions of any order. Second, one must define a method for *model fit*; that is, for a particular QTL model, with QTL in fixed positions and a defined set of epistatic interactions, how will the missing genotype data be overcome to define the fit of the model? Third, we need a method for *model search*. The space of models will be far too large for the models to be considered exhaustively; we need a procedure for exploring the model space to identify good ones, recognizing that we will only be able to consider small slices through the model space. Finally, and most importantly, we need a method for *model comparison*. In forming a model comparison procedure, it can be useful to imagine that we could fit all possible models; which model would we then choose? Larger models will provide an improved fit, and so we must balance quality of fit with model complexity. How much of an improvement in fit should be required in order to incorporate an additional QTL into the model?

We will discuss these four aspects of a model selection procedure in the following subsections. We will then bring these separate streams of thought back together, to emphasize the tradeoffs accompanied by any set of choices.

9.1.1 Class of models

The first choice, in forming a model selection procedure, is of the class of models. The simplest class is that of strictly additive QTL models.

$$y = \mu + \sum_j \beta_j q_j + \epsilon$$

With such an additive model, the effect of a QTL is constant, independent of the genotypes at other loci.

One may also wish to include models with pairwise interactions.

$$y = \mu + \sum_j \beta_j q_j + \sum_{j,k} \beta_{jk} q_j q_k + \epsilon$$

In doing so, we generally enforce a hierarchy, under which the inclusion of a QTL $\times$ QTL interaction requires the inclusion of main effects for each QTL. One advantage of such a hierarchical structure is that the coding of QTL effects becomes immaterial. The model space is also restricted. The enforcement of such a hierarchy is similar to the choice, in single-QTL analysis in an intercross, to always include both degrees of freedom at any QTL, and so not explore strictly dominant or recessive models, or models in which the two alleles at a locus are strictly additive. The hierarchical structure can make it difficult to identify loci with important interactions but limited marginal

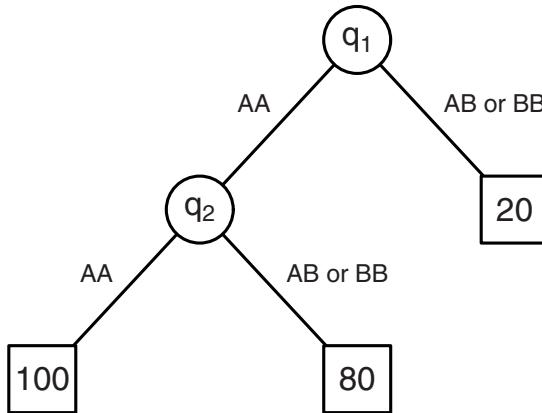


Figure 9.1. An example of a regression tree. This is a type of decision tree that describes a QTL model. Individuals with genotype AB or BB at QTL 1 have average phenotype 20, individuals with genotype AA at both QTL 1 and QTL 2 have average phenotype 100, and individuals with genotype AA at QTL 1 and either AB or BB at QTL 2 have average phenotype 80.

effects since, in an intercross, one brings in all eight degrees of freedom for a pair of interacting loci.

One can, of course, expand the model space to include higher-order interactions, including three-way interactions (in which the strength of interaction between two loci depends on the genotype at a third locus), four-way interactions, and so on. Evidence for higher-order interactions has been observed, particularly for the case of a $\text{QTL} \times \text{QTL} \times \text{covariate}$ interaction, but this expansion of the model space can come at a severe price, in the form of a greater false positive rate or a decrease in power (if the false positive rate is controlled). In an intercross, it is assuredly difficult to distinguish a true three-way interaction from three QTL with only pairwise interactions, particularly because of the 1:2:1 segregation of alleles at a locus, which makes many three-locus genotypes quite rare and so potentially unobserved in a cross of ordinary size. Thus, while we admit that an investigation of three-way interactions in a backcross may be fruitful, we believe that it is likely best to focus, in an intercross, on only pairwise interactions.

Related to the class of models is what might be called the “form of expression” of a model. Here we have in mind regression trees, such as that in Fig. 9.1. This is a form of decision tree, in which terminal nodes define the average phenotype for individuals with a particular multilocus QTL genotype.

The class of regression trees is equivalent to the class of ordinary linear regression models that allow all orders of interactions, but the form in which such models are expressed is radically different. The simple regression tree in Fig. 9.1 would be cumbersome to write with a linear model, but then an additive QTL model would make a more complex regression tree.

Regression trees have not been used much in genetics, but the natural way in which complex interactions can be expressed through regression trees does make them intriguing. However, the enormous model space is forbidding.

One final point: it is sometimes necessary to restrict the model space in order to avoid artifacts or to speed up computations. For example, we often assume that any interval between markers contains at most one QTL. One may further require that any two QTL are separated by at least two markers. This is done because a model with two tightly linked QTL in repulsion (that is, having effects of opposite signs) can occasionally provide a spuriously good fit. This is particularly the case when there are a few individuals with outlying phenotype values. It is related to difficulties along the diagonal in a two-dimensional, two-QTL genome scan.

Another way to restrict model space is to assume that putative QTL must reside at the marker locations. In the case of high-density genotype data, this can be a reasonable approximation, and can allow great speed-up in the computations, particularly if the marker genotype data are relatively complete. Related to this is the density of the grid (i.e., the step size) in standard interval mapping: a more coarse grid gives less precise results but the calculations are much faster.

9.1.2 Model fit

A central part of any QTL mapping procedure is a method for fitting QTL models. In the case of complete genotype data at the putative QTL, one would use linear regression. (Recall that we are focusing, in this chapter, on continuously varying quantitative traits with normally distributed residual variation.) The fit of a model could be described by the residual sum of squares, $RSS(\gamma) = \sum_i (y_i - \hat{y}_i)^2$, where γ denotes a model, y_i is the phenotype for individual i and $\hat{y}_i$ is the corresponding fitted value under the model. Equivalently, one could use the LOD score, $LOD(\gamma) = (n/2) \log_{10}[RSS_0/RSS(\gamma)]$, where $RSS_0 = \sum_i (y_i - \bar{y})^2$ is the residual sum of squares for the null model. Again, the good models are those for which $RSS(\gamma)$ is small and so $LOD(\gamma)$ is large.

In general, one will have no genotype data at the putative QTL, and will need to use data at linked markers to infer the QTL genotypes. This is the missing data problem, which we have now rolled into the model selection problem.

The various methods for fitting a single-QTL model (described in Chap. 4) are all readily extended to the case of a multiple-QTL model, though with different degrees of difficulty. The relative advantages and disadvantages of the different methods largely remain, though the multiple imputation approach is finally given an opportunity to shine in the context of multiple-QTL models.

The simplest method is Haley–Knott regression. We replace the indicators for QTL genotypes with their expected values, given the available marker data, and so perform linear regression of the phenotype on the multiple QTL

genotype *probabilities*. The great advantage of this approach is computational speed: we perform a single regression for each QTL model of interest. The disadvantage of the approach is that it again fails to make complete use of the available genotype data, and it can give spuriously large evidence for linkage in the presence of selectively genotyped data. But in the case of relatively dense markers and relatively complete marker genotype data, Haley–Knott regression is clearly the preferred method.

The multiple imputation approach extends to the case of multiple-QTL models without modification, and while with single- and two-QTL models the computation time for the imputations and for the fixed set of linear regressions to be performed at each putative QTL or QTL pair weakened the value of the approach, the imputations are performed just once, and so in the exploration of multiple-QTL models, the multiple imputation approach is quite valuable.

The extended Haley–Knott method can be applied for the case of multiple-QTL models, but it has not yet been implemented in software. While we expect that its speed and robustness properties will carry over to the case of multiple-QTL models, this remains to be tested.

The extension of standard interval mapping (maximum likelihood via an EM algorithm) to the case of multiple QTL models is called multiple interval mapping (MIM). The theory is straightforward, though there are some practical difficulties. We seek to fit a model of the form

$$y = X\beta + \epsilon$$

where X is a matrix of QTL genotype indicators (and possibly indicators for QTL $\times$ QTL interactions), which are not observed. The EM algorithm to derive maximum likelihood estimates under this model involves first calculating the expected value of the X and $X'X$ matrices, given the observed phenotype and marker genotype data, and given current estimates for the parameters β and σ :

$$\begin{aligned} Z^{(s)} &= E(X|y, M, \hat{\beta}^{(s-1)}, \hat{\sigma}^{(s-1)}) \\ W^{(s)} &= E(X'X|y, M, \hat{\beta}^{(s-1)}, \hat{\sigma}^{(s-1)}) \end{aligned}$$

With $Z^{(s)}$ and $W^{(s)}$ in hand, the M-step of the EM algorithm is relatively easy. Updated estimates of the model parameters, β , are obtained by solving the normal equations, with X and $X'X$ replaced by their expected values:

$$W^{(s)}\hat{\beta}^{(s)} = [Z^{(s)}]'y$$

The updated estimate of the residual SD is obtained as follows.

$$\hat{\sigma}^{(s)} = \sqrt{(y'y - y'Z^{(s)}\hat{\beta}^{(s)})/n}$$

It is the E-step (calculating Z and W) that is difficult. Part of the problem, particularly for calculating W , is one of bookkeeping: handling arbitrary

models with an arbitrary number of QTL and an arbitrary set of epistatic interactions. But with that aside, there remains the difficulty, in the case of p QTL, of summing over the 2^p possible multilocus QTL genotypes in a backcross (or 3^p in an intercross). This must be done for each individual and at each iteration of the EM algorithm, and so for models with a large number of QTL, the computational demands can be great. One technique that has been used is to trim multilocus QTL genotypes that have low prior probability, say < 0.001 . By prior probability, we mean the probability of a multilocus genotype pattern conditional on the available marker genotype data (but not conditional on the observed phenotypes and the current estimates of the model parameters). This trimming can greatly reduce the number of possible multilocus QTL genotypes; the result is an approximation to the true likelihood, but it can give a great gain in computational speed.

With any of the four methods for fitting a QTL model (which we will denote γ), we will characterize model fit by the LOD score: the $\log_{10}$ likelihood ratio comparing the model γ to the null model (denoted $\emptyset$).

9.1.3 Model search

The model space is far too large for all models to be considered exhaustively. If we restrict ourselves to the marker positions and to additive QTL models, with 100 markers there are $2^{100} \approx 10^{30}$ models. If we assume that there are no more than 25 QTL but allow pairwise interactions between QTL, there are about 10^{113} possible models.

The enormous size of the model space thus requires a search algorithm, so that we may identify the good models even though we may visit only a minuscule proportion of the possible models. Let us focus initially on additive QTL models.

The simplest algorithm is forward selection. We begin by considering all possible single-QTL models, and we pick the best of these (i.e., that giving the largest LOD score), say $\gamma_1 = \{\lambda_1\}$. We then consider all two-QTL models that include our first QTL, and pick the locus that gives the greatest increase in LOD, to obtain $\gamma_2 = \{\lambda_1, \lambda_2\}$. Next, we consider all three-QTL models that include the first two QTL identified, to obtain $\gamma_3 = \{\lambda_1, \lambda_2, \lambda_3\}$. We continue in this way, building a nested sequence of models of increasing size. In the case of 100 putative QTL locations and additive QTL models, we would visit a set of 5051 models (out of the total of $\sim 10^{30}$ models).

Forward selection has a rather poor reputation in the statistical literature, as once a term enters the model, it is never removed. And in many cases one will find that, even with an extremely large data set, a single covariate that does not belong in the model may mimic a pair of covariates that do belong in a way that the single false covariate will always enter the model before either of the pair of covariates that truly belong. However, with the simple correlation structure among genotypes at putative QTL locations, one can

show that in the case of additive QTL models, this problem will not occur, at least for very large data sets.

The reverse of forward selection is backward elimination. One begins with a large model (perhaps obtained by forward selection), and drops the covariates from the model one at a time, at each step dropping the covariate that results in the smallest decrease in the LOD score. Thus we construct a nested sequence of models of decreasing size.

One may similarly construct stepwise algorithms that include some forward and some backward steps. There are also numerous randomized algorithms (including Markov chain Monte Carlo, simulated annealing, and genetic algorithms) that take a random walk through model space, and generally do not require a strict improvement in the LOD score at each step.

In the case that pairwise interactions between loci are to be considered, a forward selection algorithm would also consider the addition of an interaction between the loci in the current model, and of a new locus that interacts with one of the loci in the current model. One may also wish to include two-step jumps, in which at each step of forward selection, a two-dimensional scan is performed, and so two novel QTL (whether additive or interacting) may be brought into the model together. This would be particularly useful for identifying a pair of interacting loci that show no marginal effects, or for a pair of loci linked in repulsion (having effects of opposite signs); in both of these cases, the two loci may not appear interesting individually, and so would likely be missed by a single-step forward selection algorithm.

It can be useful, in a search algorithm for multiple QTL mapping, to consider a refinement of the QTL locations at each step of a stepwise algorithm, as the likely position for a QTL may change as additional QTL are brought into the model. A simple but effective algorithm is formed by refining the location of each QTL in the model one at a time, keeping the locations of all other QTL fixed. The QTL are not moved between chromosomes, and the order of the QTL within a chromosome is not altered, and so in refining the position of a given QTL, one scans across its chromosome, or along the interval defined by the positions of the flanking QTL, if there are multiple QTL on the chromosome. We would generally consider the QTL in a random order, and would iterate the refinement process until no further change in QTL position is observed.

In general, we would prefer the most exhaustive search possible, though more extensive searches are accompanied by greater computation time and may give no improvement, in that the optimal model might be found with a simpler and less computationally intensive search.

Our preferred approach is to use forward selection to a model of moderate size (say 10 or 20 QTL), followed by backward elimination all the way to the null model. The chosen model would be that which optimized the model comparison criterion (see the next section), among all models visited. A simple and effective search algorithm, allowing for the possibility of pairwise interactions

(implemented in the `stepwiseqtl` function in R/`qtl`; see Sec. 9.3.7), is the following.

1. Start by performing a single-QTL genome scan, and choose the position giving the largest LOD score.
2. With a fixed QTL model in hand:
 - a. Scan for an additional additive QTL.
 - b. For each QTL in the current model, scan for an additional interacting QTL.
 - c. If there are ≥ 2 QTL in the current model, consider adding one of the possible pairwise interactions.
 - d. Optionally perform a two-dimensional, two-QTL scan, seeking to add a pair of novel QTL, either additive or interacting.
 - e. Step to the model that gives the largest value for the model comparison criterion, among those considered at the current step.
3. Refine the locations of the QTL in the current model.
4. Repeat steps 2 and 3 up to a model with some predetermined number of loci.
5. Perform backward elimination, all the way back to the null model. At each step, consider dropping one of the current main effects or interactions; move to the model that maximizes the model comparison criterion, among those considered at this step. Follow this with a refinement of the locations of the QTL.
6. Finally, choose the model having the largest model comparison criterion, among all models visited.

In this forward/backward algorithm, it is likely best to build up to an overly large model and then prune it back. Note that there is no “stopping rule;” the chosen model is that which optimizes the model comparison criterion, among all models visited. The search can be time consuming, particularly if a two-dimensional scan is performed at each forward step. Such two-dimensional scans may be useful for identifying QTL linked in repulsion (having effects of opposite sign) or interacting QTL with limited marginal effects, but our limited experience suggests that they are not necessary; important linked or interacting QTL pairs can be picked up in the forward selection to a large model, and will be retained in the backward elimination phase.

9.1.4 Model comparison

By far the most important aspect of the QTL mapping problem is the criterion for choosing among possible QTL models. For models of the same size, we would choose that with the largest LOD score (i.e., that with the maximum likelihood). However, the LOD score can always be increased by adding an additional QTL to a model. Thus, we must seek some balance between the fit of a model (represented by the LOD score), and the complexity of the

model (the number of QTL and interactions). Our goal is to define a criterion that will appropriately balance the false positive and false negative rates. In particular, we seek to control the false positive rate (that is, the rate of inclusion of extraneous loci) at some chosen level and then identify as many true QTL as possible.

We will not attempt to discuss this issue in detail, but rather will focus on a single approach that we consider most appropriate for the QTL mapping problem in biomedical research.

Much of the literature on model selection criteria has focused on minimizing prediction error; while these approaches may also work well for identifying the important players (which is our goal), in general they tend to produce overly large models, as the inclusion of a few extraneous covariates will not weaken predictions so much as missing a few important covariates. Also, much work has focused on the asymptotic behavior of the criteria (that is, the case of an infinitely large sample), but in the large sample case, some important features of the data (such as the number of potential covariates) are no longer seen to matter, and so the asymptotic behavior of a procedure can be a poor guide to its small sample performance.

Let us begin by considering additive QTL models. We prefer to use a penalized LOD score.

$$\text{pLOD}_a(\gamma) = \text{LOD}(\gamma) - T|\gamma| \quad (9.1)$$

where γ denotes a model, $|\gamma|$ is the number of QTL in the model, and T is a penalty on the size of the model.

We seek a penalty T that will control the rate of inclusion of extraneous loci at some chosen rate (e.g., 5%). We would like the rate to be controlled no matter the true model, but consider, in particular, the case of the null model, $\emptyset$, and that we perform a single-QTL genome scan. The penalized LOD for the null model is $\text{pLOD}_a(\emptyset) = 0$, since the LOD score is defined relative to the null model and $|\emptyset| = 0$. The penalized LOD for a single-QTL model is $\text{pLOD}_a(\{\lambda\}) = \text{LOD}(\lambda) - T$, where $\{\lambda\}$ is the model with a single QTL at position λ and $\text{LOD}(\lambda)$ is the LOD score from a single-QTL scan at position λ .

We would choose the model $\{\lambda\}$ over the null model when $\text{LOD}(\lambda) > T$. This suggests setting T to be the 95th percentile of the genome-wide maximum LOD score under the global null hypothesis (of no QTL anywhere), which may be estimated via a permutation test. This extends the usual procedure for single-QTL analysis to a criterion useful for choosing among additive QTL models of any size. The choice of penalty guarantees the control of the false positive rate at the target level only for the case that the truth is the null model and that the search considers models with no more than one QTL. But computer simulation experiments indicate that the false positive rate is maintained reasonably well for larger models and for more extensive searches.

To extend this approach to the case of pairwise interactions among QTL, we could apply separate penalties on the main effects and the interactions, T_m and T_i :

$$\text{pLOD}(\gamma) = \text{LOD}(\gamma) - T_m |\gamma|_m - T_i |\gamma|_i \quad (9.2)$$

where $|\gamma|_m$ is the number of QTL in the model and $|\gamma|_i$ is the number of interaction terms. We will focus on the case that a hierarchical structure is imposed on the model, with the inclusion of an interaction requiring the inclusion of both of the corresponding main effects.

We take T_m to be the significance threshold from a single-QTL genome scan, as before. With this choice, the extended criterion in equation (9.2) is equivalent to that in equation (9.1) if one restricts the search to additive QTL models.

We are left with the choice of the penalty on interaction terms. We can apply the same sort of logic that led us to the penalty on main effects. Imagine there are two additive QTL, and that we perform a two-dimensional, two-QTL genome scan. We could then define the interaction penalty as follows:

$$T_i^H = 95\text{th percentile of } [\max_{\lambda_1, \lambda_2} \text{LOD}_f(\lambda_1, \lambda_2) - \max_{\lambda_1, \lambda_2} \text{LOD}_a(\lambda_1, \lambda_2)] \quad (9.3)$$

where LOD_f and LOD_a are the LOD scores for full and additive two-QTL models, respectively (see Chap. 8). With this choice, we would control the rate of inclusion of an extraneous interaction at our target rate. While ideally one would determine the distribution of $\max \text{LOD}_f - \max \text{LOD}_a$ in the presence of two additive QTL, it is not likely to be too different from the distribution under the null hypothesis of no QTL, and so we may use a permutation test in a two-dimensional genome scan to estimate T_i . We call this choice the heavy penalty, T_i^H , as we will introduce a light penalty next.

If the logic in the previous paragraph is reasonable, why not extend it? Imagine that there is a single QTL, and that we perform a two-dimensional, two-QTL scan. To control the rate of inclusion of a false interacting QTL, we define a light interaction penalty, T_i^L , through the equation

$$T_m + T_i^L = 95\text{th percentile of } [\max_{\lambda_1, \lambda_2} \text{LOD}_f(\lambda_1, \lambda_2) - \max_{\lambda} \text{LOD}_1(\lambda)] \quad (9.4)$$

where LOD_f is the LOD score for the full model, with two interacting QTL, and LOD_1 is the LOD score for a single-QTL model. With the interaction penalty defined in this way, we ensure that, in the case of a single QTL,

Table 9.1. Estimated penalties, derived by computer simulation, for a genome modeled after the mouse and with markers at a 10 cM spacing.

Penalty	Cross	
	Backcross	Intercross
T_m	2.69	3.52
T_i^H	2.62	4.28
T_i^L	1.19	2.69

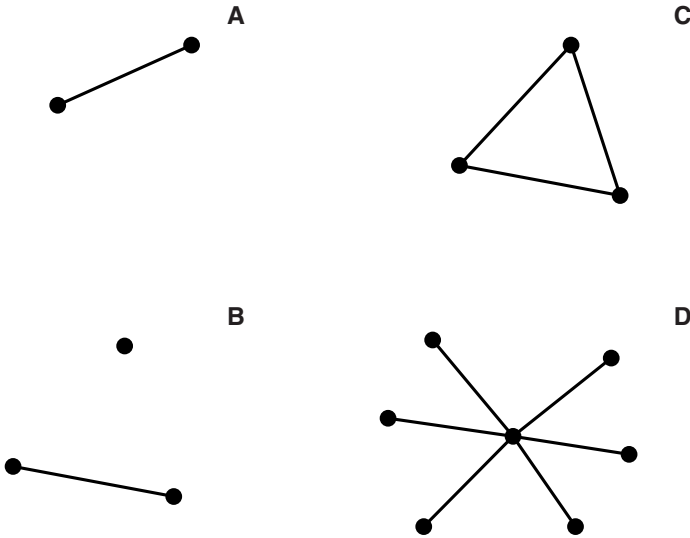


Figure 9.2. Graphical depictions of QTL models, with nodes corresponding to QTL and edges indicating interactions. **A:** Two interacting QTL. **B:** Three QTL, of which two interact. **C:** Three QTL with all pairs interacting. **D:** A seven-QTL model, with one QTL exhibiting pairwise interactions with each of the other QTL.

the rate of inclusion of a second, interacting QTL is controlled at the chosen rate. In the presence of two additive QTL, the interaction term will be falsely included at a much higher rate, but we are less concerned about such errors and seek principally to control the rate of false positive QTL.

In thinking about these penalties, and the difference between the heavy and light interaction penalties, it is good to have some particular numbers in mind. Estimated penalties, derived via computer simulation of backcrosses and intercrosses with genomes modeled after that of the mouse and with markers at a 10 cM spacing, are shown in Table 9.1. The light interaction penalty is *much* smaller than the heavy interaction penalty.

It is useful, in the consideration of QTL models with possible pairwise interactions (and with our imposed hierarchical structure that requires the inclusion of both main effects along with any interaction term), to depict such models as graphs, with nodes (i.e., dots) corresponding to QTL and edges (i.e., line segments between QTL) indicating interactions. Consider Fig. 9.2. Model A has two interacting QTL. Model B has three QTL, of which two interact. Model C has three QTL with all possible pairwise interactions. In model D, there are seven QTL, with one QTL interacting with each of the other six.

In the analysis of QTL data with the exclusive use of the light interaction penalty, T_i^L , we often identified models like that in Fig. 9.2D, with a single QTL interacting with many others. This seems implausible. Computer

simulation experiments confirmed this sentiment: in the case of an additive QTL model with many QTL, the exclusive use of the light interaction penalty will often result in the identification of an extraneous locus, interacting with many or all of the true loci. The problem is that, with the light interaction penalty, we imagine that the truth is a “dot” (one QTL), and we control the rate of inclusion of an extraneous “pin” (an additional, interacting QTL). But in the presence of multiple QTL, an extraneous locus can purchase its entry into the chosen model via numerous lightly weighted interactions: we apply too small a penalty to multipronged pins.

An *ad hoc* modification of our penalized LOD score can eliminate this undesired behavior. Consider the graph corresponding to a particular model. (For example, imagine that the four parts of Fig. 9.2 formed a single model, with a total of 15 QTL and 11 pairwise interactions.) For each connected component of the graph (that is, for each cluster of interacting QTL), we apply a single light interaction penalty and give all other interactions the heavy penalty. (For the model formed from all parts of Fig. 9.2, there would be 15 main effect penalties, T_m , four light interaction penalties, T_i^L , and seven heavy interaction penalties, T_i^H .) This approach serves as a compromise between using only heavy interaction penalties (which would result in low power to detect interacting loci) and only light interaction penalties (which can lead to a high rate of extraneous loci).

The use of this model comparison criterion is guaranteed to control the rate of inclusion of extraneous loci only in the presence of one or two QTL, and with the model search not extending beyond two QTL. Moreover, we should expect that the inclusion of extraneous loci will increase with the size of the model, as in the presence of many QTL, there are many more ways to incorporate an additional extraneous interacting locus (i.e., to attach an extraneous “pin” to the model). But such behavior can probably not be eliminated and is not unreasonable; having one extraneous locus among nine identified QTL is not so bad as one extraneous locus among three identified QTL.

9.1.5 Further discussion

A model selection procedure for QTL mapping has four parts: a choice of the class of models, a method for model fit, a method for model search, and a criterion for comparing models. We prefer to keep these pieces distinct. In particular, we are opposed to the use of “stopping rules,” which combine model search and model comparison; rather, a model comparison criterion should be chosen, imagining one could visit all possible models in the class under consideration, and the aim of the model search algorithm should be to optimize this criterion.

Some strategies seek to restrict the search over models in order to increase power. The argument is that, if a smaller number of models are visited, the adjustment for the range of models visited will be less severe and so a more permissive model comparison criterion may be used. We prefer instead to

restrict the class of models (e.g., focusing solely on additive QTL models). If the truth is approximately additive, greater power to detect QTL (and a reduced false positive rate) can then be achieved by not allowing the possibility of interactions. However, if there exist large interactions and important loci with limited marginal effect, a search over additive models will have low power to detect such QTL.

The model comparison criterion is by far the most important aspect of a model selection procedure. Still, the model search procedure remains important. One will ideally identify optimal models with the shortest computation time. In the case of a single phenotype, a more extensive search may be feasible. But if a model selection approach is to be applied to many phenotypes, a reduced search may be required in order that the computations may be performed in a reasonable amount of time.

The penalized LOD criterion defined in Sec. 9.1.4 fails to consider covariates and QTL $\times$ covariate interactions. It is a simple matter to include a fixed set of additive covariates, but if one wishes to choose among a larger set of covariates or if one seeks to identify potential QTL $\times$ covariate interactions, the criterion would need to be modified, with special penalties for such terms.

In addition, the X chromosome generally requires special treatment. As described in Sec. 4.4, the potentially different number of parameters for a QTL on the X chromosome requires an X-chromosome-specific threshold, and so X-linked QTL may require a separate penalty. Similarly, epistatic interactions between QTL on the X chromosome, and between a QTL on the X chromosome and one on an autosome, also require special penalties (see Sec. 8.3).

Finally, linked QTL may deserve special treatment, as one may wish to be more permissive in identifying multiple linked QTL. The existence of multiple linked QTL can be extremely important in defining the course of fine-mapping experiments. Our penalized LOD criterion is quite strict in requiring strong evidence for an additional QTL, in order to control the rate of inclusion of extraneous loci. A more liberal approach for linked QTL could be valuable.

9.2 Bayesian QTL mapping

Our description of model selection in QTL mapping has followed a relatively traditional approach (though with some important differences). However, there has been much interest in, and important developments of, Bayesian methods for QTL mapping, for the most part relying on Markov chain Monte Carlo (MCMC). While we view a complete description of the Bayesian methods and their application as beyond the scope of this book, we would be lax to not include at least a brief description of these methods. In this section, we seek to highlight some of the advantages and disadvantages of the Bayesian methods for multiple QTL mapping.

Let y denote the phenotype, M the marker genotypes, q the unknown QTL genotypes, γ a QTL model (possibly with interactions), λ the locations

of the QTL, and μ all other model parameters (including QTL effects and the residual SD). The classical and Bayesian methods rely on the same likelihood function

$$\begin{aligned} L(\lambda, \mu, \gamma|y, M) &= \Pr(y|M, \lambda, \mu, \gamma) \\ &= \sum_q \Pr(y, q|M, \lambda, \mu, \gamma) \\ &= \sum_q \Pr(q|M, \lambda, \gamma) \Pr(y|q, \mu, \gamma) \end{aligned}$$

Note that, given the QTL genotypes q , the likelihood separates into two parts. $\Pr(q|M, \lambda, \gamma)$ concerns the relationship between the marker and QTL genotypes; $\Pr(y|q, \mu, \gamma)$ is the phenotype model.

In the classical approach, one maximizes over λ and μ (QTL positions and effects) to obtain the likelihood for a QTL model.

$$L(\gamma|y, M) = \max_{\lambda, \mu} L(\lambda, \mu, \gamma|y, M)$$

One chooses among QTL models, γ , by considering this likelihood, penalized for model complexity.

In the Bayesian approach, one specifies a prior distribution on QTL models and on QTL positions and effects, $\Pr(\lambda, \mu, \gamma)$. The prior is intended to capture one's initial uncertainty in the state of nature. Inference is then conducted through the posterior distribution, given the data.

$$\Pr(\gamma, \lambda, \mu|y, M) \propto L(\lambda, \mu, \gamma|y, M) \Pr(\lambda, \mu, \gamma)$$

In particular, one may consider the marginal posterior on QTL models, averaging (i.e., integrating) out QTL positions and effects.

There are thus two key distinctions between the classical and Bayesian approaches to QTL mapping. First, in the classical approach one maximizes over QTL positions and effects, while in the Bayesian approach one averages over these unknown parameters, using a suitable prior. Second, in the classical approach one considers the maximized likelihood for a model, penalized by model complexity, while in the Bayesian approach, one specifies a prior distribution on QTL models and then considers the posterior distribution given the data.

The key technical issue in the Bayesian approach concerns the calculation of the posterior distribution. The distribution is too complex to be described directly, and so we instead sample from the distribution and use the distribution across samples as an approximation to the posterior. Independent random samples are not feasible, and so we form a Markov chain whose stationary distribution is the target posterior distribution. (This is called Markov chain Monte Carlo, MCMC.) Let $\theta = (\lambda, \mu, \gamma)$. We form a Markov chain $\theta_1, \theta_2, \theta_3, \dots$ (that is, a sequence of random draws such that the distribution of θ_i depends only on θ_{i-1} and not on the entire history), whose limiting distribution is the target posterior, $\lim_{i \rightarrow \infty} \Pr(\theta_i) = \Pr(\theta|y, M)$.

There are a variety of techniques for constructing such a Markov chain. While constructing such a chain is relatively easy, great care is required to identify a chain with appropriate mixing properties (to reduce serial dependence and ensure rapid convergence to the stationary distribution). Such details are often confusing to the novice, and so we will omit them. The essence of the approach is that one obtains a sequence of draws, $\theta_1, \dots, \theta_k$, which we may view as a dependent sample from the posterior distribution, $\Pr(\theta|y, M)$. We may then derive a number of valuable summaries, including the posterior distribution of the number of QTL, the posterior probability that a particular genomic location is involved in the phenotype, and the posterior probability for a particular QTL model.

In the classical approach, we consider a quite strict definition for a QTL model, with the QTL in defined positions. In the Bayesian approach, with the locations of QTL varying across MCMC samples, it is best to soften one's view of a QTL model, and speak instead of a QTL pattern, such as "two QTL on chromosome 1, one on chromosome 4, and one on each of chromosomes 6 and 15, with the QTL on chromosomes 6 and 15 interacting." One may approximate the posterior probability of such a pattern by the proportion of MCMC samples for which the QTL model fits that pattern. One might then choose the pattern with the largest estimated posterior probability.

The Bayesian approach to QTL mapping has a number of advantages. It unifies all aspects of the problem (model fit, search and comparison), it provides a more natural expression of uncertainty in the results (particularly regarding the chance that a particular locus contributes to the phenotype), it more fully captures uncertainty (e.g., the estimated QTL effects take account of uncertainty in QTL positions), and extensions to include QTL $\times$ covariate interactions or alternate phenotype models are relatively straightforward.

The key difficulty concerns the specification of the prior distribution (particularly regarding the number of QTL and the number of interactions). In a sense, the choice of such a prior is equivalent to specification of the target false positive rate (e.g., increasing the expected number of QTL or interactions in the prior will lead to higher false positive rates), but the exact relationship is not easy to anticipate. In addition, a particular QTL model may be seen in only a small proportion of the MCMC samples, and not in the best possible light (as the QTL effects are also sampled). This is not a problem, if one focuses on the posterior probability for specific features of the underlying genetic architecture (such as the chance that a particular locus is involved), but it can make it difficult to define more complex features of the QTL model and so to compare the results of the Bayesian analysis to the results of the classical approach to the problem.

We prefer to say that the classical and Bayesian approaches to QTL mapping are complementary, with different features and different goals, but it may be that we are just being nice to our Bayesian colleagues or failing to admit the defeat of the classical approach.

In summary, the Bayesian approach to QTL mapping can provide a more satisfying set of results, with a more natural expression of the uncertainty in the inferential statements, but the specification of prior distributions is more difficult than the specification of false positive rates (as required for the classical approach). Moreover, the construction of appropriate MCMC algorithms requires great care, and use of MCMC in practice may require considerable training.

9.3 Multiple QTL mapping in R/qrtl

R/qrtl contains a variety of functions for the fit and exploration of multiple-QTL models, using the classical approach described in Sec. 9.1. [For Bayesian QTL mapping, consider the R/qrtlbim package (Yandell *et al.*, 2007).] We will first give a brief overview of the available functions. In the following subsections, we provide a detailed illustration of their use.

Only multiple imputation and Haley–Knott regression are currently implemented for the fit of a multiple-QTL model. The use of multiple interval mapping (MIM) and extended Haley–Knott regression, once implemented, will follow that of Haley–Knott regression, with any instances of `method="hk"` replaced by `method="em"` (for MIM) or `method="ehk"` (for extended Haley–Knott).

The two most basic functions are `makeqtl` and `fitqtl`. The function `makeqtl` is used to create a “QTL object” (of class “qtl”); this specifies the locations of a set of putative QTL to be considered. The function `fitqtl` is used to fit a defined QTL model, with QTL in fixed positions and with a defined set of covariates and potentially QTL $\times$ QTL and QTL $\times$ covariate interactions. The form of the QTL model is specified through a formula, such as `y~Q1+Q2+Q3*Q4`. The `fitqtl` function is particularly important, as it can be used to obtain estimates of QTL effects. One may also perform a “drop-one-QTL-at-a-time” analysis, to assess the support for individual loci and interactions.

The function `refineqtl` is used to refine the locations of QTL in the context of a multiple QTL model. It uses an iterative algorithm with the aim of obtaining the maximum likelihood estimates of the QTL positions. If the function is called with `keeplodprofile=TRUE`, one may then use the function `plotLodProfile` to plot LOD profiles for each QTL, again in the context of the multiple-QTL model, as is commonly used in multiple interval mapping.

With the function `addqtl`, one may scan for a single QTL to be added to a multiple-QTL model; with `addpair`, one may scan for an additional pair of QTL to be added. The output of these functions is of the forms produced by `scanone` and `scantwo`, and so one may use the corresponding plot and summary functions to inspect the results. The functions `addqtl` and `addpair` make use of a more basic and highly flexible function, `scanqtl`, for performing general, multidimensional scans in the context of a multiple-QTL model. The

output of `scanqtl` can be quite complicated to interpret, and, for most users, `addqtl` and `addpair` are sufficient, and so we will not discuss the use of `scanqtl` in this book.

The function `addint` may be used to test all possible pairwise interactions among the QTL in a multiple-QTL model.

There are several functions for manipulating a QTL object (created by `makeqtl`). The function `addtoqtl` is used to add additional QTL to an object, `dropfromqtl` is used to remove QTL from an object, `replaceqtl` is used to move QTL to new positions, and `reorderqtl` is used to change the order of loci within a QTL object.

Finally, the function `stepwiseqtl` provides a fully automated model selection algorithm, using the search algorithm described in Sec. 9.1.3 to optimize the penalized LOD score criterion described in Sec. 9.1.4.

9.3.1 `makeqtl` and `fitqtl`

We again return to the `hyper` data of Sugiyama *et al.* (2001) (see Sec. 2.3). We will use multiple imputation, as Haley–Knott regression performs poorly in the case of selectively genotyping, which was used for these data.

First we need to load the package and the data.

```
> library(qtl)
> data(hyper)
```

We must first run `sim.geno` to perform the imputations. We'll use 128 imputations; this is insufficient for the current data, which has extensive missing genotype information, but suffices to illustrate the methods. In practice, it is a good idea to repeat the analysis with independent imputations. If the results are much changed, increase the number of imputations. We will perform calculations on a 2 cM grid; a finer grid would provide more precise results but at the expense of greater computation time.

```
> hyper <- sim.geno(hyper, step=2, n.draws=128, err=0.001)
```

The results of `scanone` and `scantwo`, which we won't revisit, indicated QTL on chr 1, 4, 6 and 15, with an interaction between the QTL on chr 6 and 15, and the possibility of a second QTL on chr 1. (See Sec. 4.2.1 and 8.1.) We will begin by fitting this four-QTL model. (We take the QTL locations from the `scantwo` results on page 221.) The function `makeqtl` is used to create a QTL object; it pulls out the imputed genotypes at the selected locations.

```
> qtl <- makeqtl(hyper, chr=c(1, 4, 6, 15),
+               pos=c(68.3, 30, 60, 18))
```

Note that if you type the name of the QTL object, you get a brief summary. The QTL locations are not exactly as requested, as we are using a different step size.

```
> qtl
```

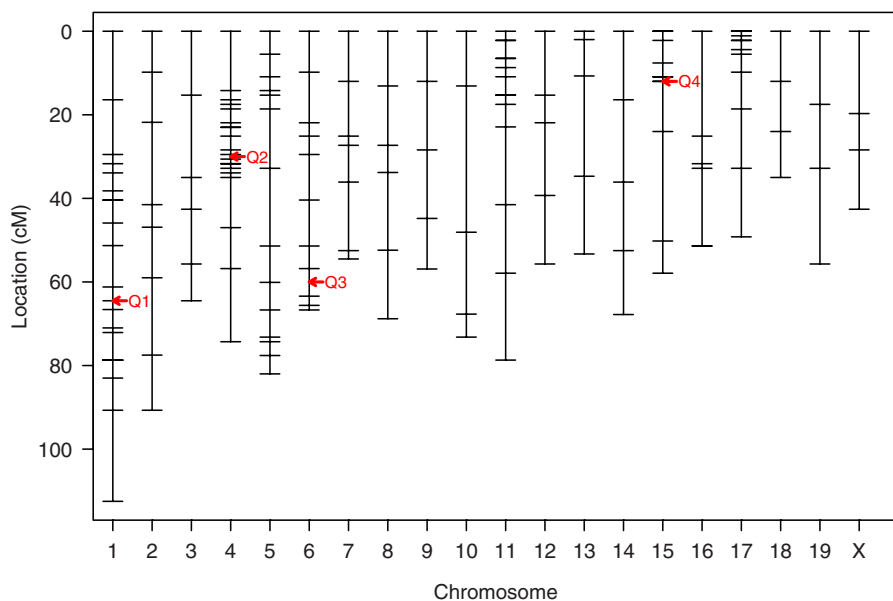


Figure 9.3. Locations of the QTL object `qtl` on the genetic map for the `hyper` data.

	name	chr	pos	n.gen
Q1	1@67.8	1	67.8	2
Q2	4@30.0	4	30.0	2
Q3	6@60.0	6	60.0	2
Q4	15@17.5	15	17.5	2

Also, there is a plot function for displaying the locations of the QTL on the genetic map. See Fig. 9.3.

```
> plot(qtl)
```

We may now use `fitqtl` to fit the model (with QTL in fixed positions). We use a model formula to indicate the model; in particular, we use `Q3*Q4` to indicate that QTL 3 and 4 should interact. (More on this shortly; see page 263.) The function `summary.fitqtl` is used to get a summary of the results.

```
> out.fq <- fitqtl(hyper, qtl=qtl, formula=y~Q1+Q2+Q3*Q4)
> summary(out.fq)
```

Full model result

```
-----
Model formula: y ~ Q1 + Q2 + Q3 + Q4 + Q3:Q4
```

	df	SS	MS	LOD	%var	Pvalue(Chi2)	Pvalue(F)
Model	5	5870	1174.01	21.92	33.22	0	0
Error	244	11799	48.36				
Total	249	17669					

Drop one QTL at a time ANOVA table:

	df	Type III SS	LOD	%var	F value
1@67.8	1	1319.608	5.755	7.469	27.289
4@30.0	1	2940.393	12.079	16.642	60.807
6@60.0	2	1615.617	6.967	9.144	16.705
15@17.5	2	1464.060	6.350	8.286	15.138
6@60.0:15@17.5	1	1174.205	5.150	6.646	24.282
		Pvalue(Chi2)		Pvalue(F)	
1@67.8		0.0000002629244384		0.000000375579930	
4@30.0		0.00000000000000876		0.0000000000000183	
6@60.0		0.0000001079672044		0.000000158669280	
15@17.5		0.0000004468012174		0.000000634616856	
6@60.0:15@17.5		0.0000011153038687		0.000001536448359	

The initial table indicates the overall fit of the model; the LOD score of 21.9 is relative to the null model (with no QTL). The sums of squares (SS) and mean square (MS) are as from an analysis of variance. The percent variance explained (%var) is the estimated proportion of the phenotype variance explained by all terms in the model.

In the second table, each locus is dropped from the model, one at a time, and a comparison is made between the full model and the model with the term omitted. If a QTL is dropped, any interactions it is involved in are also dropped, and so the loci on chr 6 and 15 are associated with 2 degrees of freedom, as the 6×15 interaction is dropped when either of these QTL is dropped.

The results indicate strong evidence for all of these loci as well as for the interaction. Let us briefly describe the columns in the table. Most important are the LOD scores, which are the $\log_{10}$ likelihood ratios comparing the full model (with all terms) to the reduced models (with one term omitted); the “Type III sums of squares” indicates the increase in the sum of the squared residuals accompanied by omitting the term. The F statistics are the ratio of the mean square (the sum of squares divided by the degrees of freedom) to the error mean square from the first table. The percent variance explained (%var) is the estimated proportion of the phenotypic variance explained by that term. Two pointwise p -values are provided. The first, `Pvalue(Chi2)`, is based on the LOD score, with the assumption that $\text{LOD} \times (2 \ln 10)$ follows a χ^2 distribution with the appropriate degrees of freedom. The second, `Pvalue(F)`, is based on the F statistic. Both p -values should be considered with caution, as they are *pointwise* and so do not account for the search over the genome

that led us to the current model. If the `summary.fitqtl` function is called with `pvalues=FALSE`, the two columns of *p*-values are omitted.

The “drop-one” analysis is particularly valuable for studying the support for the individual terms in the model. Another important use of `fitqtl` is to get estimated QTL effects. This is obtained with the use of `get.ests=TRUE`. We may use `dropone=FALSE` to suppress the drop-one analysis.

```
> out.fq2 <- fitqtl(hyper, qtl=qtl, formula=y~Q1+Q2+Q3*Q4,
+                   dropone=FALSE, get.ests=TRUE)
> summary(out.fq2)
```

Full model result

Model formula: y ~ Q1 + Q2 + Q3 + Q4 + Q3:Q4

	df	SS	MS	LOD	%var	Pvalue(Chi2)	Pvalue(F)
Model	5	5870	1174.01	21.92	33.22	0	0
Error	244	11799	48.36				
Total	249	17669					

Estimated effects:

	est	SE	t
Intercept	101.5642	0.4533	224.042
1@67.8	-4.7428	0.9159	-5.178
4@30.0	-7.0314	0.9191	-7.650
6@60.0	3.8161	0.9545	3.998
15@17.5	-2.3571	0.9197	-2.563
6@60.0:15@17.5	-9.0567	1.8941	-4.781

The estimated effects are derived by coding the homozygous and heterozygous genotypes (in a backcross) as -0.5 and $+0.5$, respectively. Thus, the estimated effect for a QTL is the difference between the phenotype averages for the heterozygotes and homozygotes. The `hyper` data come from the backcross $(B \times A) \times B$, with A and B being the A/J and C57Bl/6J mouse strains. The estimated effects for the chr 1, 4 and 15 loci being negative indicates that the A allele results in a decrease in blood pressure (i.e., heterozygotes, AB, have lower blood pressure than homozygotes, BB). (And note that the A strain has lower blood pressure than the B strain.) The chr 6 locus is a so-called *transgressive* QTL; the A allele is associated with an increase in blood pressure. (See also Fig. 8.7 on page 227.)

The interaction effect for the loci on chr 6 and 15 is large and negative. It is based on the product of the genotype columns for the two QTL. It can be interpreted as the difference in the effect of the chr 6 locus, according to whether an individual is heterozygous or homozygous at the chr 15 locus (and vice versa).

In the above, we have used multiple imputation for the calculations. To use Haley–Knott regression for such calculations, one must first call `calc.genoprob` (to calculate the conditional QTL genotype probabilities, given the marker data) rather than `sim.geno`. Then, in the call to `makeqtl`, one must use the argument `what="prob"`, to define QTL based on the genotype probabilities rather than the imputations. Finally, in the calls `fitqtl` (and the related functions, described below), one must use the argument `method="hk"`.

Model formulas deserve further explanation. Such formulas are used widely in R (see the R help file for `formula`); R/qtl uses a restricted version. First note that we always write “`y ~`” (to be read as “the phenotype y is modeled as...”) at the beginning of a QTL formula. QTL are indicated by their numeric index with the QTL object (Q1, Q2, etc.). Interactions between QTL may be specified using a colon; for example, `Q3:Q4` indicates that the interaction between QTL 3 and 4 should be included, and `Q1:Q3:Q4` indicates the three-way interaction between QTL 1, 3 and 4. An asterisk is similar, but indicates that all lower order interactions should also be included. For example, the term `Q3*Q4` is equivalent to `Q3+Q4+Q3:Q4`, and the term `Q1*Q3*Q4` indicates all first-order terms, all pairwise interactions, and the three-way interaction. That is, `Q1*Q3*Q4` is equivalent to `Q1+Q3+Q4+Q1:Q3+Q1:Q4+Q3:Q4+Q1:Q3:Q4`.

In all cases, we enforce a hierarchy in the QTL models, so that the inclusion of a pairwise interaction requires the inclusion of both of the corresponding main effects, and the inclusion of a three-way interaction requires the inclusion of the main effects and all pairwise interactions.

Covariates may be included in the QTL model fit by `fitqtl`. As with `scanone` and `scantwo`, the covariates must be strictly numeric. (See Chap. 7 and Sec. 8.4.) And here the set of covariates, which will be indicated through the argument `covar`, must form a matrix (or data frame). Rather than separately indicating additive and interactive covariates, one refers to covariates and QTL $\times$ covariate interactions in the model formula. Refer to the covariates in the formula by their column names.

9.3.2 `refineqtl`

The function `refineqtl` allows us to get improved estimates of the locations of the QTL. The position for each QTL is varied, one at a time, keeping all other QTL locations fixed, and keeping the chromosome assignments and order of QTL fixed. The process is iterated to convergence. We use `verbose=FALSE` to suppress the display of tracing information.

```
> rqt1 <- refineqtl(hyper, qtl=qtl, formula=y~Q1+Q2+Q3*Q4,
+                   verbose=FALSE)
```

The output is a modified QTL object, with loci in new positions. We can type the name of the new QTL object to see the new locations.

```
> rqt1
```

	name	chr	pos	n.gen
Q1	1@67.8	1	67.8	2
Q2	4@30.0	4	30.0	2
Q3	6@66.7	6	66.7	2
Q4	15@17.5	15	17.5	2

The locus on chr 6 changed position slightly. Let us use `fitqtl` to assess the improvement in fit; we'll skip the drop-one analysis.

```
> out.fq3 <- fitqtl(hyper, qtl=rqtl, formula=y~Q1+Q2+Q3*Q4,
+                  dropone=FALSE)
> summary(out.fq3)
```

Full model result

Model formula: y ~ Q1 + Q2 + Q3 + Q4 + Q3:Q4

	df	SS	MS	LOD	%var	Pvalue(Chi2)	Pvalue(F)
Model	5	5893	1178.64	22.03	33.35	0	0
Error	244	11776	48.26				
Total	249	17669					

The LOD score comparing the full model to the null model has increased by 0.1, from 21.9 to 22.0.

By default, `refineqtl` saves the LOD traces at the last iteration, which can then be plotted with `plotLodProfile`, as follows.

```
> plotLodProfile(rqtl, ylab="Profile LOD score")
```

The LOD profiles in Fig. 9.4 are similar to the usual LOD curves, but instead of comparing a model with a single QTL at a particular position to the null model, we compare, at each position for a given QTL, the model with the QTL of interest at that particular position (and with the positions of all other QTL fixed at their maximum likelihood estimates) to the model with the QTL of interest omitted (and with the positions of all other QTL fixed at their maximum likelihood estimates). For the loci on chr 6 and 15, the 6×15 interaction is omitted when either of the two loci is omitted.

And so in the LOD profile for the locus on chr 1, we compare the four-QTL model (including the 6 × 15 interaction and with the position of the chr 1 QTL varying but with the other three QTL fixed at their estimated locations) to the three-QTL model with the chr 1 locus omitted. In the LOD profile on chr 15, we compare the four-QTL model (with the position of the chr 15 locus varying but with the other three QTL fixed at their estimated locations) to the three-QTL additive model (that is, without the chr 15 locus and without the 6 × 15 interaction). Note that the maximum LOD for each of the LOD profiles should be the value observed in the drop-one analysis from `fitqtl`.

These profile LOD curves are useful for the assessment of both the evidence for the individual QTL and the precision of localization of each QTL, but note

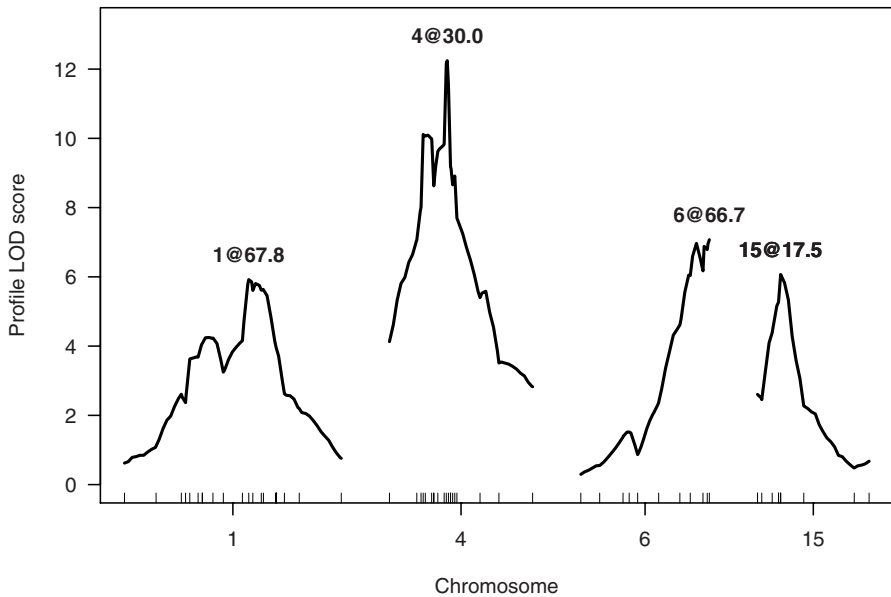


Figure 9.4. LOD profiles for a four-QTL model with the `hyper` data.

that they fail to take account of the uncertainty in the location of the other QTL in the model.

The functions `lodint` and `bayesint` (see Sec. 4.5) can be used to derive approximate confidence intervals for the locations of the QTL, using the LOD profiles calculated by `refineqt1`. However, these should be viewed with some caution, as they are calculated assuming that the locations of the other QTL are known without error (that is, they fail to account for the uncertainty in the locations of the other QTL), and the performance of these approximate intervals in the context of a multiple-QTL model is not well understood. Thus, they are sure to be overly liberal (that is, their coverage probabilities are likely less than 95%).

To calculate a 1.5-LOD support interval and an approximate 95% Bayesian credible interval for the QTL on chr 4, we refer to the QTL, in `lodint` and `bayesint`, by its numeric index (in this case, 2).

```
> lodint(rqt1, qtl.index=2)

      chr pos  lod
D4Mit288  4 28.4  9.83
c4.loc30  4 30.0 12.24
D4Mit80   4 31.7  9.18

> bayesint(rqt1, qtl.index=2)
```

	chr	pos	lod
D4Mit164	4	29.5	12.17
c4.1oc30	4	30.0	12.24
D4Mit178	4	30.6	11.55

These intervals are *much* more narrow than the intervals calculated in the context of a single-QTL model (see Sec. 4.5). The 1.5-LOD support interval is 3.3 cM long (versus 12.0 cM), and the approximate 95% Bayesian credible interval is. 1.1 cM long (versus 13.1 cM).

9.3.3 addint

The function `addint` is used to test, one at a time, all possible QTL $\times$ QTL interactions that are not already included in a model. For our model with loci on chr 1, 4, 6 and 15, and with a 6 $\times$ 15 interaction, we consider each of the five other possible pairwise interactions, and compare the base model (with the four QTL and just the 6 $\times$ 15 interaction) to the model with the additional interaction included.

The syntax of the function is similar to that of `fitqtl`. The output is a table of results similar to that provided by the drop-one analysis of `fitqtl`. As with `fitqtl`, by default two columns of pointwise p -values are provided: one based on the assumption that, under the null hypothesis, $\text{LOD} \times (2 \ln 10)$ follows a χ^2 distribution with the appropriate degrees of freedom, and a second based on the F statistic. As in `fitqtl`, the p -values should be considered with caution, as they are *pointwise* and so do not account for the search over the genome that led us to the current model. To save space, we will omit the p -values from the tabular results via `pvalues=FALSE`.

```
> addint(hyper, qtl=rqtl, formula=y~Q1+Q2+Q3*Q4, pvalues=FALSE)
```

```
Model formula: y ~ Q1 + Q2 + Q3 + Q4 + Q3:Q4
```

```
Add one pairwise interaction at a time table:
```

	df	Type III SS	LOD	%var	F	value
1@67.8:4@30.0	1	61.380843	0.283709	0.347394	1.273	
1@67.8:6@66.7	1	1.402998	0.006468	0.007940	0.029	
1@67.8:15@17.5	1	71.144546	0.328975	0.402653	1.477	
4@30.0:6@66.7	1	64.916064	0.300094	0.367402	1.347	
4@30.0:15@17.5	1	16.225653	0.074853	0.091832	0.335	

There is little evidence for any of these interactions.

Different results are obtained if we use as the formula `y~Q1+Q2+Q3+Q4` (that is, omitting the 6 $\times$ 15 interaction).

```
> addint(hyper, qtl=rqtl, formula=y~Q1+Q2+Q3+Q4, pvalues=FALSE)
```

Model formula: $y \sim Q1 + Q2 + Q3 + Q4$

Add one pairwise interaction at a time table:

	df	Type III SS	LOD	%var	F value
1067.8:4030.0	1	60.30574	0.25455	0.34131	1.147
1067.8:6066.7	1	11.62534	0.04898	0.06580	0.220
1067.8:15017.5	1	86.59440	0.36589	0.49009	1.650
4030.0:6066.7	1	39.81677	0.16793	0.22535	0.756
4030.0:15017.5	1	58.92350	0.24870	0.33349	1.120
6066.7:15017.5	1	1115.46429	4.91316	6.31314	23.113

The 6×15 interaction is also tested, and the LOD scores for the other interactions are somewhat different, as they concern comparisons between the four-locus additive model and the model with that one interaction added.

9.3.4 addqtl

The `addqtl` function is used to scan for an additional QTL, to be added to the model. By default, the new QTL is strictly additive.

```
> out.aq <- addqtl(hyper, qtl=rqtl, formula=y~Q1+Q2+Q3*Q4)
```

The output of `addqtl` has the same form as that from `scanone`, and so we may use the same summary and plot functions. For example, we can identify the genome-wide maximum LOD score with `max.scanone`.

```
> max(out.aq)

      chr pos lod
D5Mit31  5 66.7 1.58
```

We may plot the results with `plot.scanone`; see Fig. 9.5.

```
> plot(out.aq, ylab="LOD score")
```

The LOD scores compare the base model to the model with one additional QTL. There is a suggestion of an additional QTL on chr 5, but the evidence is not strong.

We may also use `addqtl` to scan for an additional QTL, interacting with one of the current loci. This is done by including the additional QTL in the model formula, with the relevant interaction term. For example, let's scan for an additional QTL interacting with the chr 15 locus.

```
> out.aqi <- addqtl(hyper, qtl=rqtl,
+                   formula=y~Q1+Q2+Q3*Q4+Q4*Q5)
```

We plot the results as follows; see Fig. 9.6.

```
> plot(out.aqi, ylab="LOD score")
```

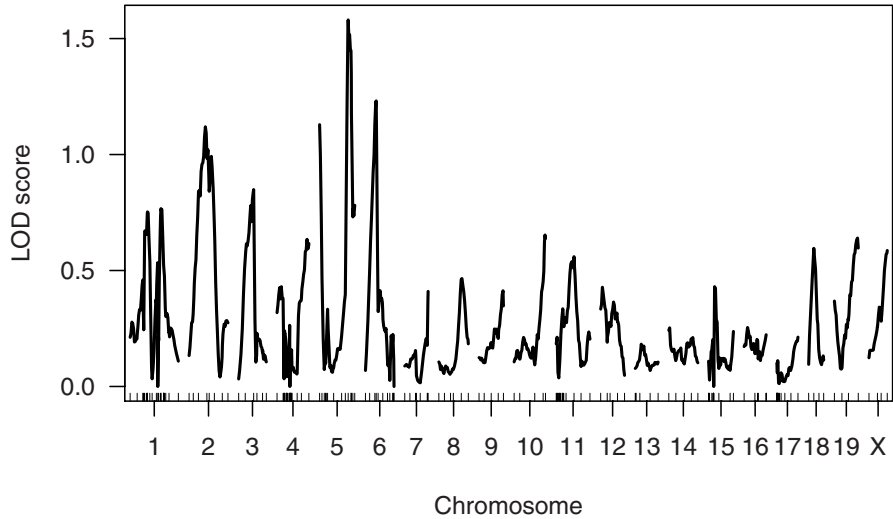


Figure 9.5. LOD curves for adding one QTL to the four-QTL model, with the `hyper` data.

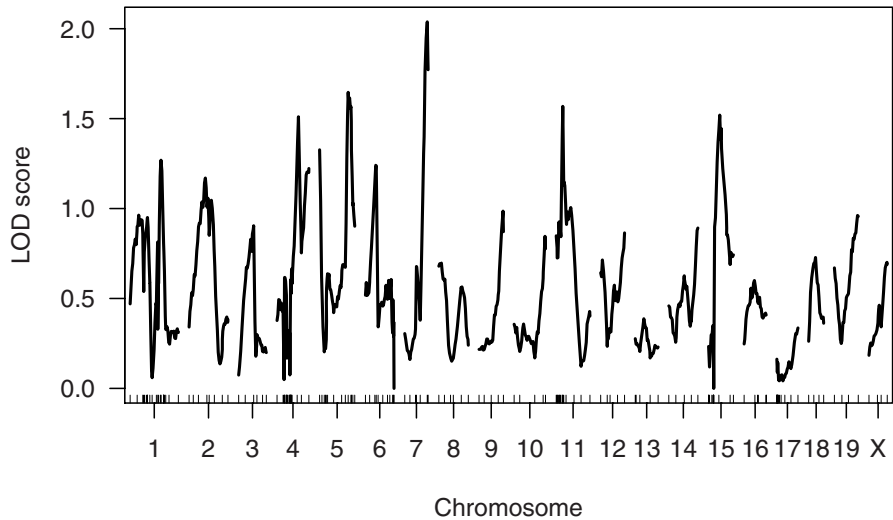


Figure 9.6. LOD curves for adding one QTL, interacting with the chr 15 locus, to the four-QTL model, with the `hyper` data.

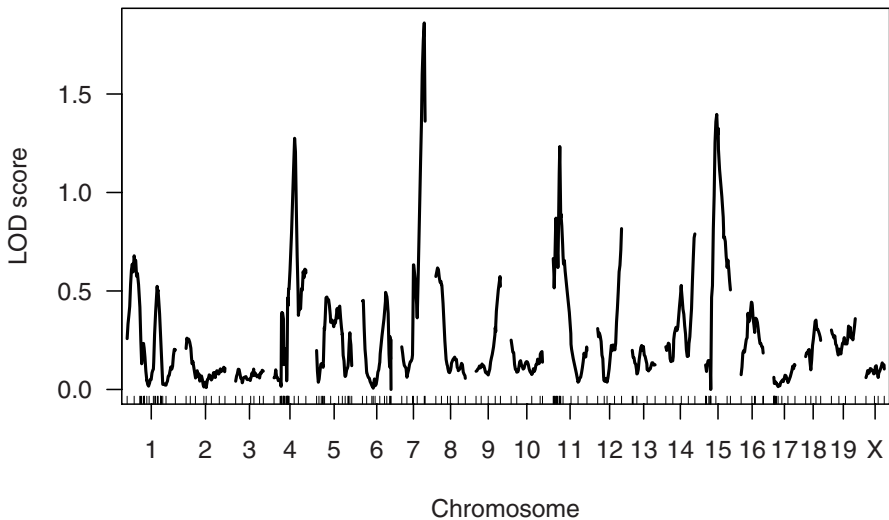


Figure 9.7. Interaction LOD curves in the scan for an additional QTL, interacting with the chr 15 locus, to be added to the four-QTL model, with the `hyper` data.

Also of interest are the LOD scores for the interaction between the chr 15 locus and the new locus being scanned, which are the differences between the LOD scores in `out.aqi` and `out.aq`. See Fig. 9.7.

```
> plot(out.aqi - out.aq, ylab="LOD score")
```

There is nothing particularly exciting in either of these plots (though there is a suggestion of a QTL on chr 7, interacting with the QTL on chr 15).

9.3.5 `addpair`

The function `addpair` is similar to `addqtl`, but it performs a two-dimensional scan to seek a pair of QTL to add to a multiple-QTL model. By default, `addpair` performs a two-dimensional scan analogous to that of `scantwo`: for each pair of positions for the two putative QTL, it fits both an additive model and a model including an interaction between the two QTL.

Recall that in the single-QTL analysis with the `hyper` data, there were two peaks in the LOD curve on chr 1, indicating that there may be two QTL on that chromosome. In the context of our multiple-QTL model, the LOD profile on chr 1 (see Fig. 9.4) still shows two peaks, though the distal peak is more prominent.

We may use `addpair` to investigate the possibility of a second QTL on chr 1. To do so, we omit the chr 1 locus from our formula, and perform a two-dimensional scan just on chr 1.

```
> out.ap <- addpair(hyper, qtl=rqtl, chr=1, formula=y~Q2+Q3*Q4,
+                   verbose=FALSE)
```

The output is of the same form as that produced by `scantwo`, and so we may use the same summary and plot functions.

```
> summary(out.ap)
```

```
      pos1f pos2f lod.full lod.fv1 lod.int      pos1a pos2a
c1:c1  43.3  73.3   7.83    1.98  0.458    45.3  77.3
      lod.add lod.av1
c1:c1    7.37    1.52
```

There is little evidence for an interaction, and the LOD score comparing the model with two additive QTL on chr 1 to that with a single QTL on chr 1 is 1.52, indicating relatively weak evidence for a second QTL on chr 1.

Let us also plot the results. We'll focus on the evidence for a second QTL on the chromosome, displaying LOD_{fv1} (evidence for a second QTL, allowing for an interaction) and LOD_{av1} (evidence for a second QTL, assuming the two are additive). See Fig 9.8.

```
> plot(out.ap, lower="cond-int", upper="cond-add")
```

There is a good deal of flexibility in the way that `addpair` may be used. As in `addqtl`, where one can scan for loci that interact with a particular locus in the model, we can use `addpair` to scan for an additional pair, with any prespecified set of interactions.

For example, we may retain the loci on chr 1, 4 and 6, and scan for an additional pair of interacting loci, one of which also interacts with chr 6. This would be useful for assessing evidence for an additional QTL interacting with the chr 15 locus, but allowing the location of the locus on chr 15 to vary. We use the formula $y \sim Q1 + Q2 + Q3 + Q5 * Q6 + Q3 : Q5$, as we will omit the chr 15 locus ($Q4$), scan for an additional interacting pair ($Q5 * Q6$), and allow the first QTL in the additional pair to interact with the chr 6 locus ($Q3$). Note that the positions of the chr 1, 4 and 6 loci are assumed known. A three-dimensional scan could be performed with the `scanqtl` function, but we do not discuss such searches in this book.

To save time, we will focus just on chr 7 and 15.

```
> out.ap2 <- addpair(hyper, qtl=rqt1, chr=c(7,15), verbose=TRUE,
+                   formula=y~Q1+Q2+Q3+Q5*Q6+Q3:Q5)
```

Because we are using a special formula here, with one of the new QTL interacting with the chr 6 locus, the results are similar to, but not quite the same as, those from `scantwo`. Rather than fitting an additive and an interactive model at each pair of positions, we fit just the single model specified in the formula. And note that as the formula is not symmetric in $Q5$ and $Q6$, we must do a full 2-dimensional scan, rather than just scan the triangle. (That is, we need $Q5$ and $Q6$ assigned to chromosomes (7,15) as well as (15,7).)

The summary of the results are somewhat different here. For each pair of chromosomes, a set of three LOD scores are presented. `lod.2v0` compares the

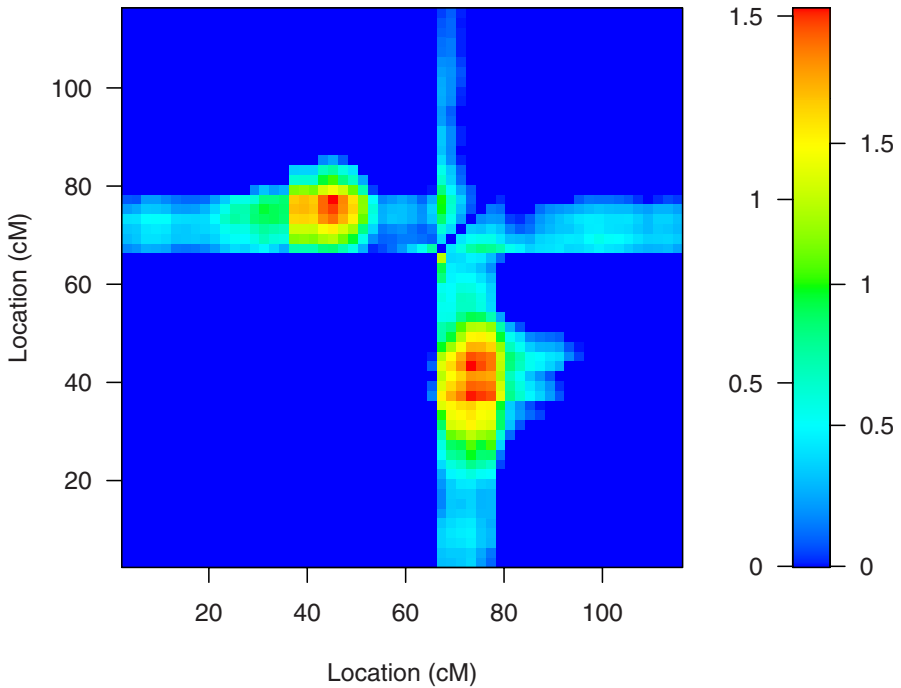


Figure 9.8. Results of a two-dimensional, two-QTL scan on chr 1, in the context of a model with additional QTL on chr 4, 6 and 15, and a 6×15 interaction, with the **hyper** data. LOD_{av1} is in the upper left triangle, and LOD_{fv1} is in the lower right triangle. In the color scale on the right, numbers to the left and right correspond to LOD_{av1} and LOD_{fv1} , respectively.

full model to the model with neither of the two new QTL included, `lod.2v1b` compares the full model to the model with the first of the two new QTL omitted, and `lod.2v1a` compares the full model to the model with the second QTL omitted. When a QTL is omitted, any interactions involving that QTL are also omitted.

```
> summary(out.ap2)
```

	pos1a	pos2a	lod.2v0	lod.2v1b	lod.2v1a
c7 :c7	29.1	25.1	2.89	2.54	1.55
c7 :c15	51.1	15.5	3.84	2.51	2.51
c15:c7	17.5	53.1	8.08	7.72	2.01
c15:c15	17.5	31.5	7.59	6.26	1.52

Note that, because of the lack of symmetry in the formula we used in `addpair`, separate results are provided for the two cases `c7:c15` (in which the chr 7 locus interacts with the chr 6 locus) and `c15:c7` (in which the chr 15 locus interacts with the chr 6 locus). The `c15:c7` row is most interesting,

but `lod.2v1a` is 2.01, indicating little evidence for a chr 7 locus. (Note that `lod.2v1a` here concerns both the chr 7 locus and the 7×15 interaction.) This is the same as the peak on chr 7 in Fig. 9.6, in which we scanned for an additional locus, interacting with the chr 15 locus. While here we allowed the location of the chr 15 locus to vary, the estimated location was 17.5 cM, as before.

With this sort of `addpair` output, the `thresholds` argument should have length just 1 or 2 (which is different from the usual case for `summary.scantwo`). Rows will be retained if `lod.2v0` is greater than `thresholds[1]` and either of `lod.2v1a` or `lod.2v1b` is greater than `thresholds[2]`. (If a single threshold is given, we assume that `thresholds[2]==0`.) Note that, of the other arguments to `summary.scantwo`, all but `allpairs` is ignored.

The plot of the output from `addpair`, in the case of a special formula, is also different from the usual `scantwo` plot.

```
> plot(out.ap2)
```

The plot, shown in Fig. 9.9, contains LOD scores comparing the full five-QTL model to the three-QTL model (having loci on chr 1, 4 and 6). The *x*-axis corresponds to the first of the new QTL (Q5), which is the one that interacts with the chr 6 locus. The *y*-axis corresponds to the second of the new QTL (Q6). Clearly, the QTL interacting with the chr 6 locus wants to be on chr 15.

Note that the `lower` and `upper` arguments to `plot.scantwo` are ignored in this case.

9.3.6 Manipulating qtl objects

Our analysis of the `hyper` data, above, did not indicate much evidence for any further QTL. If we had seen evidence for additional loci, we would want to add them to the QTL object and repeat our explorations with `fitqtl`, `addint`, `addqtl`, and `addpair`.

The functions `addtoqtl`, `dropfromqtl` and `replaceqtl` can be used to facilitate such analyses. Rather than recreating a QTL object from scratch with `makeqtl`, one can use `addtoqtl` to add an additional locus to a QTL object that was previously created. For example, if we were satisfied with the evidence for an additional QTL on chr 1, it could be added to the QTL object `rqt1` as follows. We use `print` to simultaneously assign the result to an object and print it.

```
> print( rqt12 <- addtoqtl(hyper, rqt1, 1, 43.3) )
```

	name	chr	pos	n.gen
Q1	1@67.8	1	67.8	2
Q2	4@30.0	4	30.0	2
Q3	6@66.7	6	66.7	2

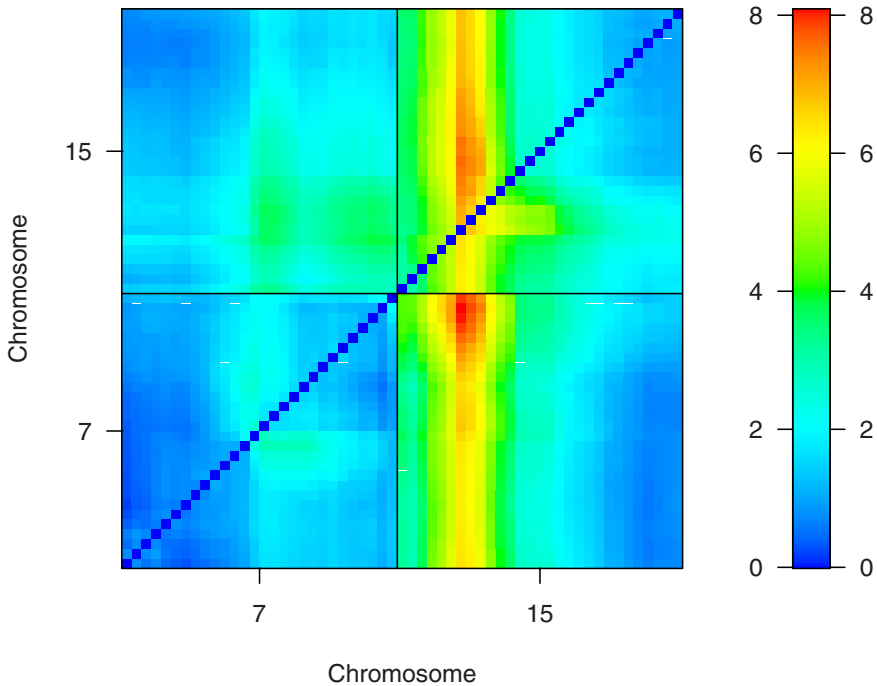


Figure 9.9. Results of a two-dimensional, two-QTL scan on chr 7 and 15, in the context of a model with additional QTL on chr 1, 4, and 6, with the `hyper` data. The two QTL being scanned were allowed to interact, and the first of them interacts with the chr 6 locus. The LOD scores displayed are for the five-QTL model relative to the three-QTL model. The x -axis corresponds to the first of the new QTL (which interacts with the chr 6 locus); the y -axis corresponds to the second of the new QTL.

```
Q4 15@17.5  15 17.5    2
Q5  1@43.3   1 43.3    2
```

The syntax of `addtoqtl` is much like that of `makeqtl`, though one also provides the QTL object to which additional QTL are to be added.

If we want to move the first QTL on chr 1 to a different position (say to 77.3 cM rather than 67.8 cM), we may use `replaceqtl`. We refer to the QTL that are to be replaced by their numeric index, with the argument `index`. (That is the first 1 below; the second 1 indicates the chromosome.)

```
> print( rqt13 <- replaceqtl(hyper, rqt12, 1, 1, 77.3) )
```

	name	chr	pos	n.gen
Q1	1@77.3	1	77.3	2
Q2	4@30.0	4	30.0	2
Q3	6@66.7	6	66.7	2

```
Q4 15@17.5 15 17.5 2
Q5 1@43.3 1 43.3 2
```

If we wish to reorder the QTL (e.g., according to their map positions), we may use `reorderqtl`. We may provide a vector of numeric indices specifying the new order.

```
> print( rqt14 <- reorderqtl(rqt13, c(4,3,2,1,5)) )
```

```
      name chr  pos n.gen
Q1 15@17.5 15 17.5 2
Q2 6@66.7 6 66.7 2
Q3 4@30.0 4 30.0 2
Q4 1@77.3 1 77.3 2
Q5 1@43.3 1 43.3 2
```

Alternatively, if `reorderqtl` is called with only the QTL object, the QTL are ordered according to their genomic positions.

```
> print( rqt15 <- reorderqtl(rqt14) )
```

```
      name chr  pos n.gen
Q1 1@43.3 1 43.3 2
Q2 1@77.3 1 77.3 2
Q3 4@30.0 4 30.0 2
Q4 6@66.7 6 66.7 2
Q5 15@17.5 15 17.5 2
```

Finally, `dropfromqtl` is used to drop a locus from a QTL object. To drop the proximal locus on chr 1 (now the first QTL in the object), we would do the following.

```
> print( rqt16 <- dropfromqtl(rqt15, 1) )
```

```
      name chr  pos n.gen
Q1 1@77.3 1 77.3 2
Q2 4@30.0 4 30.0 2
Q3 6@66.7 6 66.7 2
Q4 15@17.5 15 17.5 2
```

In `dropfromqtl`, we may refer to the QTL to be dropped either by their numeric index (through the argument `index`), by their chromosome and position (through the arguments `chr` and `pos`), or by their name (through the argument `qtl.name`).

9.3.7 stepwiseqtl

With the function `stepwiseqtl`, one may use the forward/backward stepwise search algorithm described in Sec. 9.1.3 to optimize the penalized LOD score

criterion described in Sec. 9.1.4. In this section, we illustrate the use of this function through application to the `hyper` data.

We must first derive the appropriate penalties for the penalized LOD score. This requires a permutation test with a two-dimensional, two-QTL genome scan. While we had performed such permutations in Sec. 8.1, there we had used maximum likelihood via the EM algorithm to deal with missing genotype data, and here we are using multiple imputation. As the results may be somewhat different with the two approaches, we must rerun the permutation analysis. Extremely hefty computations are required, on the order of 100 hours, in total. Thus it is best to split the permutations into batches to be performed in parallel using multiple processors. Such parallel computations were described in Sec. 8.1; see page 223.

Recall that, due to the selective genotyping, it is best to do a stratified permutation test (permuting separately within the two strata defined by the extent of genotype data available). And so we first define these strata, and then perform the permutation test with `scantwo`.

```
> strat <- (nmissing(hyper) > 50)
> operm2 <- scantwo(hyper, method="imp", n.perm=1000,
+                   perm.strat=strat)
```

The `summary.scantwoperm` function may be used to obtain estimated thresholds.

```
> summary(operm2)

bp (1000 permutations)
  full  fv1  int  add  av1  one
5%  5.41  4.19  3.79  4.34  2.29  2.55
10% 5.09  3.91  3.51  3.97  2.05  2.28
```

The function `calc.penalties` is used to derive the penalties from the permutation results. By default, we use a significance level of 5%. We use `print` in the following so that we may simultaneously assign the penalties to an object and print the results.

```
> print(pen <- calc.penalties(operm2))

main heavy light
2.553 3.793 1.641
```

Note that these are quite different from the penalties presented in Sec. 9.1.4 (Table 9.1 on page 252), derived by computer simulation (in an admittedly artificial situation). In particular, the heavy interaction penalty is quite a bit larger (3.8 vs. 2.6).

The penalties corresponding to multiple significance levels can be derived at the same time.

```
> calc.penalties(operm2, alpha=c(0.05, 0.20))
```

```

      main heavy light
5%  2.553 3.793 1.641
20% 1.983 3.197 1.610

```

With these penalties in hand, we are now prepared to apply the fully automated model search algorithm with `stepwiseqtl`. The search algorithm uses forward selection to a model with a fixed number of QTL, at each step searching for an additional additive QTL, or an additional QTL interacting with one of the QTL in the current model. The forward selection process is followed by backward elimination to the null model. The final chosen model is that with the maximal penalized LOD score, among all models visited.

```

> outsw1 <- stepwiseqtl(hyper, max.qtl=8, penalties=pen,
+                       verbose=FALSE)

```

The output is a QTL object (of class "qtl", as created by the function `makeqtl`) defining the chosen model.

```

> outsw1

      name chr  pos n.gen
Q1  1@67.8   1 67.8    2
Q2  4@30.0   4 30.0    2
Q3  6@66.7   6 66.7    2
Q4 15@17.5  15 17.5    2

```

```

Formula: y ~ Q1 + Q2 + Q3 + Q4 + Q3:Q4
pLOD: 10.18

```

This is identical to the model obtained in Sec. 9.3.2 (page 264).

If `stepwiseqtl` is run with the argument `keeplodprofile=TRUE`, one may obtain the LOD profiles for the four QTL in the model (as with the function `refineqtl`), which may then be plotted with `plotLodProfile` (see Sec. 9.3.2).

With the argument `keeptrace=TRUE`, the output will include the sequence of models visited in the forward/backward search algorithm. (That is, we retain information on the single best model at each step of forward selection and backward elimination.)

Let us rerun `stepwiseqtl` with these options, and visualize the results.

```

> outsw2 <- stepwiseqtl(hyper, max.qtl=8, penalties=pen,
+                       verbose=FALSE, keeplodprofile=TRUE,
+                       keeptrace=TRUE)

```

The LOD profiles may be displayed as follows. As the model is identical to that considered in Sec. 9.3.2, the LOD profiles are identical to those displayed in Fig. 9.4 on page 265, and so the plot will not be repeated.

```

> plotLodProfile(outsw2)

```

The sequence of models visited by `stepwiseqlt` are retained as an *attribute* (called `"trace"`) of the output, `outsw2`. Attributes are a way of hiding additional information within an object. The entire set of attributes for an object may be obtained with the `attributes` function. It is often useful to just look at the names of the attributes.

```
> names(attributes(outsw2))

[1] "names"      "class"      "map"        "lodprofile"
[5] "formula"    "pLOD"       "trace"
```

Individual attributes may be obtained with the `attr` function. So we can pull out the trace of models with the following. This is a long list, with each component being a compact representation of a QTL model, and so we will print just the first of them.

```
> thetrace <- attr(outsw2, "trace")
> thetrace[[1]]

chr pos
Q1  4 29.5

Formula: y ~ Q1
pLOD: 5.539
```

It is nicer to look at a sequence of pictures rather than a long list of models. The function `plotModel` may be used to plot a graphical representation of a model, with nodes (i.e., dots) representing QTL and edges (i.e., line segments connecting two nodes) representing pairwise interactions among QTL. The argument `chronly` is used to print just the chromosome ID for each QTL (rather than chromosome and position). The penalized LOD score for each model is saved as an attribute, `"pLOD"`; we include them in the title of each subplot, but this requires another call to `attr`.

```
> par(mfrow=c(6,3))
> for(i in seq(along=thetrace))
+   plotModel(thetrace[[i]], chronly=TRUE,
+             main=paste(i, ": pLOD =",
+             round(attr(thetrace[[i]], "pLOD"), 2)))
```

As seen in Fig. 9.10, our chosen model is identified immediately (at step 4). Note that the model at step 3 (with additive QTL on chr 1, 4 and 6) has a lower penalized LOD score than the model at step 2 (with just the chr 1 and 4 QTL), but then the inclusion of the chr 15 QTL and the 6×15 interaction gave the largest penalized LOD score, among all models visited. With the addition of a QTL on chr 5 (at step 5), the pLOD decreased somewhat; the LOD score for the model increased, but not as much as the main effect penalty.

In the above, we used the penalized LOD score criterion that balances the use of the heavy and light interaction penalties (see Sec. 9.1.4). If we call

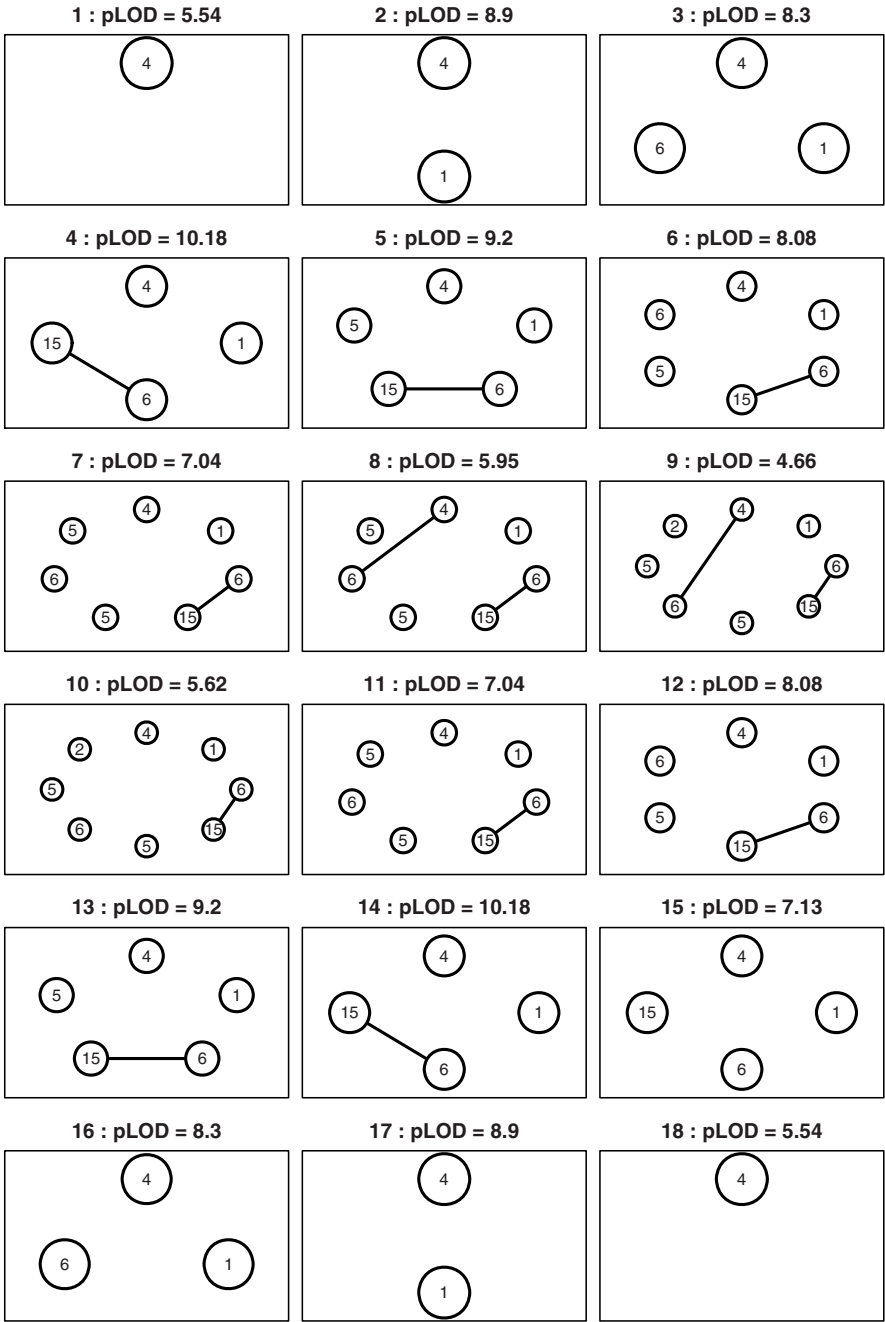


Figure 9.10. The sequence of models visited by the forward/backward search of `stepwiseqtl`, with the `hyper` data.

`stepwiseqtl` with just the first two penalties (the main effect penalties and the heavy interaction penalty), the light interaction penalty will be taken to be the same as the heavy interaction penalty, and so all pairwise interactions will be assigned the same penalty.

```
> outsw3 <- stepwiseqtl(hyper, max.qtl=8, penalties=pen[1:2],
+                       verbose=FALSE)
```

With only heavy interaction penalties, we choose the model with just the QTL on chr 1 and 4, and with no interactions.

```
> outsw3
```

	name	chr	pos	n.gen
Q1	1@67.8	1	67.8	2
Q2	4@29.5	4	29.5	2

```
Formula: y ~ Q1 + Q2
pLOD: 8.897
```

We could also consider more liberal penalties, such as those corresponding to a significance level of 20%.

```
> liberalpen <- calc.penalties(operm2, alpha=0.2)
> outsw4 <- stepwiseqtl(hyper, max.qtl=8, penalties=liberalpen,
+                       verbose=FALSE)
```

No additional QTL are obtained with these more liberal penalties. The penalties based on a significance level of 5% are conservative; the more liberal penalties can lead to a larger model (though, as seen here, not necessarily), but the false positive rate (the chance of extraneous loci being included in the chosen model) will be higher.

```
> outsw4
```

	name	chr	pos	n.gen
Q1	1@67.8	1	67.8	2
Q2	4@30.0	4	30.0	2
Q3	6@66.7	6	66.7	2
Q4	15@17.5	15	17.5	2

```
Formula: y ~ Q1 + Q2 + Q3 + Q4 + Q3:Q4
pLOD: 12.48
```

The `stepwiseqtl` function has a number of additional arguments. With `additive.only=TRUE`, one may consider only additive QTL models (allowing no interactions among QTL). With `scan.pairs=TRUE`, one may perform a two-dimensional, two-QTL scan at each step of forward selection. This could enhance our ability to identify interacting loci with limited marginal effects

and pairs of loci that are linked in repulsion (having effects of opposite signs). However, the computational effort is great, and our limited experience suggests that it may not be necessary.

We won't explore the use of `scan.pairs=TRUE`, but let us see what happens when we focus solely on additive QTL models.

```
> outsw5 <- stepwiseqtl(hyper, max.qtl=8, penalties=pen,
+                       additive.only=TRUE, verbose=FALSE)
```

The chosen model then contains only the QTL on chr 1 and 4.

```
> outsw5
```

	name	chr	pos	n.gen
Q1	1@67.8	1	67.8	2
Q2	4@29.5	4	29.5	2

```
Formula: y ~ Q1 + Q2
pLOD: 8.897
```

Finally note that while, in the above, we started forward selection at the null QTL model, one may also begin the algorithm at any defined QTL model. For example, we could start at the first model that we considered in Sec. 9.3.1, determined from the results of `scanone` and `scantwo`. The starting model is indicated through the arguments `qtl` (a QTL object, created with `makeqtl`) and `formula` (a model formula).

```
> qtl <- makeqtl(hyper, chr=c(1, 4, 6, 15),
+               pos=c(68.3, 30, 60, 18))
> outsw6 <- stepwiseqtl(hyper, max.qtl=8, penalties=pen,
+                       qtl=qtl, formula=y~Q1+Q2+Q3*Q4,
+                       verbose=FALSE)
```

We get almost the same result, though we get to it slightly faster, as we skip the first few steps of forward selection.

```
> outsw6
```

	name	chr	pos	n.gen
Q1	1@67.8	1	67.8	2
Q2	4@29.5	4	29.5	2
Q3	6@60.0	6	60.0	2
Q4	15@17.5	15	17.5	2

```
Formula: y ~ Q1 + Q2 + Q3 + Q4 + Q3:Q4
pLOD: 10.13
```

9.4 Summary

The QTL mapping problem is best viewed as one of model selection: we seek to identify the set of QTL and epistatic interactions that are best supported by the data. While the sequence of hypothesis tests used for single- and two-QTL genome scans are a useful technique and often are sufficient, multiple-QTL mapping methods have the advantages of providing potentially increased power to detect QTL, of better separating linked QTL, and of more clearly defining epistatic interactions, and the hypothesis testing approach falls apart when one is faced with multiple-QTL models.

The most important aspect of the model selection approach to QTL mapping is the criterion for comparing models. We have described a penalized LOD score approach, with the aim to identify as many true QTL as possible while controlling the rate of inclusion of extraneous loci.

Bayesian methods for QTL mapping have a number of advantages over our more classical approach to the problem. They unify all aspects of the model selection problem, they provide a more natural expression of uncertainty in the results, and they more fully capture the uncertainty. However, the specification of a prior distribution on QTL models is difficult; specifying a target false positive rate, as is done in the classical approach, is arguably more natural. Moreover, the construction of the MCMC algorithms needed for the Bayesian methods requires great care, and their use in practice may require considerable training.

R/qtl includes a number of functions for the fit and exploration of multiple-QTL models. In addition to a fully automated method (`stepwiseqtl`), there are a number of functions for exploratory analyses.

9.5 Further reading

For general discussions of model selection, see Miller (2002) and Hastie *et al.* (2009). For a review of the model selection aspects of QTL mapping, see Sillanpää and Corander (2002).

The original papers describing multiple interval mapping (Kao and Zeng, 1997; Kao *et al.*, 1999; Zeng *et al.*, 1999) used a CEM algorithm (Meng and Rubin, 1993), in which each model parameter is updated one at a time. The implementation of a full EM algorithm (Dempster *et al.*, 1977) for this problem is straightforward (Ljungberg *et al.*, 2002; Chen, 2005), but it has not been widely used. Zeng *et al.* (1999) described the technique for trimming multi-locus QTL genotypes; they also described a useful model search algorithm, on which the method described in Sec. 9.1.3 was based.

Doerge and Churchill (1996) described the use of sequential permutation tests for mapping multiple QTL in the context of forward selection. The penalized LOD score for additive QTL models is equivalent to the BIC_δ criterion proposed by Broman and Speed (2002). Bogdan *et al.* (2004) and Baierl *et al.*

(2006) proposed alternative modifications to the BIC criterion for the consideration of epistatic interactions. The penalized LOD score for multiple QTL mapping with epistasis was described in Manichaikul *et al.* (2009).

For a discussion of the most recent approaches for Bayesian QTL mapping using Markov chain Monte Carlo, see Yi (2004), Yi *et al.* (2005, 2007), and the recent review of Yi and Shriner (2008). The key software package to consider is R/qtlbim (Yandell *et al.*, 2007), which works in conjunction with R/qtl. See <http://www.qtlbim.org>.