HOS 6236 Molecular Marker Assisted Plant Breeding

QTL mapping using software R/qtl

Instructors:


```
Dr. Patricio R. Munoz
2211 Fifield Hall
352-273-4837
p.munoz@ufl.edu
Office Hours by appointment

Dr. Marcio F. Resende
2135 Fifield Hall
352-273-4772
mresende@ufl.edu
Office Hours by appointment

Dr. Kevin Folta
2239 Fifield Hall
352-273-4812
kfolta@ufl.edu
Office Hours by appointment
```


Teaching Assistant:


```
MSc. Ivone de Bem Oliveira
idebem.oliveira@ufl.edu
352-273-4836
Office Hours by appointment
```


Online resources for this tutorial

- http://www.rqtl.org

CONTENTS

TUTORIAL

Adapted from:

- A Guide to QTL Mapping with R/qtl (Broman & Sen, 2009)[1]
- Multiple-QTL Mapping (MQM) Analysis for R/qtl (Arends et al., 2014) [1]
- A shorter tour of R/qtl (Broman, 2012)[1]

The authors of the book A Guide to QTL Mapping with R/qtl book, consider the problem of mapping QTL in an experimental cross formed from two inbred lines. As they say, such crosses are the simplest populations in which we can perform QTL mapping. And also, they are the easiest to understand biologically, as well as mathematically.

Theory

- QTL are quantitative trait loci: genetic loci that contribute to variation in a quantitative trait. QTL mapping is the effort to identify QTL through an experimental cross.
- The fundamental idea underlying QTL mapping is to associate genotype and phenotype in a population exhibiting genetic variation. After to obtain phenotype data on a number of backcross or intercross progeny we try to identify regions in the genome where genotype is associated with the phenotype.
- The use of experimental crosses leads to greater control over the genetic composition of the population in experimental crosses. It helps to magnify the genetic effect of a putative QTL by judiciously choosing the crosses. This crosses also helps to randomize the genetic variation in the progeny population.
- QTL mapping in experimental crosses is an excellent first step towards more expensive investigations. The simple genetic structure of experimental crosses provides a relatively tractable framework to study the conceptual and statistical principles of genetic mapping.
- Identification of a QTL does not necessarily help us identify a gene, since the region spanned by a QTL may contain tens, and sometimes thousands, of genes.
- QTL mapping data has three interrelated data structures: the phenotypes, the genotypes, and the marker map.

- Multiple QTL Mapping (MQM) provides a sensitive approach for mapping quantitative trait loci (QTL) in experimental populations. R/qtl is a free and open source implementation of MQM, with extra features like high performance parallelization on multi-CPU computers, new plots and significance testing.
- MQM is an automatic three-stage procedure, in the first stage multiple genotypes are modeled with their estimated probabilities. In the second stage important markers are selected by multiple regression and backward elimination. In the third stage a QTL is moved along the chromosomes using these pre-selected markers as cofactors, except for the markers in the window around the interval under study. QTL are (interval) mapped using the most 'informative' model through maximum likelihood.
- The current implementation of R/qtl-MQM has the following limitations: (1) MQM is limited to experimental crosses F2, BC, and selfed RIL, (2) MQM does not treat sex chromosomes differently from autosomal chromosomes - though one can introduce sex as a cofactor. Future versions of R/qtl-MQM may improve on these points. Check the website and change log (http://www.rqtl.org/STATUS.txt) for updates.
- Despite these limitations, MQM 1 is a valuable addition to the QTL mapper's toolbox. It is able to deal with QTL in coupling phase and QTL in repulsion phase. MQM handles missing data and has higher power to detect QTL (linked and unlinked) than other methods. R/qtl's MQM is faster than other implementations and scales on multi-CPU systems and computer clusters. In this tutorial we will show you how to use MQM for QTL mapping.
- The basics steps to proceed a QTL analysis normally follows this pipeline:
- Importing the data file
- Performing diagnostic checks on the data to identify possible errors
- Performing interval mapping (or one of its variants)
- Performing a genome scan with a single-QTL model, to detect loci with important marginal effects.
- Performing statistical significance tests of putative QTL
- The result of the analysis is some set of inferred QTL, with some understanding of their effects, locations, and possible interactions. These results will be used to guide further experiments, perhaps with the aim of fine-mapping the QTL and ultimately identifying the underlying gene or genes.

Starting the practice

1 Installing and loading the package

```
# install.packages('qtl')
require('qtl')

## Loading required package: qtl
```

2 Citation

```
citation('qtl')

##
## To cite R/qtl in publications use:
##
##   Broman et al. (2003) R/qtl: QTL mapping in experimental crosses.
##   Bioinformatics 19:889-890
##
## A BibTeX entry for LaTeX users is
##
##   @Article{Broman2003,
##     title = {R/qtl: {QTL} mapping in experimental crosses},
##     author = {Karl W. Broman and Hao Wu and Saunak Sen and Gary A.
Churchill},
##     journal = {Bioinformatics},
##     year = {2003},
##     volume = {19},
##     pages = {889-890},
##   }
```

3 Data files

- Examples of files are provided on the R/qtl web site.
- Importing QTL mapping data into R is accomplished with the read.cross function.
- Please go to section 2.1 of Broman & Sen (2009).

4 Practical Example

Following the Tutorial: A shorter tour of R/qtl (Broman, 2012)

- We will consider data from Sugiyama et al., Physiol Genomics 10:5–12, 2002. The data are from an intercross between BALB/cJ and CBA/CaJ; only male offspring were considered. There are four phenotypes: blood pressure, heart rate, body weight, and heart weight. We will focus on the blood pressure phenotype, will consider just the 163 individuals with genotype data and, for simplicity, will focus on the autosomes. The data are contained in the comma-delimited file http://www.rqtl.org/sug.csv.

4.1 Loading data

```
sug <- read.cross("csv", "http://www.rqtl.org", "sug.csv",
                  genotypes=c("CC", "CB", "BB"), alleles=c("C", "B"))

##  --Read the following data:
##   163  individuals
##   93  markers
```

```
##   6  phenotypes
##  --Cross type: f2
```

- The function read.cross is for importing data into R/qtl. "sug.csv" is the name of the file, which we import directly from the R/qtl website. Genotypes indicates the codes used for the genotypes; alleles indicates single-character codes to be used in plots and such.

4.2 Diagnostics

- Generally, at this point, one would spend considerable time studying the genotype and phenotype data, looking for potential errors. In many cases, about half of the analysis time is devoted to such diagnostics. In previous tutorials, we've often gotten bogged down in this part, and so we'll skip it here, assume that the data are okay, and jump right into QTL mapping. See the longer ("brief") tour of R/qtl at http://www.rqtl.org/tutorials, or Chapter 3 of Broman and Sen (2009).

4.3 Summaries

- The data object "sug" is complex; it contains the genotype data, phenotype data and genetic map. R has a certain amount of "object oriented" facilities, so that calls to functions like summary and plot are interpreted appropriately for the object considered. The object sug has "class" "cross", and so calls to summary and plot are actually sent to the functions summary.cross and plot.cross.

4.4 Quick summary of the data

(performs a variety of checks of the integrity of the data)

```
summary(sug)
```

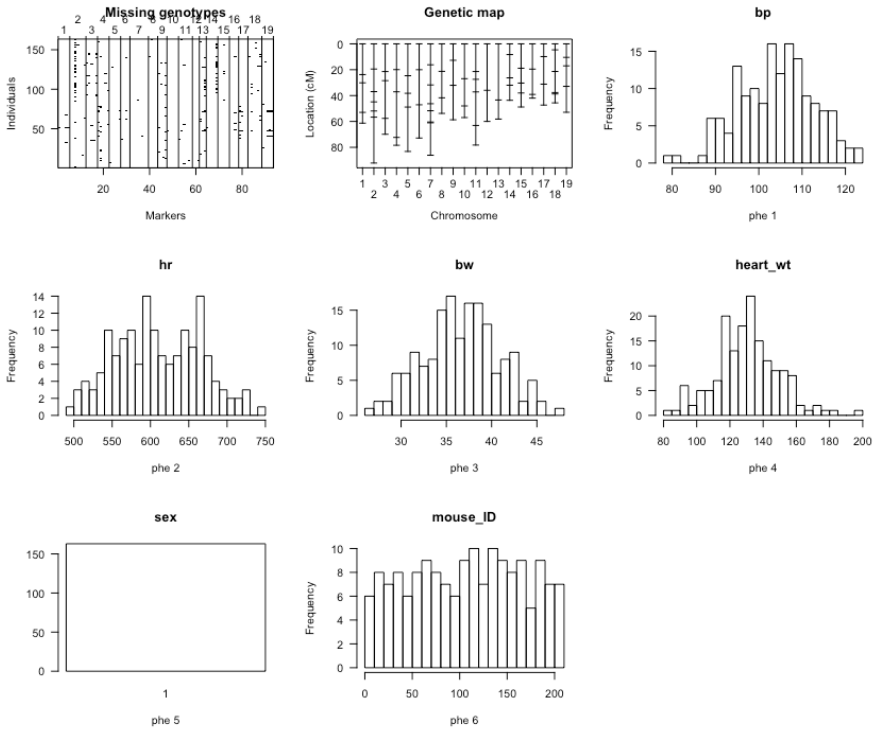
```
##     F2 intercross
##
##     No. individuals:    163
##
##     No. phenotypes:     6
##     Percent phenotyped: 95.1 95.7 99.4 99.4 100 100
##
##     No. chromosomes:    19
##         Autosomes:      1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
19
##
##     Total markers:      93
##     No. markers:        5 7 5 5 5 4 8 4 4 5 6 3 3 5 5 4 4 6 5
##     Percent genotyped:  98.3
##     Genotypes (%):      CC:23.9  CB:50.2  BB:26.0  not BB:0.0  not
CC:0.0
```

- We see that this is an intercross with 163 individuals. There are 6 phenotypes, and genotype data at 93 markers across the 19 autosomes. The genotype data is quite complete.
- There are a number of simple functions for pulling out pieces of summary information. Hopefully these are self-explanatory.

```
data.frame(
    nind(sug),
    nchr(sug),
    totmar(sug),
    nphe(sug))

##   nind.sug. nchr.sug. totmar.sug. nphe.sug.
## 1       163        19          93         6
```

Get a summary plot of the data.

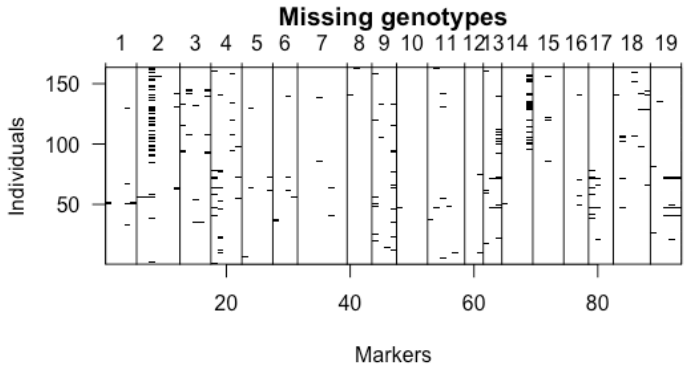
```
plot(sug)
```



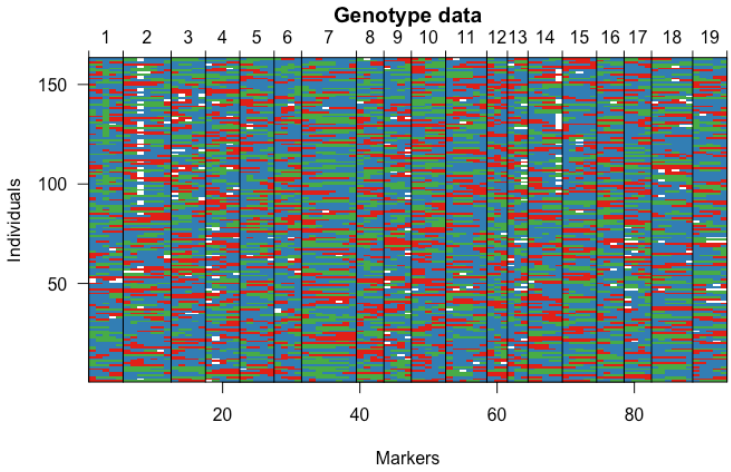
- The plot in the upper-left shows the pattern of missing genotype data, with black pixels corresponding to missing genotypes. The next plot shows the genetic map of the typed markers. The following plots are histograms or bar plots for the six phenotypes. The last two "phenotypes" are sex (with 1 corresponding to males) and mouse ID.

4.4.1 Individual parts of the above plot may be obtained as follows.

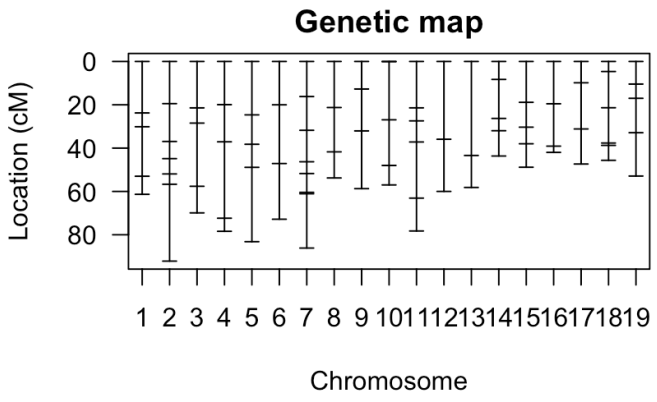
```
plotMissing(sug)
```



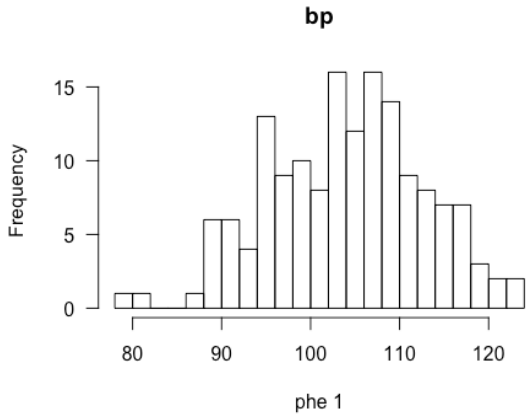

```
geno.image(sug)
```




```
plotMap(sug)
```




```
plotPheno(sug, pheno.col=1) # the same can be done for the other
phenotypes
```



- Before going to the next step (the QTL genome scan), the data has to be completed (i.e. no more missing data). There are two possibilities: use (1) the MQM data augmentation routine mqmaugment or (2) the imputation routine fill.geno. These models help us probabilistically connect unobserved genotypes to observed genotypes
- We should emphasize that we also assume that there is no segregation distortion

4.5 Data augmentation

- In an ideal world all datasets would be complete (with the genotype for every individual at every marker determined), however in the real world datasets are often incomplete. That is, genotype information is missing, or can have multiple plausible values. MQM automatically expands the dataset by adding all potential variants and attaching a probability to each. For example, information is missing (unknown) at a marker location for one individual. Based on the values of the neighbouring markers, and the (estimated) recombination rate, a probability is

attached to all possible genotypes. With MQM all possible genotypes with a probability above the parameter minprob are considered.

- Before going to the QTL genome scan, the data has to be completed (i.e. no more missing data). There are two possibilities: use (1) the MQM data augmentation routine mqmaugment or (2) the imputation routine fill.geno. Augmentation tries to analyze all possible genotypes of interest by leaving them in the solution space. In contrast, the imputation method selects the most likely genotype, and uses that single individual for further analysis.

- In the second augmentation round the user can specify what needs to be done with these individuals: (1) Only use the most likely genotype, (2) use multiple imputation to create multiple possible genotypes (up to maxaugind) or (3) remove the original genotype/individual from the analysis. Note that you can opt to use fill.geno's imputation method on your dataset, instead of augmentation, when too many individuals are dropped because of missing data.

- The function mqmaugment is specific to MQM and the recommended procedure3. In this tutorial we focus on MQM 's augmentation. The function mqmaugment fills in missing genotypes for us. For each missing genotype data, at a marker, it fills in all possible genotypes and calculates the probability. When the total probability is higher than the minprob parameter the augmented individual is stored in the new cross object, ready for QTL mapping.

4.5.1 Augmented data using mqmaugment

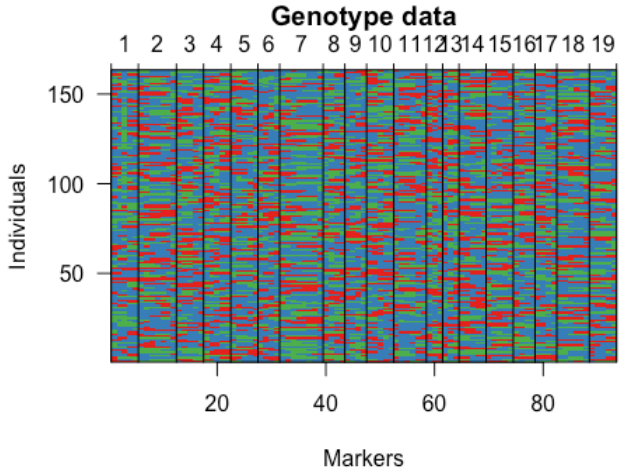
```
sug_min1 <- mqmaugment(sug, minprob=1.0, verbose = TRUE)

## INFO: Received a valid cross file type: f2 .
## INFO: Number of individuals: 163 .
## INFO: Number of chr: 19 .
## INFO: Number of markers: 93 .
## INFO: Done with augmentation
## # Unique individuals before augmentation:163
## # Unique selected individuals:163
## # Marker p individual:93
## # Individuals after augmentation:163
## INFO: Data augmentation succesfull
## INFO: DATA-Augmentation took:  0.126  seconds
```

- With a lower minprob, more augmented individuals are kept, and the resulting augmented dataset will be larger. Adding (weighted) augmented individuals with all possible genotypes theoretically leads to a more accurate mapping when dealing with missing values.

- By using the imputation strategy the individuals are kept in the set with a single 'most likely' genotype

4.5.2 Plot the data after augmentation

```
geno.image(sug_min1)
```



4.6 Single-QTL analysis

- The first step is to calculate the QTL genotype probabilities, given the observed marker data, via the function calc.genoprob. This is done at the markers and at a grid along the chromosomes. The argument step is the density of the grid (in cM), and defines the density of later QTL analyses.

```
sug <- calc.genoprob(sug_min1, step=1)
```

- The output of calc.genoprob is the same cross object as input, with additional information (the QTL genotype probabilities) inserted. We assign this back to the original object (writing over the previous data), though it could have also been assigned to a new object.

- To perform a single-QTL genome scan, swe use the function scanone. By default, it performs standard interval mapping (that is, maximum likelihood via the EM algorithm). Also, by default, it considers the first phenotype in the input cross object (in this case, blood pressure).

```
out.em <- scanone(sug)

## Warning in checkcovar(cross, pheno.col, addcovar, intcovar,
perm.strata, : Dropping 8 individuals with missing phenotypes.
```

The output has “class” "scanone". The summary function is passed to the function summary.scanone, and gives the maximum LOD score on each chromosome.

```
summary(out.em)
```

```
##           chr   pos   lod
## D1MIT36     1 76.73 1.485
## c2.loc77    2 82.80 1.865
## c3.loc43    3 53.82 1.425
## c4.loc44    4 48.23 0.800
## D5MIT223    5 86.57 1.303
## c6.loc26    6 27.81 0.633
## c7.loc45    7 47.71 6.119
## c8.loc34    8 54.90 1.598
## D9MIT71     9 27.07 0.768
## c10.loc51  10 60.75 0.959
## c11.loc36  11 40.70 2.156
## D12MIT145  12  2.23 1.472
## c13.loc19  13 26.26 0.981
## D14MIT138  14 12.52 1.092
## c15.loc8   15 11.96 5.257
## c16.loc32  16 46.69 0.661
## D17MIT16   17 17.98 1.283
## D18MIT22   18 13.41 1.744
## D19MIT71   19 56.28 0.499
```

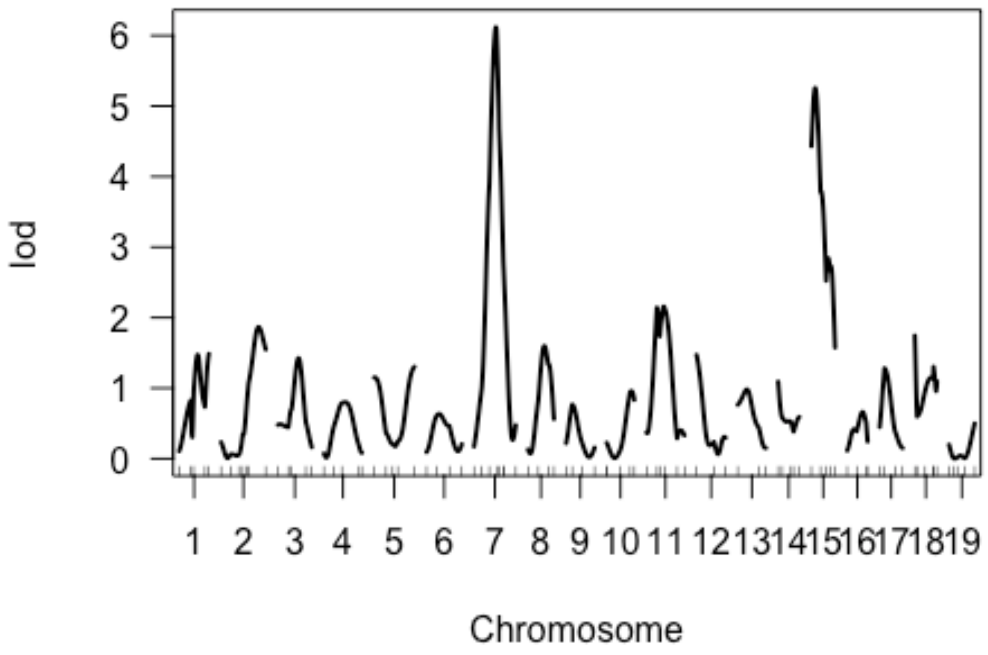
4.6.1 To only see those chromosomes with LOD > 3

```
summary(out.em, threshold=3)
```

```
##          chr  pos  lod
## c7.loc45   7 47.7 6.12
## c15.loc8  15 12.0 5.26
```

4.6.2 Plot the results

```
plot(out.em)
```



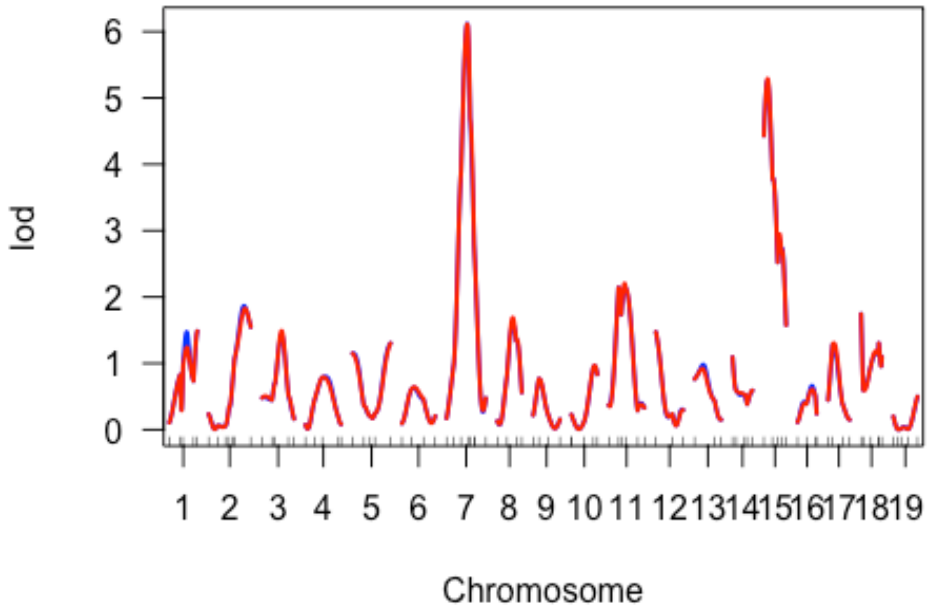
4.6.3 Genome scan via Haley-Knott regression

- By calling scanone with the argument method="hk".

```
out.hk <- scanone(sug, method="hk")

## Warning in checkcovar(cross, pheno.col, addcovar, intcovar,
perm.strata, : Dropping 8 individuals with missing phenotypes.
```

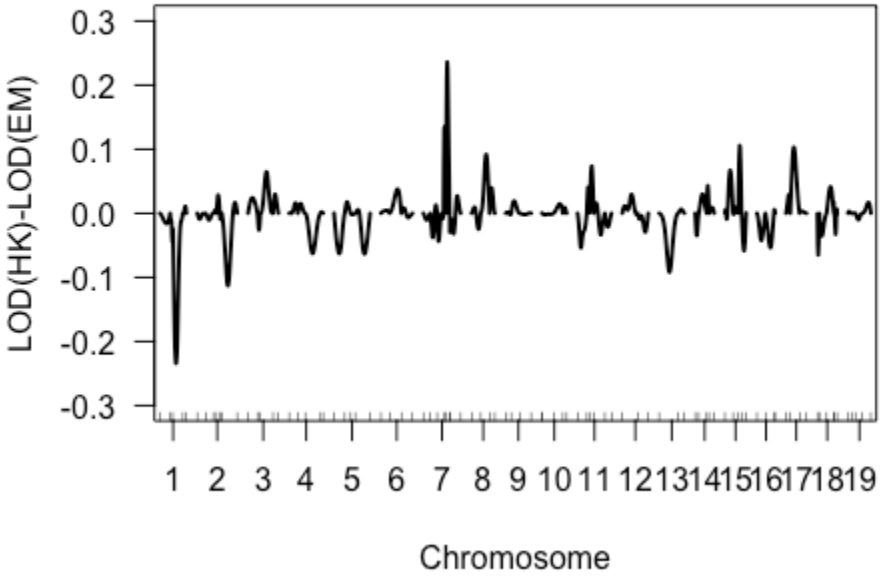
- We may plot the two sets of LOD curves together in a single call to plot.

```
plot(out.em, out.hk, col=c("blue", "red"))
```



- It's perhaps more informative to plot the differences:

```
plot(out.hk - out.em, ylim=c(-0.3, 0.3), ylab="LOD(HK)-LOD(EM)")
```



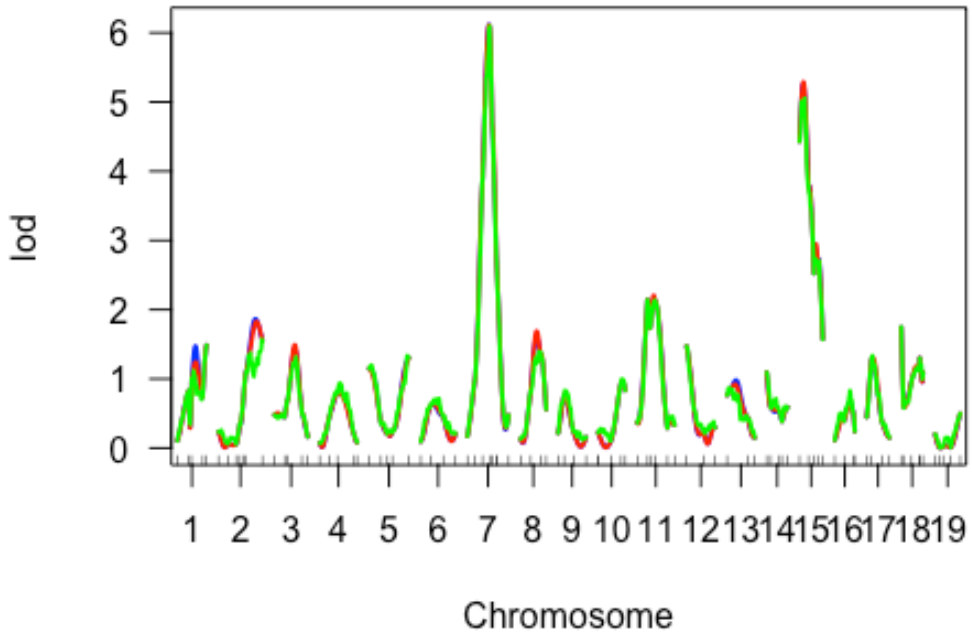
- To perform a genome scan by the multiple imputation method, one must first call sim.geno to perform the multiple imputations. This is similar to calc.genoprob, but with an additional argument, n.draws, indicating the number of imputations. We then call scanone with method="imp".

```
sug <- sim.geno(sug, step=1, n.draws=64)
out.imp <- scanone(sug, method="imp")
```

```
## Warning in checkcovar(cross, pheno.col, addcovar, intcovar,
perm.strata, : Dropping 8 individuals with missing phenotypes.
```

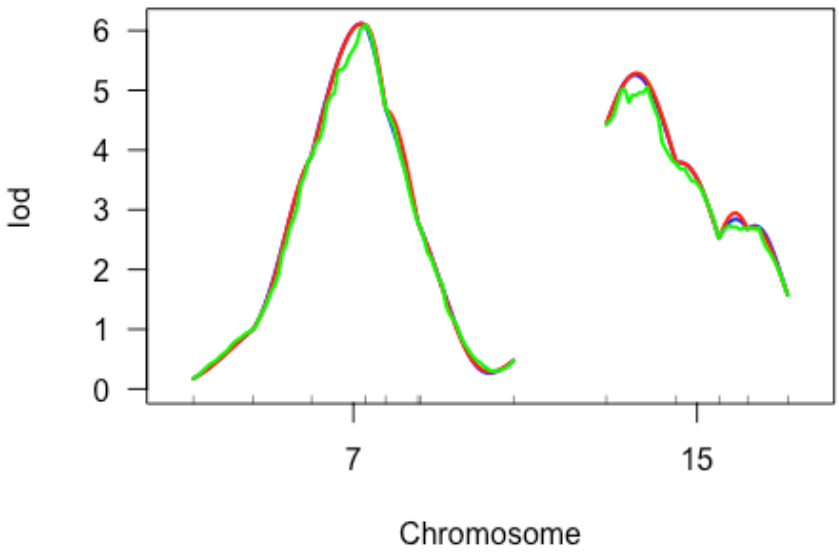
- We may plot all three curves together as follows.

```
plot(out.em, out.hk, out.imp, col=c("blue", "red", "green"))
```



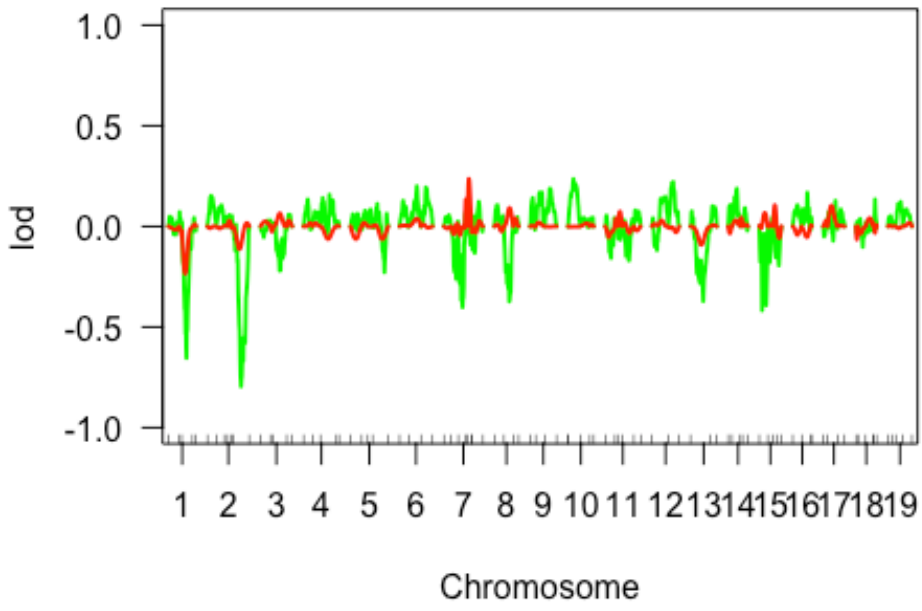
- We can plot the LOD curves for just chromosomes 7 and 15 as follows.

```
plot(out.em, out.hk, out.imp, col=c("blue", "red", "green"), chr=c(7,15))
```



- We can also look at differences.

```
plot(out.imp - out.em, out.hk - out.em, col=c("green", "red"), ylim=c(-
1,1))
```



4.6.4 Permutation tests

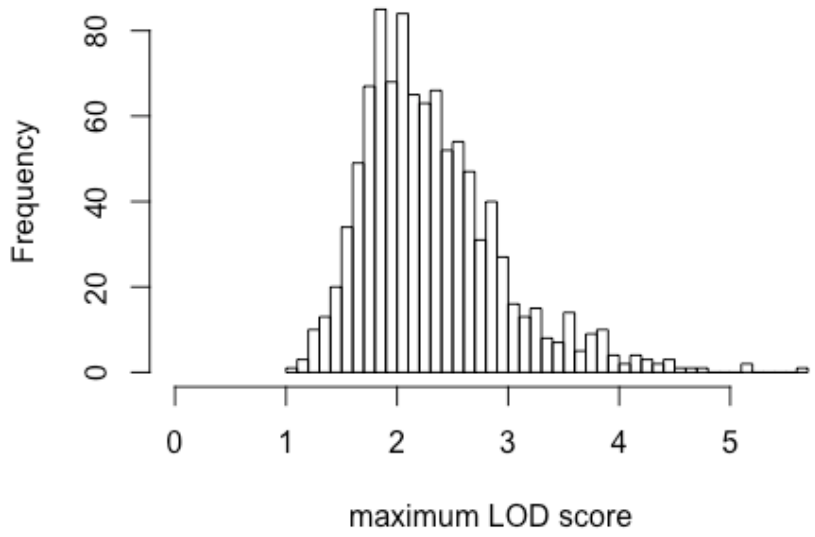
To perform a permutation test, to get a genome-wide significance threshold or genome-scan-adjusted p-values, we use scanone just as before, but with an additional argument, n.perm, indicating the number of permutation replicates. It's quickest to use Haley-Knott regression.

- The code to do the actual permutation test is the following:

```
operm <- scanone(sug, method="hk", n.perm=1000)

## Warning in checkcovar(cross, pheno.col, addcovar, intcovar,
perm.strata, : Dropping 8 individuals with missing phenotypes.

## Doing permutation in batch mode ...
```

- A histogram of the results (the 1000 genome-wide maximum LOD scores) is obtained as follows:

```
plot(operm)
```



- Significance thresholds may be obtained via the summary function: summary(operm)

```
summary(operm, alpha=c(0.05, 0.2))

## LOD thresholds (1000 permutations)
##      lod
## 5%  3.59
## 20% 2.77
```

- Most importantly, the permutation results may be used along with the scanone results to have significance thresholds and p-values calculated automatically:

```
summary(out.hk, perms=operm, alpha=0.2, pvalues=TRUE)

##          chr  pos  lod  pval
## c7.loc45   7 47.7 6.11 0.000
## c15.loc8  15 12.0 5.29 0.001
```

4.7 Interval estimates of QTL location

- For the blood pressure phenotype, we've seen good evidence for QTL on chromosomes 7 and 15. Interval estimates of the location of QTL are commonly obtained via 1.5-LOD support intervals, which may be calculated via the function lodint. Alternatively, an approximate Bayes credible interval may be obtained with bayesint.

- To obtain the 1.5-LOD support interval and 95% Bayes interval for the QTL on chromosome 7, type:

```
lodint(out.hk, chr=7)
```

```
##          chr   pos      lod
## c7.loc34   7 36.71 4.406220
## c7.loc45   7 47.71 6.110832
## c7.loc54   7 56.71 4.505219

bayesint(out.hk, chr=7)

##          chr   pos      lod
## c7.loc37   7 39.71 5.090979
## c7.loc45   7 47.71 6.110832
## c7.loc50   7 52.71 5.380157
```

- The first and last rows define the ends of the intervals; the middle row is the estimated QTL location.
- It is sometimes useful to identify the closest flanking markers; use expandtomarkers=TRUE:

```
lodint(out.hk, chr=7, expandtomarkers=TRUE)

##          chr   pos      lod
## D7MIT176   7 34.48 3.894345
## c7.loc45   7 47.71 6.110832
## D7MIT7     7 63.14 2.800148

bayesint(out.hk, chr=7, expandtomarkers=TRUE)

##          chr   pos      lod
## D7MIT176   7 34.48 3.894345
## c7.loc45   7 47.71 6.110832
## D7MIT323   7 54.45 4.690901
```

- We can calculate the 2-LOD support interval and the 99% Bayes interval as follows.

```
lodint(out.hk, chr=7, drop=2)

##          chr   pos      lod
## c7.loc32   7 34.71 3.946043
## c7.loc45   7 47.71 6.110832
## c7.loc57   7 59.71 3.849874

bayesint(out.hk, chr=7, prob=0.99)

##          chr   pos      lod
## c7.loc34   7 36.71 4.406220
## c7.loc45   7 47.71 6.110832
## c7.loc54   7 56.71 4.505219
```

- The intervals for the chr 15 locus may be calculated as follows.

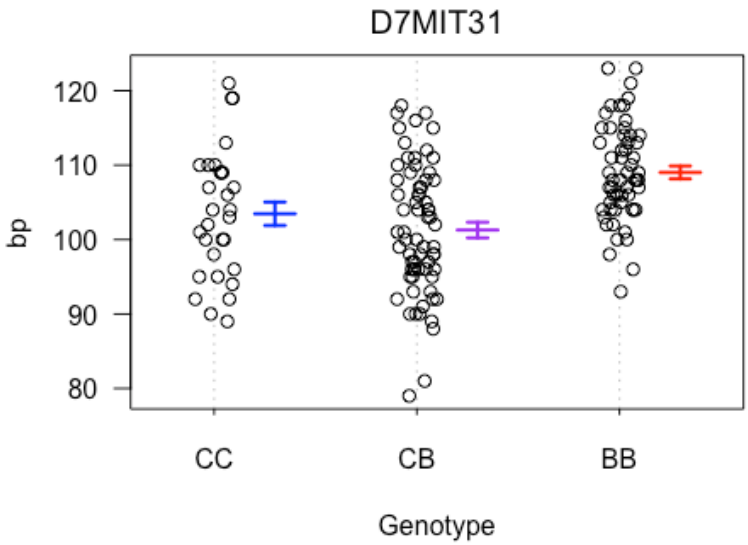
```
lodint(out.hk, chr=15)
```

```
##           chr   pos      lod
## D15MIT175  15  3.96 4.432504
## c15.loc8   15 11.96 5.290115
## c15.loc21  15 24.96 3.776696

bayesint(out.hk, chr=15)

##           chr   pos      lod
## D15MIT175  15  3.96 4.432504
## c15.loc8   15 11.96 5.290115
## c15.loc16  15 19.96 4.373699
```

4.7.1 QTL effects

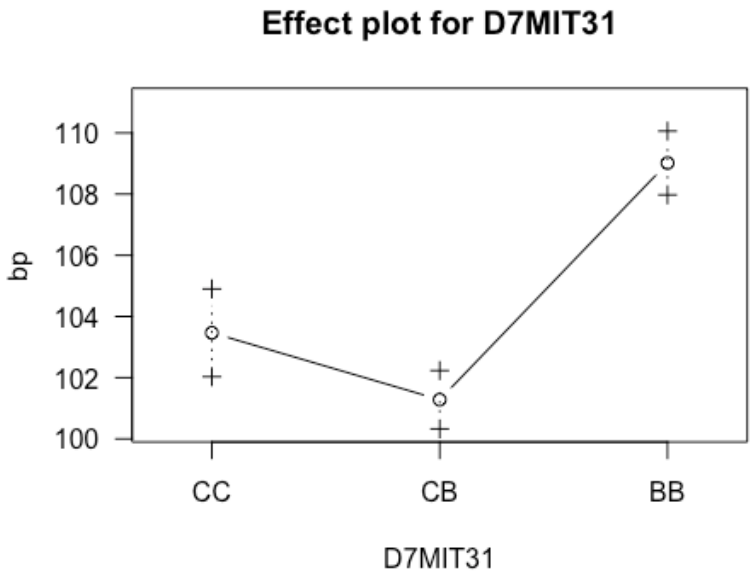
- We may obtain plots indicating the estimated effects of the QTL via plotPXG, which creates a dot plot, or effectplot, which plots the average phenotype for each genotype group. For plotPXG, we must first identify the marker closest to the QTL peak. Use find.marker.

```
max(out.hk)

##          chr  pos  lod
## c7.loc45   7 47.7 6.11

mar <- find.marker(sug, chr=7, pos=47.7)

plotPXG(sug, marker=mar)
```



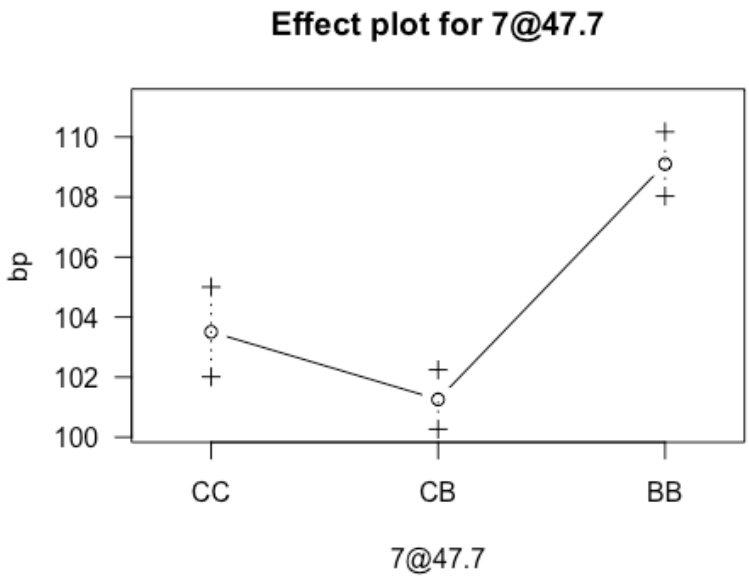
- Note that red dots correspond to inferred genotypes (based on a single imputation).
- The function effectplot uses the multiple imputation results from sim.geno.

```
effectplot(sug, mname1=mar)
```



- We may use effectplot at a position on the "grid" between markers, using "7@47.7" to indicate the position at 47.7 cM on chr 7.

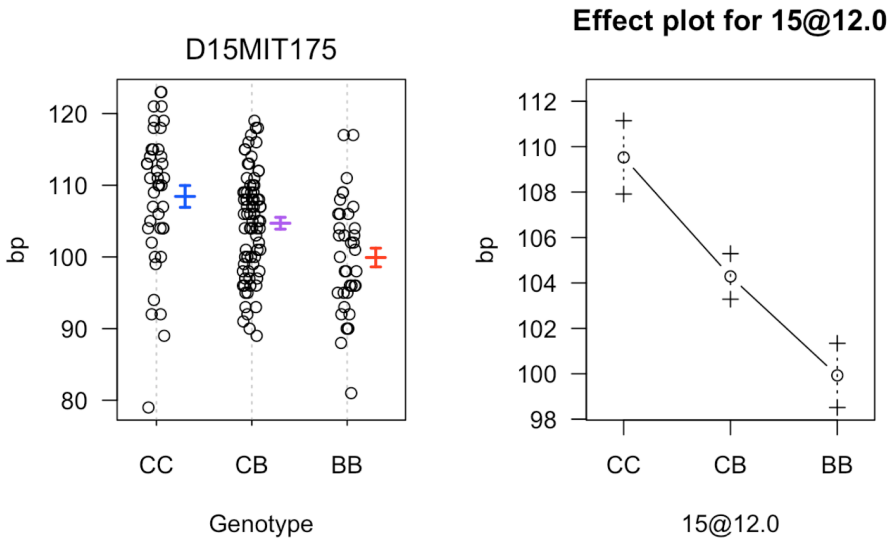
```
effectplot(sug, mname1="7@47.7")
```



- Similar plots may be obtained for the locus on chr 15.

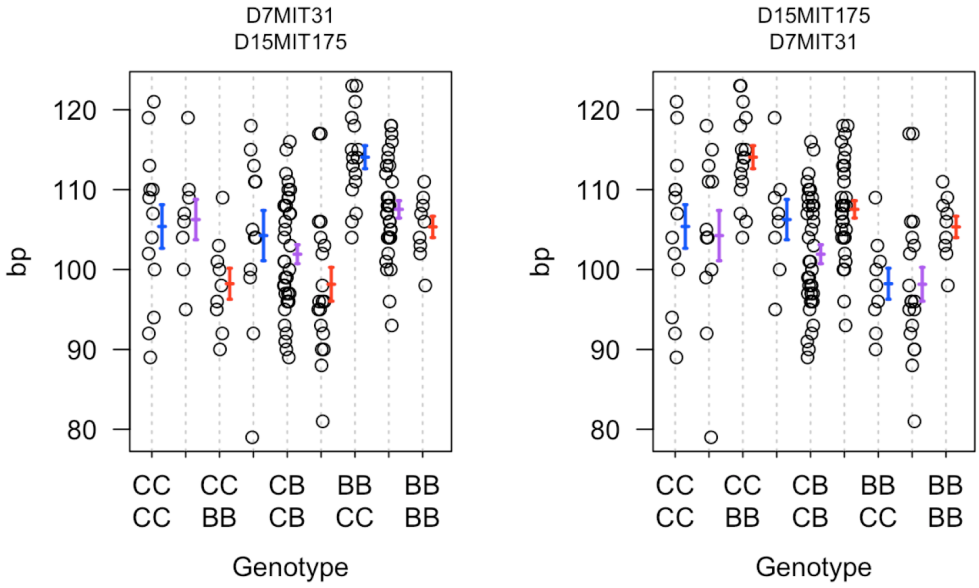
```
max(out.hk, chr=15)

##          chr pos  lod
## c15.loc8  15  12 5.29

mar2 <- find.marker(sug, chr=15, pos=12)
```

```
par(mfrow=c(1,2))
plotPXG(sug, marker=mar2)
effectplot(sug, mname1="15@12")
```



- We may plot the joint effects of the two loci via plotPXG as follows:

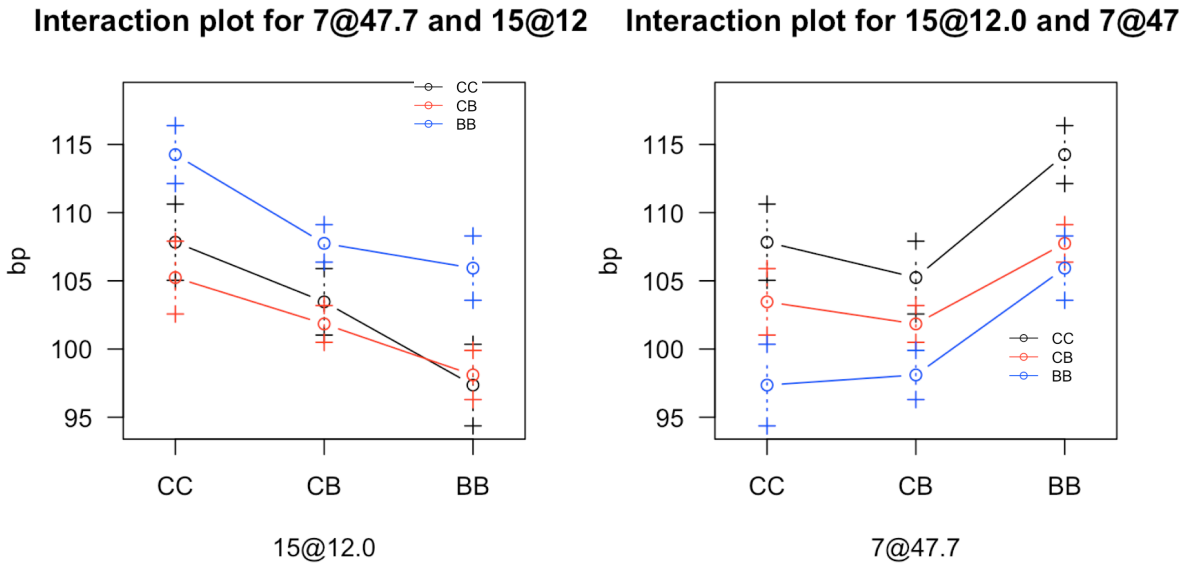
```
par(mfrow=c(1,2))
plotPXG(sug, marker=c(mar, mar2))
plotPXG(sug, marker=c(mar2, mar))
```



- The function effectplot gives more readable figures in this case; it's often useful to look at it in both ways.

```
par(mfrow=c(1,2))
```

```
effectplot(sug, mname1="7@47.7", mname2="15@12")
effectplot(sug, mname2="7@47.7", mname1="15@12")
```



- The two loci do not appear to interact.

4.8 Multiple-QTL analyses

- After performing the single- and two-QTL genome scans, it's best to bring the identified loci together into a joint model, which we then refine from which we may explore the possibility of further QTL. In this effort, we work with "QTL objects" created by makeqtl. We fit multiple-QTL models with fitqtl. A number of additional functions will be introduced below.
- Let's re-run calc.genoprob so that we are working at a step size of 1 cM again.

```
sug <- calc.genoprob(sug, step=1)
```

- First, we create a QTL object containing the loci on chr 7 and 15.

```
qtl <- makeqtl(sug, chr=c(7,15), pos=c(47.7, 12), what="prob")
```

- The last argument, what="prob", indicates to pull out the QTL genotype probabilities for use in Haley-Knott regression.
- We fit the two locus additive model as follows.

```
out.fq <- fitqtl(sug, qtl=qtl, method="hk")

## Warning in fitqtlengine(pheno = pheno, qtl = qtl, covar = covar,
formula = formula, : Dropping 8 individuals with missing phenotypes.

summary(out.fq)
```

```
##
##      fitqtl summary
##
## Method: Haley-Knott regression
## Model:  normal phenotype
## Number of observations : 155
##
## Full model result
## ----------------------------------
## Model formula: y ~ Q1 + Q2
##
##        df        SS        MS      LOD    %var Pvalue(Chi2)
Pvalue(F)
## Model   4  3209.294 802.32346 11.04233 27.9692  2.39715e-10 4.527265e-
10
## Error 150  8265.093  55.10062
## Total 154 11474.387
##
##
## Drop one QTL at a time ANOVA table:
## ----------------------------------
##         df Type III SS   LOD  %var F value Pvalue(Chi2) Pvalue(F)
## 7@47.7   2        1540 5.752 13.42   13.98            0  2.71e-06 ***
## 15@12.0  2        1304 4.932 11.37   11.83            0  1.69e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- A key part of the output is the "drop one term at a time" table, which compares the fit of the two-QTL model to the reduced model in which a single QTL is omitted.
- We may obtain the estimated effects of the QTL via get.ests=TRUE. We use dropone=FALSE to suppress the drop-one-term analysis.

```
summary(fitqtl(sug, qtl=qtl, method="hk", get.ests=TRUE, dropone=FALSE))
```

```
## Warning in fitqtlengine(pheno = pheno, qtl = qtl, covar = covar,
formula = formula, : Dropping 8 individuals with missing phenotypes.

##
##      fitqtl summary
##
## Method: Haley-Knott regression
## Model:  normal phenotype
## Number of observations : 155
##
## Full model result
## ----------------------------------
## Model formula: y ~ Q1 + Q2
##
##        df        SS        MS      LOD    %var Pvalue(Chi2)
```

```
Pvalue(F)
## Model   4  3209.294 802.32346 11.04233 27.9692  2.39715e-10 4.527265e-
10
## Error 150  8265.093  55.10062
## Total 154 11474.387
##
##
## Estimated effects:
## -----------------
##                est       SE       t
## Intercept 103.7915   0.6151 168.738
## 7@47.7a     3.0160   0.8720   3.459
## 7@47.7d    -4.0799   1.3119  -3.110
## 15@12.0a   -4.7848   0.9838  -4.864
## 15@12.0d   -0.3761   1.5625  -0.241
```

- Since this is an intercross, we obtain estimates of the additive effect and dominance deviation for each locus.
- To assess the possibility of an interaction between the two QTL, we may fit the model with the interaction, indicated via a model "formula". The QTL are referred to as Q1 and Q2 in the formula, and we may indicate the interaction in a couple of different ways.

```
out.fqi1 <- fitqtl(sug, qtl=qtl, method="hk", formula=y~Q1*Q2)

## Warning in fitqtlengine(pheno = pheno, qtl = qtl, covar = covar,
formula = formula, : Dropping 8 individuals with missing phenotypes.

out.fqi2 <- fitqtl(sug, qtl=qtl, method="hk", formula=y~Q1+Q2+Q1:Q2)

## Warning in fitqtlengine(pheno = pheno, qtl = qtl, covar = covar,
formula = formula, : Dropping 8 individuals with missing phenotypes.
```

First way

```
summary(out.fqi1)

##
##      fitqtl summary
##
## Method: Haley-Knott regression
## Model:  normal phenotype
## Number of observations : 155
##
## Full model result
## ----------------------------------
## Model formula: y ~ Q1 + Q2 + Q1:Q2
##
##        df        SS        MS      LOD     %var Pvalue(Chi2)
Pvalue(F)
```

```
## Model   8  3419.831 427.47886 11.91081 29.80404 4.718638e-09 1.2377e-
08
## Error 146  8054.556  55.16819
## Total 154 11474.387
##
##
## Drop one QTL at a time ANOVA table:
## ---------------------------------
##                df Type III SS    LOD   %var F value Pvalue(Chi2)
Pvalue(F)
## 7@47.7          6      1750.9 6.6207 15.260  5.2897        0.000
5.81e-05
## 15@12.0         6      1514.7 5.8000 13.201  4.5761        0.000
0.000276
## 7@47.7:15@12.0  4       210.5 0.8685  1.835  0.9541        0.406
0.434757
##
## 7@47.7         ***
## 15@12.0        ***
## 7@47.7:15@12.0
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Second way

```
summary(out.fqi2)

##
##      fitqtl summary
##
## Method: Haley-Knott regression
## Model:  normal phenotype
## Number of observations : 155
##
## Full model result
## ---------------------------------
## Model formula: y ~ Q1 + Q2 + Q1:Q2
##
##        df        SS        MS      LOD     %var Pvalue(Chi2)
Pvalue(F)
## Model   8  3419.831 427.47886 11.91081 29.80404 4.718638e-09 1.2377e-
08
## Error 146  8054.556  55.16819
## Total 154 11474.387
##
##
## Drop one QTL at a time ANOVA table:
## ---------------------------------
##                df Type III SS    LOD   %var F value Pvalue(Chi2)
Pvalue(F)
## 7@47.7          6      1750.9 6.6207 15.260  5.2897        0.000
```

```
5.81e-05
## 15@12.0          6      1514.7 5.8000 13.201  4.5761        0.000
0.000276
## 7@47.7:15@12.0   4       210.5 0.8685  1.835  0.9541        0.406
0.434757
##
## 7@47.7          ***
## 15@12.0         ***
## 7@47.7:15@12.0
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Note that there is no evidence for an interaction.
- Another way to assess interactions is with the function addint, which adds one interaction at a time, in the context of a multiple-QTL model. This is most useful when there are more than two QTL being considered.

```
addint(sug, qtl=qtl, method="hk")

## Warning in addint(sug, qtl = qtl, method = "hk"): Dropping 8
individuals with missing phenotypes.

## Method: Haley-Knott regression
## Model:  normal phenotype
## Model formula: y ~ Q1 + Q2
##
## Add one pairwise interaction at a time table:
## ---------------------------------------------
##                 df Type III SS    LOD  %var F value Pvalue(Chi2)
Pvalue(F)
## 7@47.7:15@12.0   4       210.5 0.8685 1.835  0.9541        0.406
0.435
```

- The locations of the two QTL are as estimated via the single-QTL scan. We may refine our estimates of QTL location in the context of the multiple-QTL model via refineqtl. This function uses a greedy algorithm to iteratively refines the locations of the QTL, one at a time, at each step seeking to improve the overall fit.

```
rqtl <- refineqtl(sug, qtl=qtl, method="hk")

## pos: 47.71 11.96
## Iteration 1
##  Q2 pos: 11.96 -> 12.96
##     LOD increase:  0.024
##  Q1 pos: 47.71 -> 46.71
##     LOD increase:  0.005
## all pos: 47.71 11.96 -> 46.71 12.96
## LOD increase at this iteration:  0.029
## Iteration 2
##  Q1 pos: 46.71 -> 46.71
##     LOD increase:  0
```

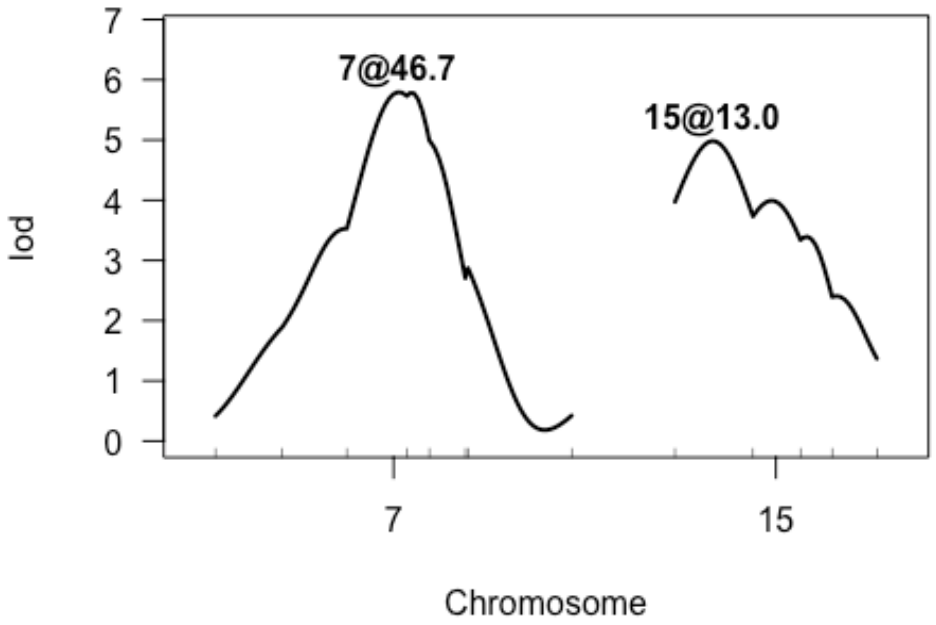
```
##  Q2 pos: 12.96 -> 12.96
##     LOD increase:  0
## all pos: 46.71 12.96 -> 46.71 12.96
## LOD increase at this iteration:  0
## overall pos: 47.71 11.96 -> 46.71 12.96
## LOD increase overall:  0.029

rqtl

##   QTL object containing genotype probabilities.
##
##       name chr   pos n.gen
## Q1  7@46.7   7 46.71     3
## Q2 15@13.0  15 12.96     3
```

- The location of each QTL changed slightly, and the overall LOD score increased by 0.03.
- We can re-run fitqtl to get the revised drop-one-term table.

```
summary(out.fqr <- fitqtl(sug, qtl=rqtl, method="hk"))

## Warning in fitqtlengine(pheno = pheno, qtl = qtl, covar = covar,
formula = formula, : Dropping 8 individuals with missing phenotypes.

##
##      fitqtl summary
##
## Method: Haley-Knott regression
## Model:  normal phenotype
## Number of observations : 155
##
## Full model result
## ----------------------------------
## Model formula: y ~ Q1 + Q2
##
##        df        SS       MS      LOD     %var Pvalue(Chi2)
Pvalue(F)
## Model   4  3216.377 804.0943 11.07119 28.03093  2.24868e-10 4.254246e-
10
## Error 150  8258.010  55.0534
## Total 154 11474.387
##
##
## Drop one QTL at a time ANOVA table:
## ----------------------------------
##         df Type III SS   LOD  %var F value Pvalue(Chi2) Pvalue(F)
## 7@46.7   2        1551 5.793 13.52   14.09            0  2.48e-06 ***
## 15@13.0  2        1318 4.984 11.49   11.97            0  1.50e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

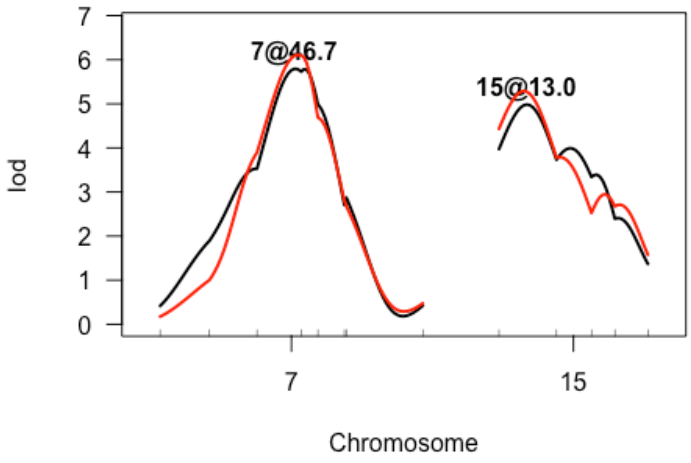
- The plotLodProfile function plots LOD profiles obtained during the call to refine qtl.These give one-dimensional views of the precision of QTL localization, in the context of the multiple-QTL model.

```
plotLodProfile(rqtl)
```



- For each position on the curve for the chr 7 QTL, we compare the two-QTL model with the chr 7 locus in varying position but with the chr 15 locus fixed at its estimated position, to the single-QTL model with just the chr 15 locus. The chr 15 curve is similar. These are actually slightly lower than the curves obtained from the single-QTL analysis with scanone.

```
plotLodProfile(rqtl)
plot(out.hk, chr=c(7,15), col="red", add=TRUE)
```

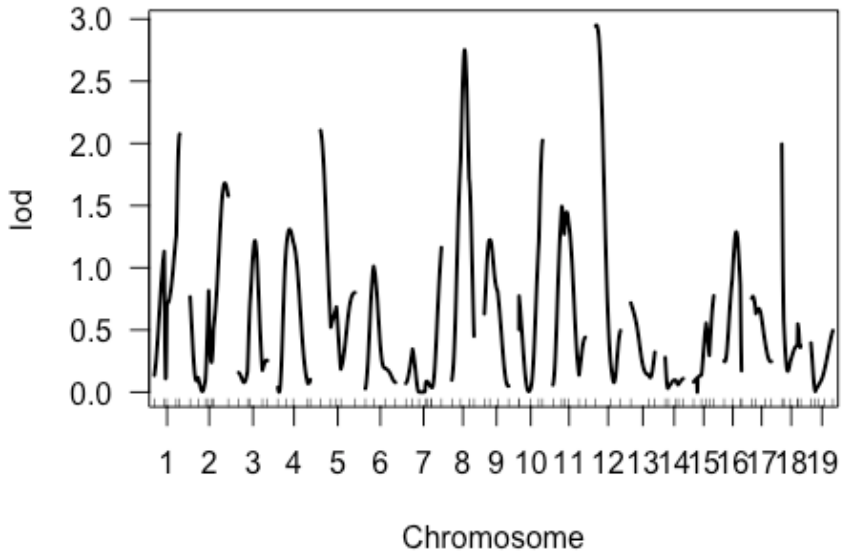


- The function addqtl is used to scan for an additional QTL to be added to the model.

```
out.aq <- addqtl(sug, qtl=rqtl, method="hk")

## Warning in addqtl(sug, qtl = rqtl, method = "hk"): Dropping 8
individuals with missing phenotypes.
```

- The biggest peaks are on chr 8 and 12, but nothing is particularly exciting.

```
plot(out.aq)
```



- There is also a function addpair, for scanning for a pair of QTL to be added.
- We can also consider the function stepwiseqtl, which is our fully automated stepwise algorithm to optimize the penalized LOD scores of Manichaikul et al. (2009). For this see the (Broman, 2012)

References

- Arends et al. (2014) Multiple-QTL Mapping (MQM) Analysis for R/qtl. Available at: http://www.rqtl.org/tutorials/
- Broman KW, Sen S´ (2009) A guide to QTL mapping with R/qtl. Springer.
- Broman KW, (2012) A shorter tour of R/qtl. Available at: http://www.rqtl.org/tutorials/
- Dalgaard P (2008) Introductory statistics with R, 2nd edition. Springer.
- Manichaikul A, Moon JY, Sen S´, Yandell BS, Broman KW (2009) A model selection approach for the identification of quanti- tative trait loci in

experimental crosses, allowing epistasis. Genetics 181:1077–1086.

- Sugiyama F, Churchill GA, Li R, Libby LJM, Carver T, Yagami K-I, John SWM, Paigen B (2002) QTL associated with blood pressure, heart rate, and heart weight in CBA/CaJ and BALB/cJ mice. Physiol Genomics 10:5–12.
- Venables WN, Ripley BD (2002) Modern applied statistics with S, 4th edition. Springer.