The University of Newcastle

Kerrie Mengersen

Introduction to Bayesian Methods for QTL Analysis - 7

Armidale 2004

Software for QTL analysis

- Input into BDA
  - Mapmaker
- Libraries to assist specialist programming
  - Mcmc toolpack: R add-on
- QTL programming within general Bayesian software
  - WinBUGS: qtl modelling within Bayesian MCMC software
- QTL modules within general stats software
  - Bmapqtl, bim: R add-ons
- Specialist programs (public and private)
  - qvalue
  - R, C, Fortran, Matlab etc code for papers and problems

Mapmaker

- One example of multitude of QTL software packages
- Whitehead Institute public domain software
- Focus: genetic linkage maps
- Analysis is based on log likelihood. (No Bayes?)
- Multiple testing overcome by setting ‘strict’ LOD score and then relaxing this to assess impact.
- ‘Ripple’ algorithm swaps neighbours to avoid ‘stickiness’

QTL Cartographer

Wang S., C. J. Basten, and Z. -B. Zeng (2001-2003) Windows QTL Cartographer 2.0. Department of Statistics, North Carolina State University, Raleigh, NC. (http://statgen.ncsu.edu/qtlcart/WQTLCart.htm)

- WinQtlCart : mapping quantitative trait loci (QTL) in cross populations from inbred lines. The software is based on QTL Cartographer with interface designed and developed under Microsoft Visual C++ 6.0 environment.
- The current implemented statistical methods include one marker analysis, interval mapping, composite interval mapping, multiple interval mapping and multiple trait analysis.
- Menus include File, Edit, View, Data, Tools, Help
- Does Bayesian interval mapping

MCMCpack

R package that contains functions for posterior simulation for a number of statistical models. All simulation is done in compiled C++. All models return coda mcmc objects that can then be summarized using coda functions or the coda menu interface. The package also contains some useful utility functions, including some additional PDFs and pseudo-random number generators for statistical distributions.

MCMCpack

- Currently MCMCpack allows the user to simulate from the posterior density of the following models: linear regression (with Gaussian errors), a general linear panel model, Wakefield's ecological inference model, Quinn's dynamic ecological inference model, Wakefield's hierarchial ecological inference model, a probit model, a logistic regression model, a one-dimensional item response theory model, a K-dimensional item response theory model, a Normal theory factor analysis model, an ordinal item response theory model, a Poisson regression, and an ordered probit model.
- The package also contains densities and random number generators for commonly used distributions that are not part of the standard R distribution, some additional functions that are useful for manipulating `mcmc` objects, and some data visualization tools for ecological inference.

Some functions in MCMCpack

```
MCMCfactanal            Markov chain Monte Carlo for Normal Theory Factor Analysis Model
MCMClogit               Markov chain Monte Carlo for Logistic Regression
MCMCoprobit             Markov chain Monte Carlo for Ordered Probit Regression
MCMCordfactanal         Markov chain Monte Carlo for Ordinal Data Factor Analysis Model
MCMCpoisson             Markov chain Monte Carlo for Poisson Regression
MCMCprobit              Markov chain Monte Carlo for Probit Regression
MCMCregress             Markov chain Monte Carlo for Gaussian Linear Regression
ddirichlet              Evaluate Density of Dirichlet Distribution
dinvgamma               Evaluate the Density of the Inverse Gamma Distribution
diwish                  Evaluate the Density of the Inverse Wishart Distribution
dnoncenhypergeom        Evaluate Density of Noncentral Hypergeometric Distribution
dwish                   Evaluate the Density of the Wishart Distribution
rdirichlet              Generate Random Draws from the Dirichlet Distribution
rinvgamma               Generate Random Draw from Inverse Gamma Distribution
riwish                  Generate Random Draw from Inverse Wishart Distribution
rnoncenhypergeom        Generate Random Draw from Noncentral Hypergeometric Distribution
rwish                   Generate Random Draw from Wishart Distribution
```

Analysis in WinBUGS

http://www.maths.bris.ac.uk/~masas/work/qtl.htm

□ • one-qtl.bug - this code searches for one QTL on a single chromosome, under two restrictions. 1) there are markers exactly at both ends of the linkage group, and 2) the markers are **equally** spaced.

□ • n-qtl.bug - this code **potentially** searches for n QTL's on a single chromosome. there are no restrictions on inter-marker distances, but there must still be a marker at each end of the chromosome. This has been tested for 1 and 2 QTL's, and became very slow when finding 2. Also the amount of computation required in the CODA analysis stage was immense.

http://www.maths.bris.ac.uk/~masas/work/qtl.htm

Data

- data1.dat - has the following characteristics :

```
# Data set for 200 markers with 1 QTL, with additive effect
# a=3.000000, dominance effect dom=0.000000, mean mu=10.000000
# and variance=1.000000
# M1 - (0.20) - M2 - (0.20) - M3 - (0.05) - M4 - (0.15) - M5 - (0.20) - M6
# where the qtl is marker 4
# *true* QTL indicators are : (0=QQ, 1=qQ or Qq, 2=qq)
# 1 0 1 2 2 1 0 1 2 2 0 0 2 2 0 2 2 0 1 0
# 0 1 2 1 1 2 2 1 1 2 0 0 1 1 0 1 1 2 1 0
# 2 1 0 1 0 1 1 2 0 1 1 1 0 1 2 1 1 1 1 1
# 2 1 2 0 0 0 1 0 1 0 0 0 1 0 1 1 1 2 1 1
# 0 0 0 2 2 1 0 2 2 1 1 2 0 1 1 1 2 2 0 1
# 2 1 0 2 1 1 0 2 1 0 0 1 2 2 2 0 2 0 1 1
# 1 0 0 1 2 1 2 2 0 0 2 1 2 2 2 2 1 0 0 0
# 1 1 2 2 2 0 2 1 2 0 1 1 1 1 1 1 0 1 2 1
# 1 1 1 1 2 1 1 2 2 2 0 2 0 1 1 1 0 1 2 0
# 0 1 1 2 1 2 1 0 2 1 1 0 0 1 2 1 1 0 0 1
# seed generator is  891962883
```

```
model OneQtl;

const
  N=200, # number of progeny
  L=5, # number of markers
  d=0.2; # distance in cMorgans between markers

var
  m[N,5], # marker data
  y[N], # phenotype data
  q[N], # individuals qtl genotype -> 0=QQ, 1=Qq or qQ, 2=qq
  beta[3], # specific mean for each individual genotype
  tau, # 1/sigmasq
  u[L-1], # prior qtl interval probs
  mu, # overall mean level
  a, # additive effect
  d, # dominance effect
  sigmasq, # overall trait variance
  table[9,3], # table of genotype probs given flanking markers
  r0, # \
  r1, # |-> used to simplify table calculations
  r2, # /
  index[N], # pointer to the correct recombination probs
  lambda, # interval containing qtl
  lambda2, # lambda+1
  r[L-1], # recombination probs (per interval)
  theta; # qtl recombination fraction
```

```
data in "data1.dat";

{
mu ~ dnorm(0, 0.0000001);
a ~ dnorm(0, 0.0000001);
d ~ dnorm(0, 0.0000001);
beta[1] <- mu + a;
beta[2] <- mu + d;
beta[3] <- mu - a;
lambda ~ dcat(u[]);
lambda2 <- lambda+1;
theta ~ dunif(0, 0.5*(1-log(-2*d)));
r0 <- r[lambda];
r1 <- theta;
r2 <- (r[lambda]-theta)/(1-2*theta);


table[1,1] <- (1-r1)*(1-r1)*(1-r2)*(1-r2)/((1-r0)*(1-r0));
table[1,2] <- 2*r1*(1-r1)*r2*(1-r2)/((1-r0)*(1-r0));
table[1,3] <- r1*r1*r2*r2/((1-r0)*(1-r0));
table[2,1] <- (1-r1)*(1-r1)*r2*(1-r2)/(r0*(1-r0));
table[2,2] <- r1*(1-r1)*(r2*r2+(1-r2)*(1-r2))/(r0*(1-r0));
table[2,3] <- r1*r1*r2*(1-r2)/(r0*(1-r0));
table[3,1] <- (1-r1)*(1-r1)*r2*r2/(r0*r0);
table[3,2] <- 2*r1*(1-r1)*r2*(1-r2)/(r0*r0);
table[3,3] <- r1*r1*(1-r2)*(1-r2)/(r0*r0);
```

etc

```
table[9,3] <- (1-r1)*(1-r1)*(1-r2)*(1-r2)/((1-r0)*(1-r0));
tau ~ dexp(10);
sigmasq <- 1/tau;
for (i in 1:N) {
  index[i] <- 3*m[i,lambda] + m[i,lambda2]+1;
  q[i] ~ dcat(table[index[i], ]); y[i] ~ dnorm(beta[q[i]], tau); } }
```

data.inits

```
list(lambda=c(1,7), a=c(1,1), d=c(1,1))
```

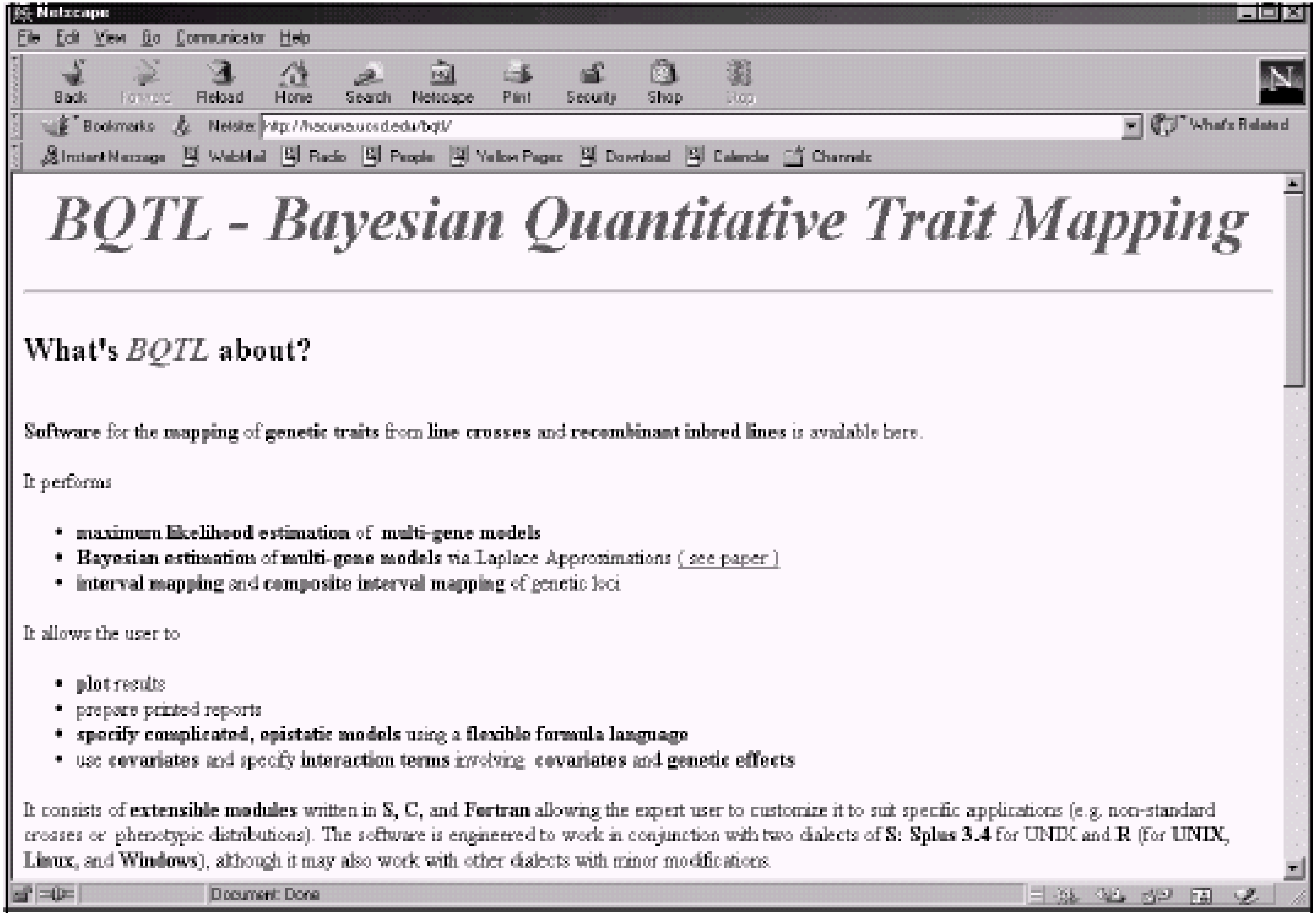
```
data1.dat

list(

  m=structure(

.Data=c(1,1,1,1,1,1,0,0,1,1,0,0,1,1,2,2,2,2,2,2,0,0,0,2,1,1,1,1,0,0,

1,0,0,0,0,2,1,1,1,1,0,0,2,2,1,1,2,2,1,0,0,0,0,1,2,0,1,0,0,0,

0,2,2,1,1,1,2,2,0,1,1,0,0,0,0,1,1,2,2,1,1,2,2,2,2,1,1,0,0,0,

1,1,1,0,0,0,0,0,1,1,1,1,0,1,1,2,2,1,1,2,2,2,2,2,2,0,1,1,2,2,
```

etc

```
0,0,0,0,0,0,0,1,1,2,2,2,2,2,1,0,1,1,2,1,2,2,1,1,1,0,0,0,0,0,
             0,0,0,0,1,0,1,1,2,1), .Dim=c(200,5)),


  y=c( 9.070279,12.729174, 9.453709, 6.488317, 7.856653,
8.505410,10.603332,
      10.655591, 6.668347, 6.773763,12.859454,14.280485, 7.944763,
6.637695,
```

etc

```
  r=c(0.16484, 0.16484, 0.16484, 0.16484),


  u=c(0.25, 0.25, 0.25, 0.25)


)
```


Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Netsite: http://hacuna.ucsd.edu/bqtl/ What's Related
Instant Message WebMail Radio People Yellow Pages Download Calendar Channels
Document: Done

BQTL - Bayesian Quantitative Trait Mapping

What's *BQTL* about?

Software for the **mapping** of **genetic traits** from **line crosses** and **recombinant inbred lines** is available here.

It performs

- **maximum likelihood estimation** of **multi-gene models**
- **Bayesian estimation** of **multi-gene models** via Laplace Approximations (see paper)
- **interval mapping** and **composite interval mapping** of genetic loci

It allows the user to

- **plot** results
- prepare printed reports
- **specify complicated, epistatic models** using a **flexible formula language**
- use **covariates** and specify **interaction terms** involving **covariates** and **genetic effects**

It consists of **extensible modules** written in **S, C,** and **Fortran** allowing the expert user to customize it to suit specific applications (e.g. non-standard crosses or phenotypic distributions). The software is engineered to work in conjunction with two dialects of **S: Splus 3.4** for UNIX and R (for **UNIX, Linux,** and **Windows**), although it may also work with other dialects with minor modifications.

http://hacuna.ucsd.edu/bqtl

Main objects and functions

- The ‘**`analysis.object`**’ bundles most of the needed data and ‘meta–data’
- **`bqtl()`** – maximum likelihood/posterior estimation
- **`linear.bayes()`** – fast MCMC sampling via approximate posterior
- **`loglik()`**

Some bqtl functions

Bayesian QTL mapping toolkit

bqtl	Bayesian QTL Model Fitting
lapadj	Approximate marginal posterior for chosen model
linear.bayes	Bayesian QTL mapping via Linearized Likelihood

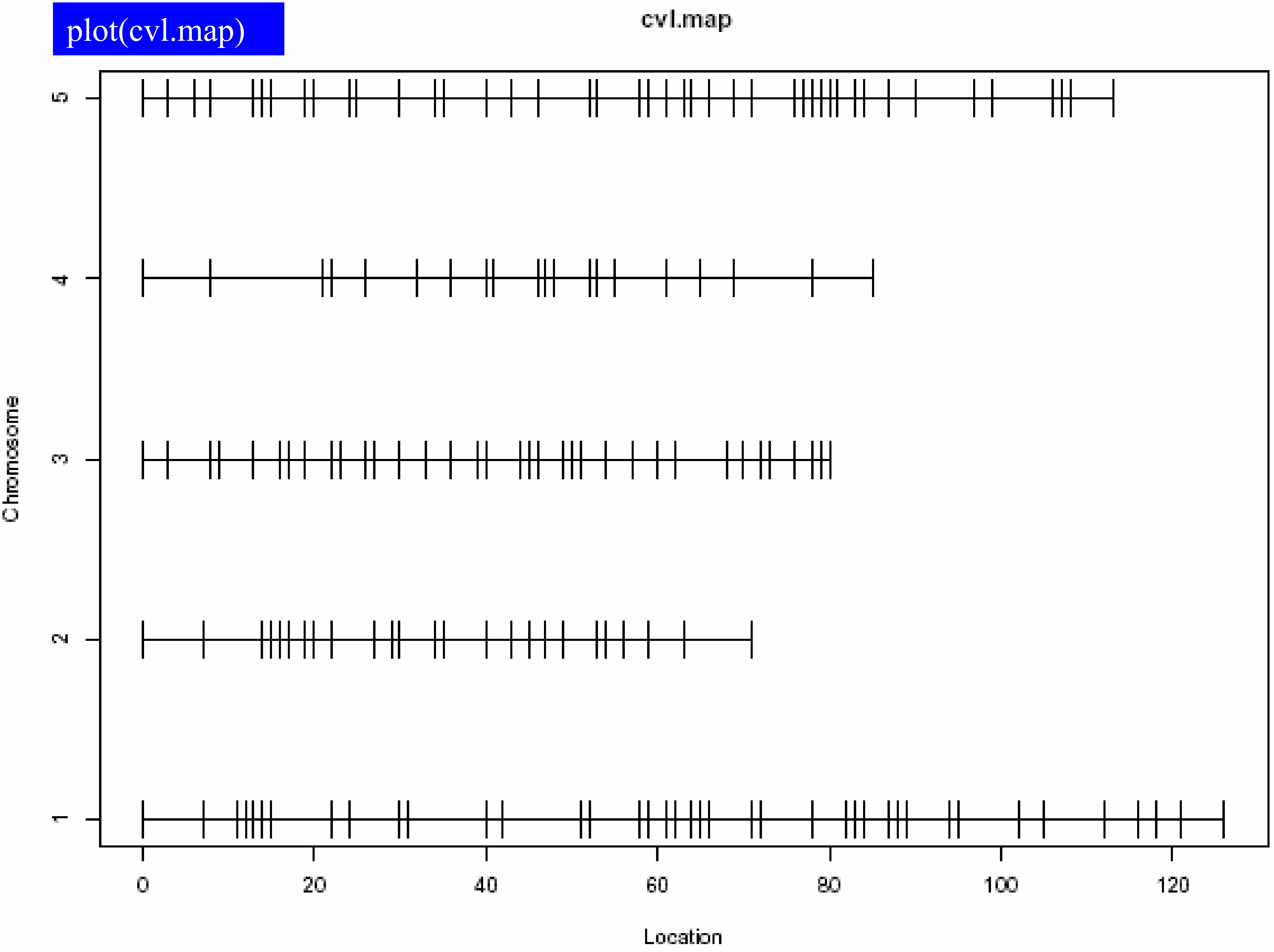
(simulated datasets, marker data, phenotype data)

loglik	Extract loglikelihood, log posterior, or posterior from fitted models
make.analysis.obj	Set up data for QTL mapping
make.location.prior	Provide a default prior
make.regressor.matrix	Create regressors using expected marker values
plot.analysis.object	plots by chromosome location
predict.bqtl	fitted values from QTL models
summary.adj	Summarize Laplace approximations
summary.swap	Summarize Gibbs samples for a k-gene model
swap	MCMC sampling of multigene models
swapbc1	Sample BC1 or Recombinant Inbred loci via approximate posterior.
swapf2	Sample F2 loci via approximate posterior
twoh	One and Two Gene Models Using Linearized Posterior

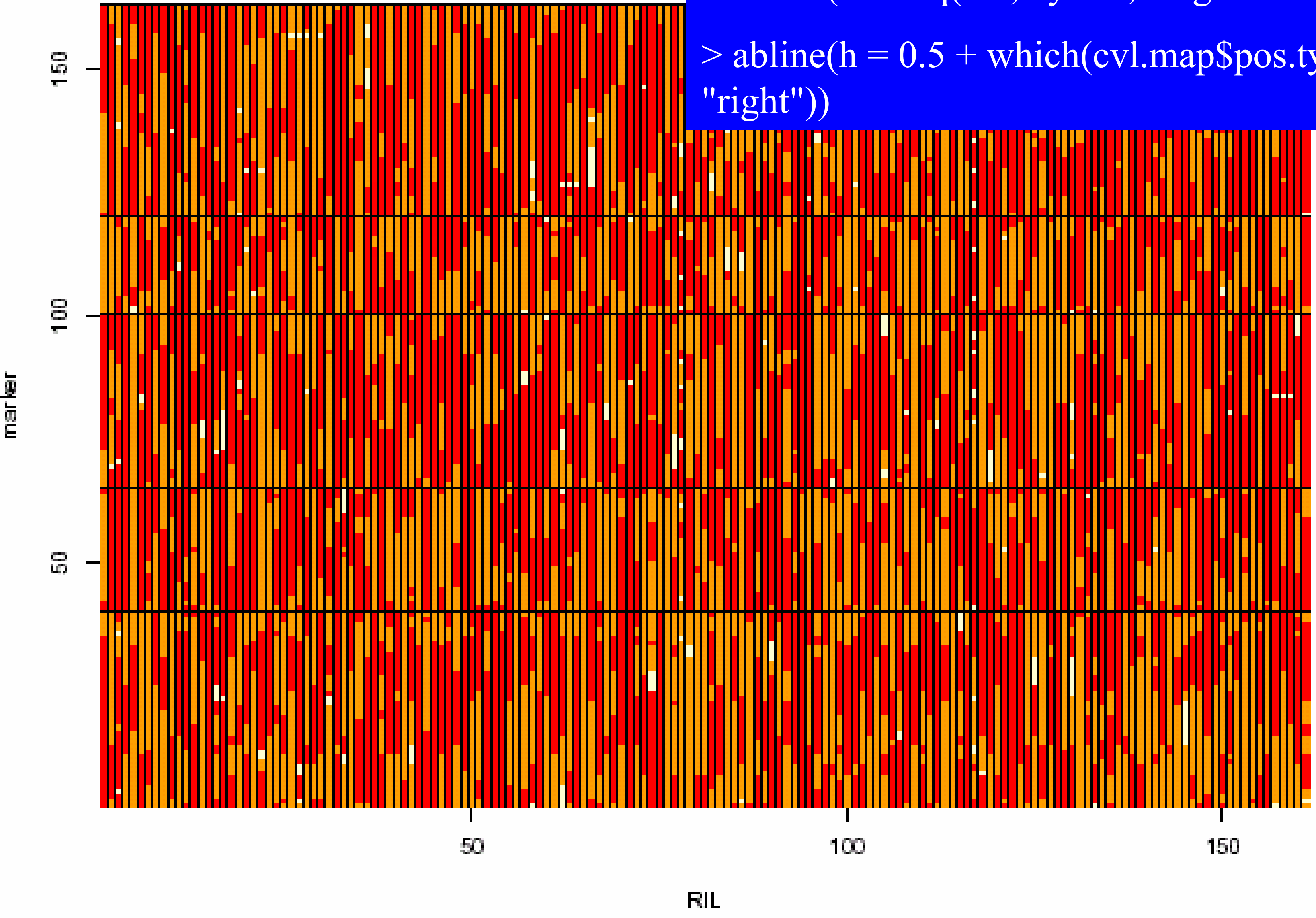
Example bqtl program

```
R : Copyright 2001, The R Development Core Team Version 1.2.2  (2001-02-26)
> postscript()
> source("demo.R",echo=TRUE)
> library(bqtl)
> cvl.map <- make.map.frame(read.table("../plant/cvl-data/cvl.map", header =TRUE))
> plot(cvl.map)
> cvl.markers <- read.csv("../plant/cvl-data/cvl.markers")
> cvl.codes<-apply(cvl.markers,2,function(x) ifelse(x=="AA",1,ifelse(x=="aa",2, 3)))
> image(1:161, 1:163, cvl.codes, xlab = "RIL", ylab = "marker")
> abline(v = seq(1.5, by = 1, length = 160))
> abline(h = 0.5 + which(cvl.map$pos.type == "right"))
➢increasing.cM <- function(x, extra = 1) {add.to.x <- cumsum(extra +
   c(0, x$cM)[which(x$pos.type == "left")]) add.to.x[x$chr.num] + x .... [TRUNCATED]
> x.cM <- increasing.cM(cvl.map)
> x.cM <- c(x.cM, x.cM[163] + 1)
> image(1:161, x.cM, cvl.codes, xlab = "RIL", ylab = "cM")
> abline(v = seq(1.5, by = 1, length = 160))
> abline(h = x.cM[which(cvl.map$pos.type == "right")[1:4] + 1])
➢pheno.dat <- read.csv("../plant/cvl-data/pheno.dat")
```

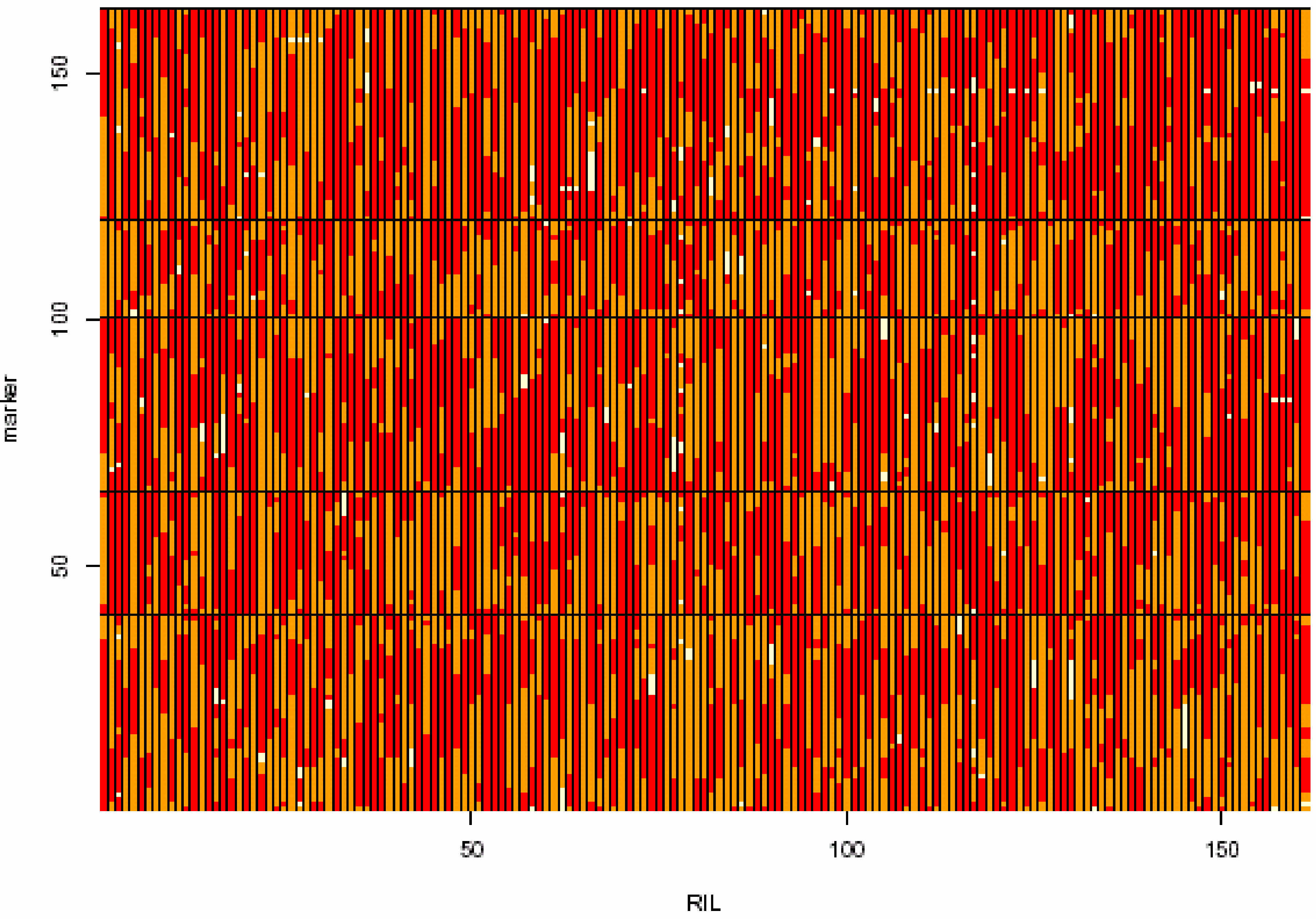
plot(cvl.map)



```
image(1:161, 1:163, cvl.codes, xlab = "RIL",
ylab = "marker")

> abline(v = seq(1.5, by = 1, length = 160))

> abline(h = 0.5 + which(cvl.map$pos.type ==
"right"))
```



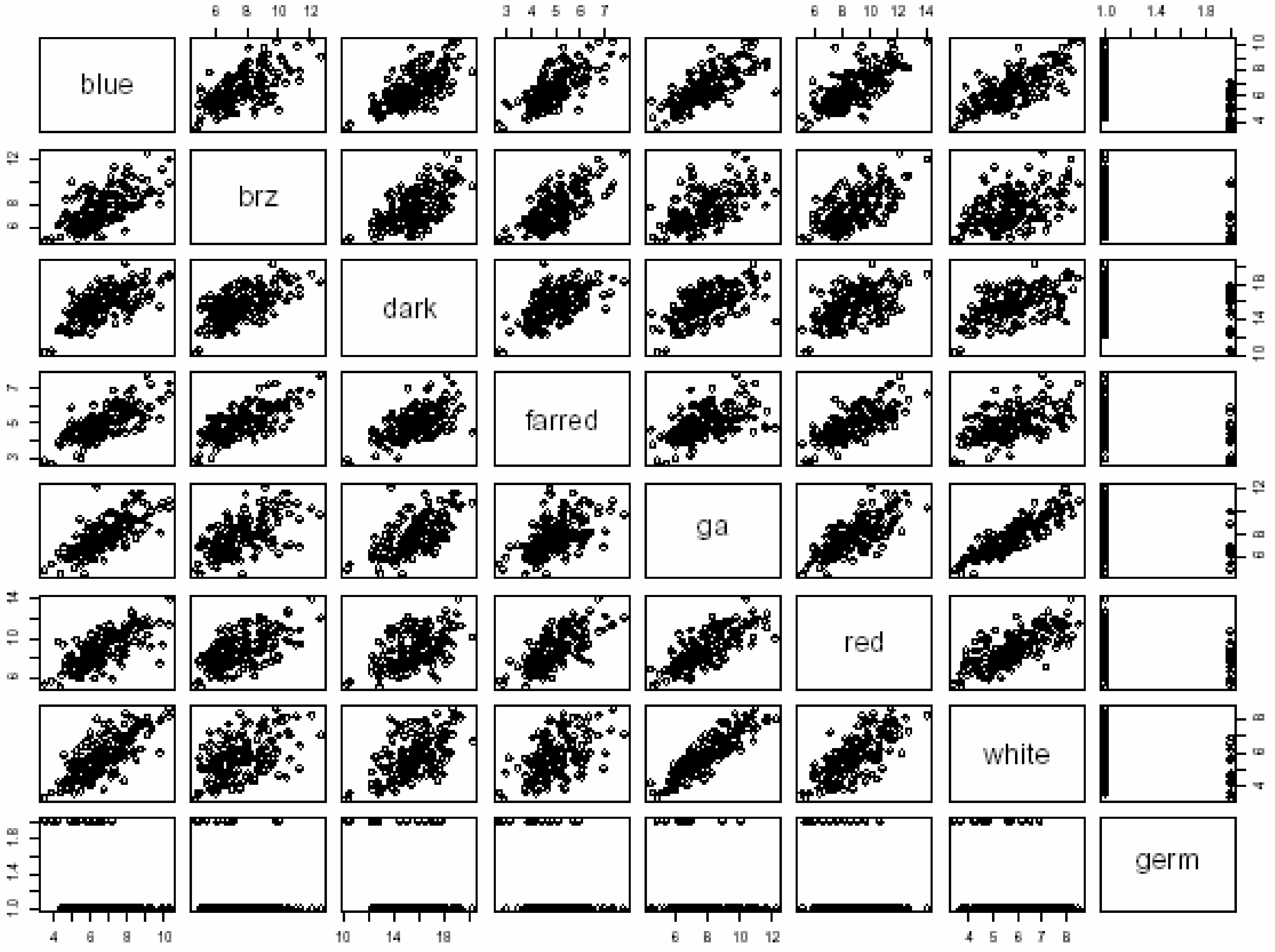
Increased cM



Example (cont)

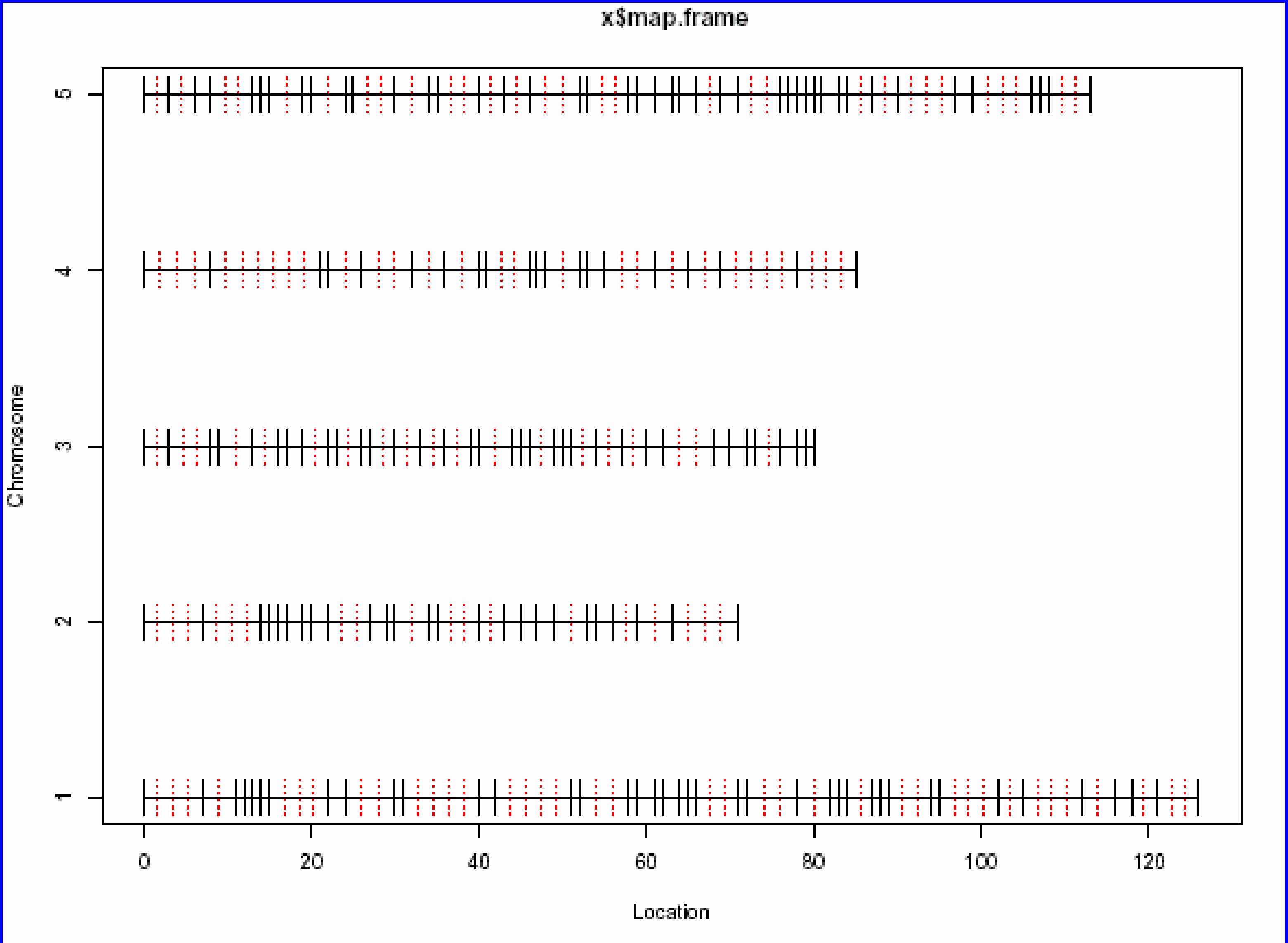
```
> summary(pheno.dat)
          blue           brz            dark          farred
 Min.   : 3.49   Min.   : 4.800   Min.   :10.38   Min.   :2.700
 1st Qu.: 5.66   1st Qu.: 6.370   1st Qu.:14.52   1st Qu.:4.370
 Median : 6.46   Median : 7.520   Median :15.73   Median :4.870
 Mean   : 6.66   Mean   : 7.662   Mean   :15.68   Mean   :4.975
 3rd Qu.: 7.58   3rd Qu.: 8.820   3rd Qu.:17.01   3rd Qu.:5.430
 Max.   :10.27   Max.   :12.640   Max.   :20.29   Max.   :7.760
       ga             red            white            germ
 Min.   : 4.510   Min.   : 5.150   Min.   :3.370   good    :148
 1st Qu.: 6.730   1st Qu.: 7.590   1st Qu.:4.870   not.good: 13
 Median : 7.700   Median : 8.750   Median :5.650
 Mean   : 7.882   Mean   : 8.886   Mean   :5.783
 3rd Qu.: 8.860   3rd Qu.:10.100   3rd Qu.:6.650
 Max.   :12.200   Max.   :14.080   Max.   :8.510
```

```
> pairs(pheno.dat)


> round(cor(data.matrix(pheno.dat)), 2)
        blue   brz  dark farred    ga   red white  germ
blue    1.00  0.64  0.68   0.73  0.76  0.70  0.74 -0.28
brz     0.64  1.00  0.62   0.72  0.58  0.60  0.49 -0.17
dark    0.68  0.62  1.00   0.61  0.62  0.51  0.56 -0.17
farred  0.73  0.72  0.61   1.00  0.54  0.70  0.49 -0.25
ga      0.76  0.58  0.62   0.54  1.00  0.76  0.89 -0.17
red     0.70  0.60  0.51   0.70  0.76  1.00  0.73 -0.19
white   0.74  0.49  0.56   0.49  0.89  0.73  1.00 -0.18
germ   -0.28 -0.17 -0.17  -0.25 -0.17 -0.19 -0.18  1.00
```

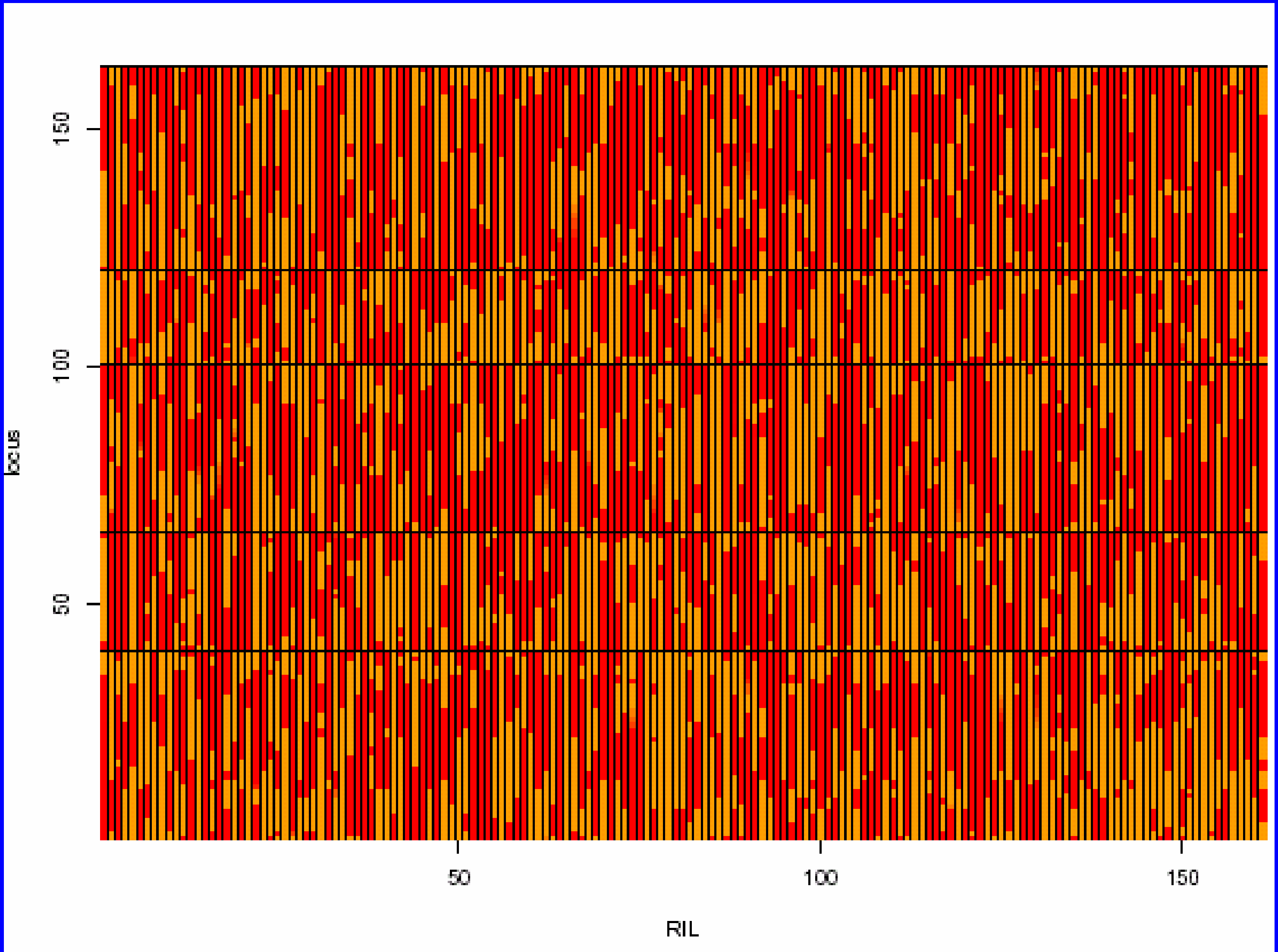


blue
brz
dark
farred
ga
red
white
germ

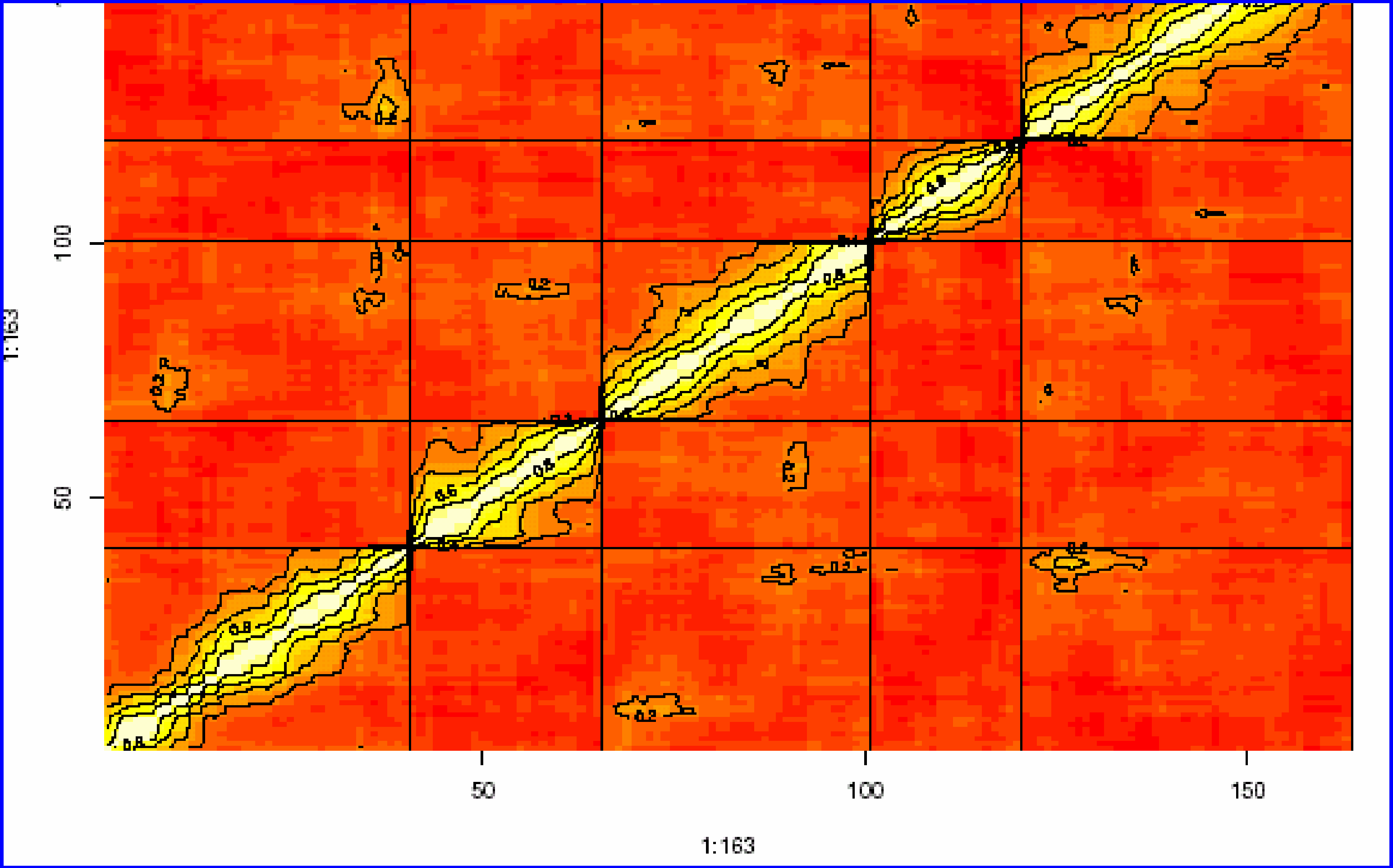
```
> cvl.ana <- make.analysis.obj(pheno.dat, make.map.frame(cvl.map,
    reso = 2), cvl.markers, method = "RI.self")
> plot(cvl.ana)
```



```
> image(1:161, 1:163, cvl.ana$state.matrix[, cvl.ana$map.frame$is.marker,
  2], xlab = "RIL", ylab = "locus", zlim = c(0, 2))
> abline(v = seq(1.5, by = 1, length = 160))
> abline(h = 0.5 + which(cvl.map$pos.type == "right"))
```



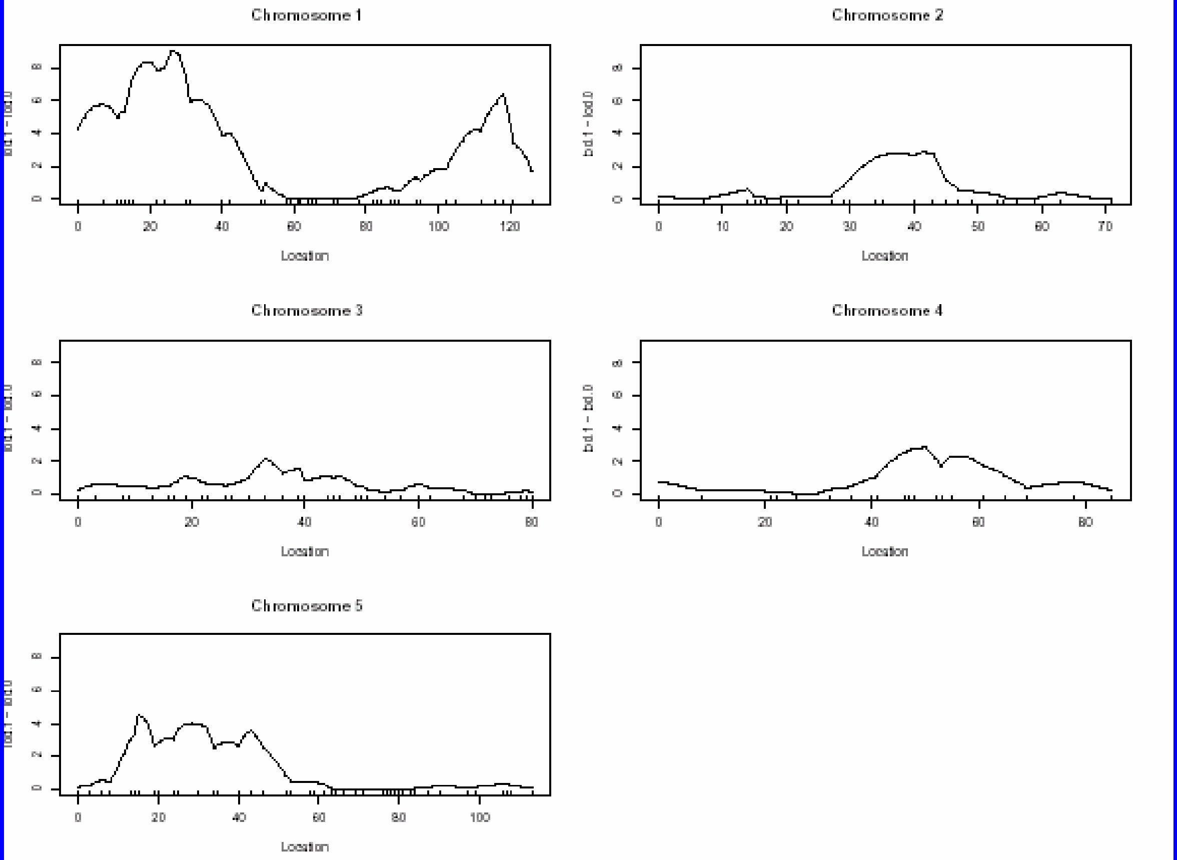
```
> image(1:163, 1:163, cor(cvl.ana$state.matrix[, cvl.ana$map.frame$is.marker, 2]))

> contour(1:163, 1:163, cor(cvl.ana$state.matrix[, cvl.ana$map.frame$is.marker, 2]),
levels = c(0.2, 0.4, 0.6, 0.8), add = T)

> abline(v = 0.5 + which(cvl.map$pos.type == "right"))

> abline(h = 0.5 + which(cvl.map$pos.type == "right"))
```



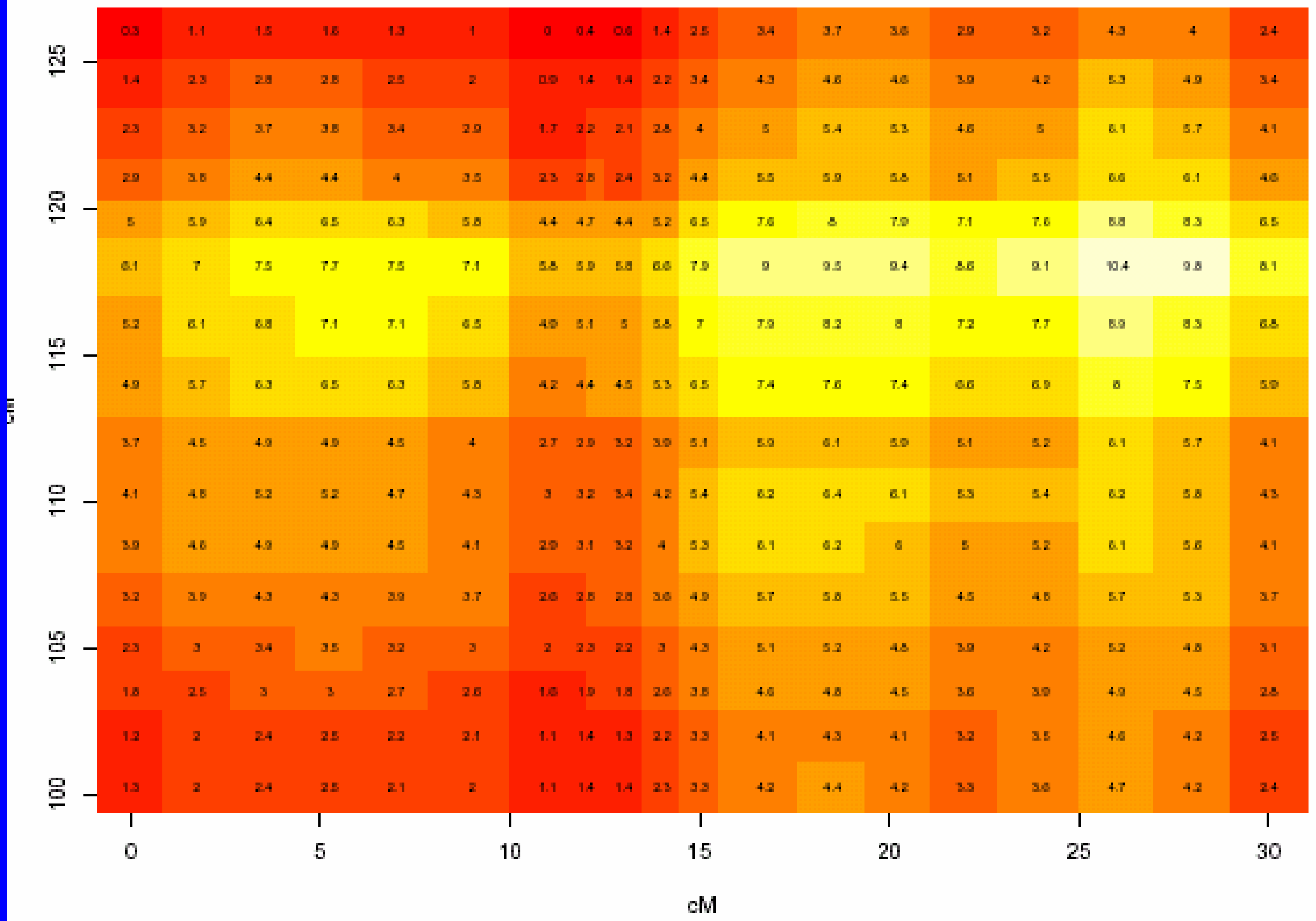
```
> fit.null <- bqtl(white ~ 1, cvl.ana)
> summary(fit.null)
$coefficients
Intercept
 5.782733
$loglik
[1] -256.0794
$std.res
[1] 1.187223
$N
NULL
```

```
> fit.PVV4 <- bqtl(white ~ PVV4, cvl.ana)
> summary(fit.PVV4)
$coefficients
Intercept      PVV4
5.7503866 0.4044548
$loglik
[1] -246.3247
$std.res
[1] 1.116674
$N
     N N.omit N.used
   161      0    161
```

```
> scan.1 <- bqtl(white ~ locus(all), cvl.ana)
> lod.ratio <- log10(exp(1))
> lod.0 <- loglik(fit.null) * lod.ratio
> lod.1 <- loglik(scan.1) * lod.ratio
> par(mfrow = c(3, 2))
> plot(cvl.ana, lod.1 - lod.0)
```

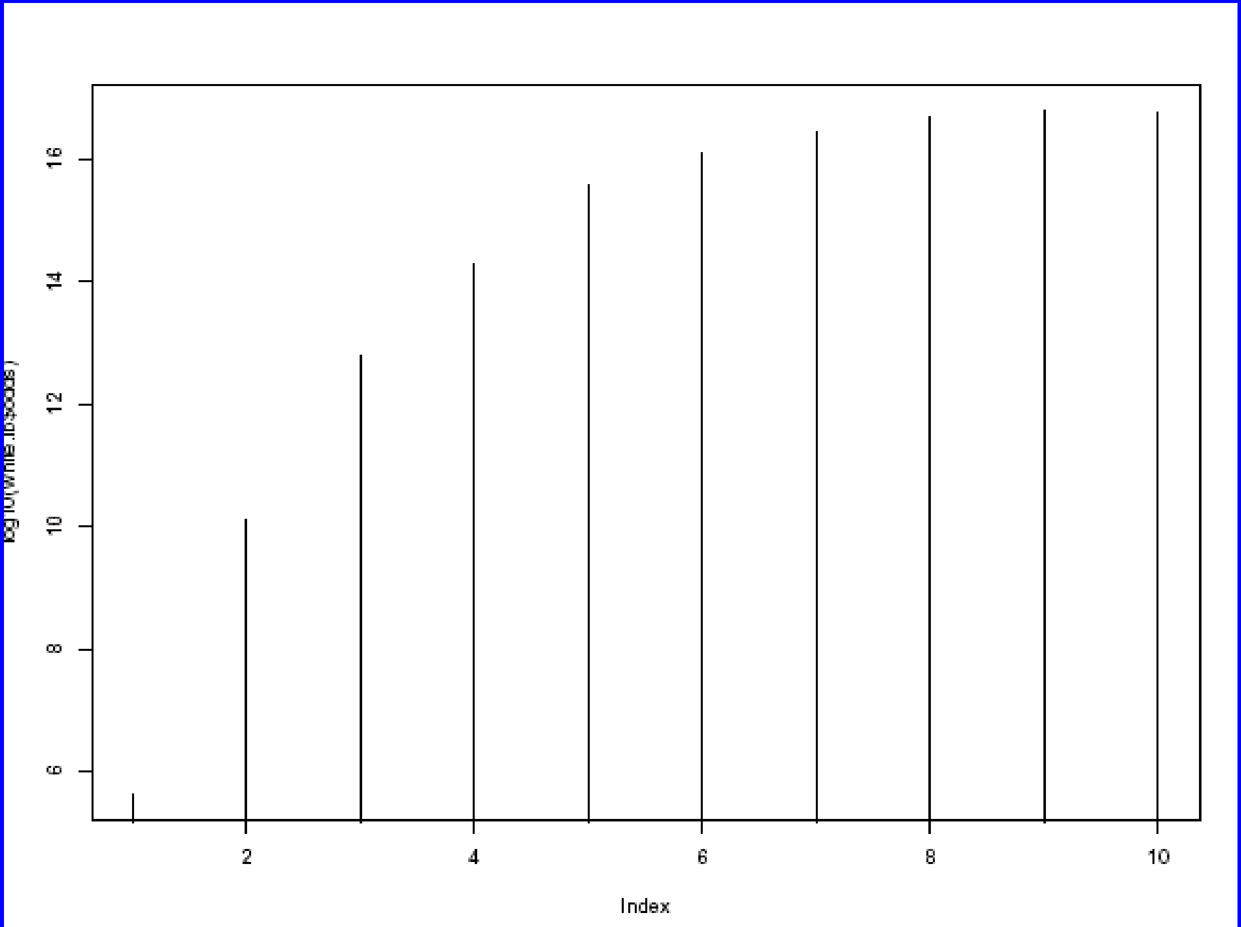


```
> white.perm <- sample(1:161)

> perm.1 <- bqtl(white[white.perm] ~ locus(all), cvl.ana)

> scan.2 <- bqtl(white ~ locus(chromo = 1, cM = c(0, 30)) + locus(chromo = 1, cM = c(100,
150)), cvl.ana)

> scan.2.lod <- loglik(scan.2) * lod.ratio

> index.2 <- map.loc(scan.2)


> vec.2 <- unlist(index.2)

> ind.1 <- as.numeric(vec.2[names(vec.2) == "cM1"])

> ind.2 <- as.numeric(vec.2[names(vec.2) == "cM2"])

> image(unique(ind.1), unique(ind.2), matrix(scan.2.lod,

   nr = length(unique(ind.1))), xlab = "cM", ylab = "cM")
```

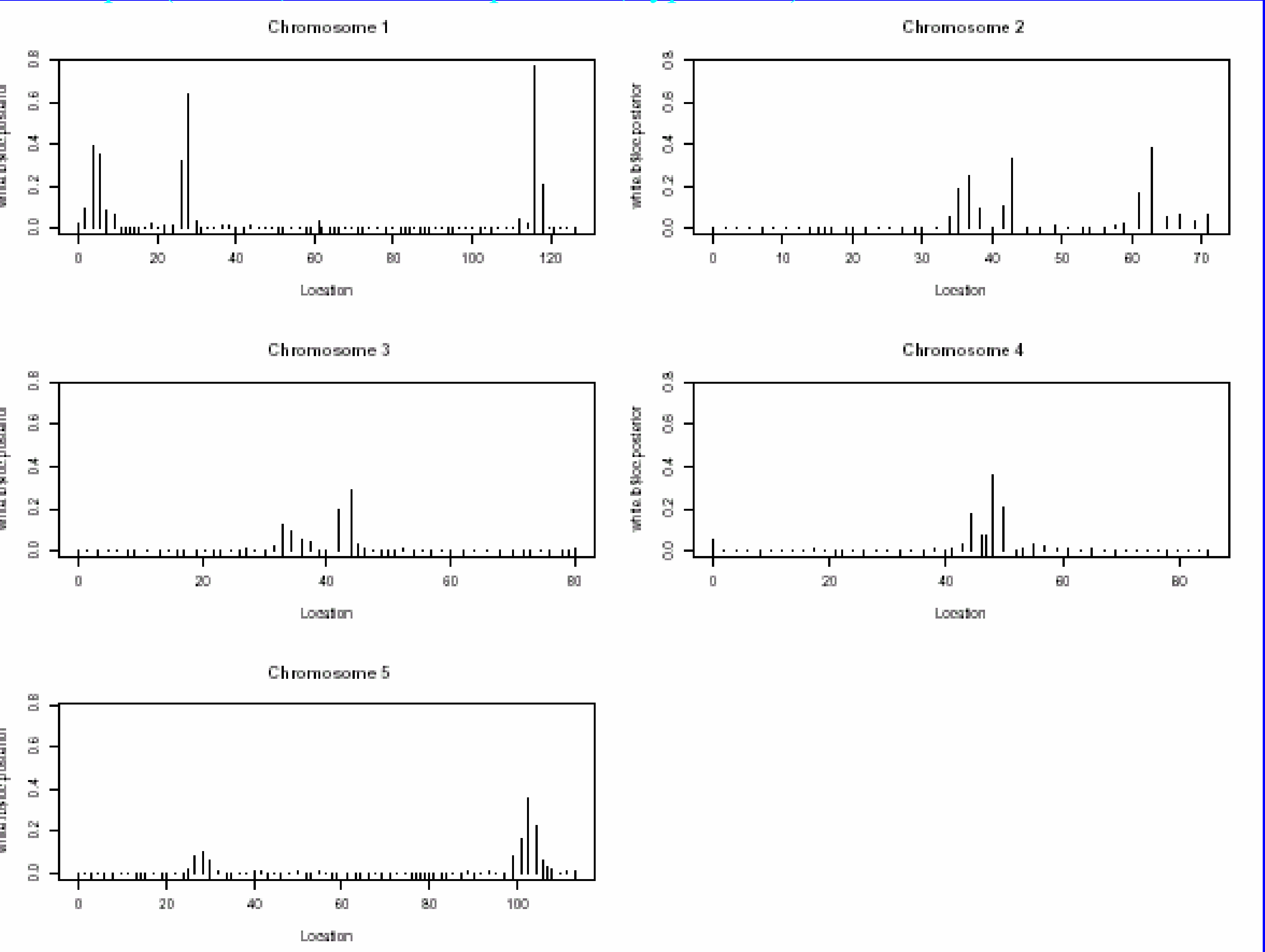


100
105
110
115
120
125
0
5
10
15
20
25
30
cM

```
> text(ind.1, ind.2, round(scan.2.lod - min(scan.2.lod),  1), cex = 0.6)
> lb.spec <- list(gene.number = 1:10, n.cycles = c(0,  400, rep(100, 8)))
> white.lb <- linear.bayes(white ~ locus(all), cvl.ana,   rp = 1, spec = lb.spec)
> par(mfrow = c(1, 1))
> plot(log10(white.lb$odds), type = "h")
```

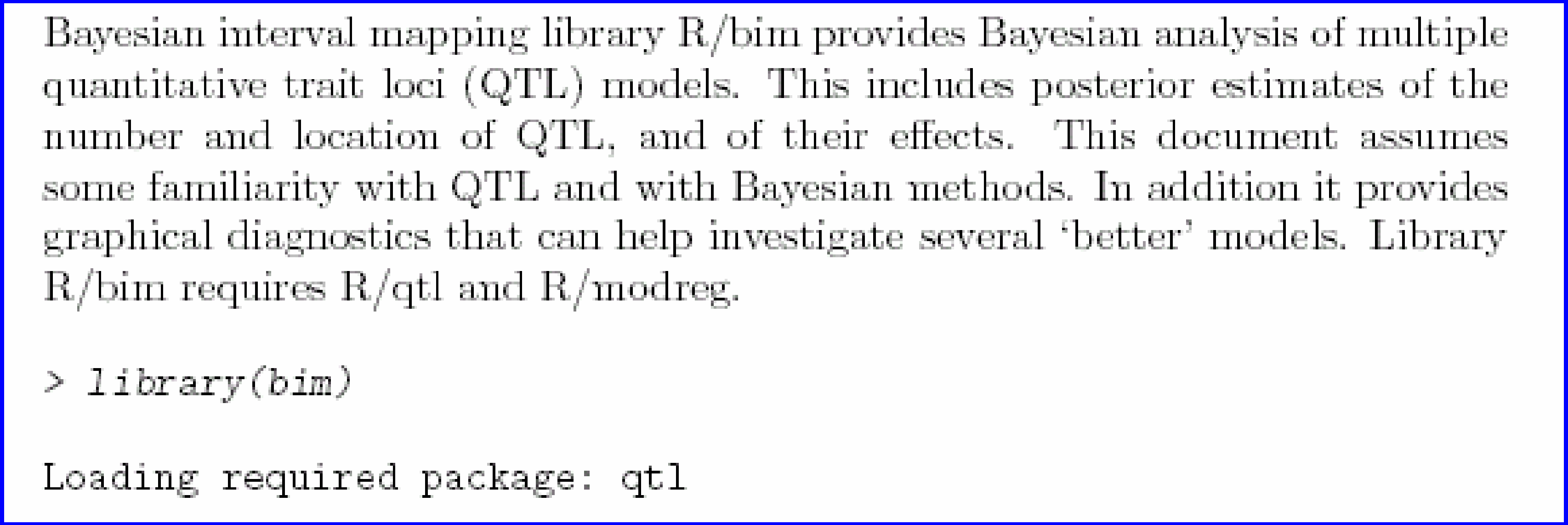


```
> par(mfrow = c(3, 2))
> plot(cvl.ana, white.lb$loc.posterior, type = "h")
```



Bmapqtl: Bayesian interval mapping

R/bim is available directly from the Comprehensive R Archive Network (http://cran.r-project.org) under Contributed Software.



Bayesian interval mapping library R/bim provides Bayesian analysis of multiple quantitative trait loci (QTL) models. This includes posterior estimates of the number and location of QTL, and of their effects. This document assumes some familiarity with QTL and with Bayesian methods. In addition it provides graphical diagnostics that can help investigate several 'better' models. Library R/bim requires R/qtl and R/modreg.

```
> library(bim)

Loading required package: qtl
```

See separate paper

bim/Index

bim.effects	Bayesian QTL map of loci and effects
bim.fdr	Bayesian False Discovery Rate for QTL mapping
bim.model	Bayesian model selection for number and pattern of QTL across genome
bim.	Bayesian QTL estimation and mapping of loci
bmapqtl.options	Options Settings for BmapQTL
Bnapus	Cross structure for complete Brassica napus data
fisch	Eight QTL Stephens and Fisch simulated data
plot.bim.diag diagnostics	Marginal and model-conditional summaries of Bayesian interval mapping
plot.bim.loci	Jittered plot of Bayesian QTL loci samples by chromosome
plot.bim.mcmc	Bayesian MCMC sequence plots for burnin and iterations.
plot.bim.model	Graphical model assessment for Bayesian interval mapping
plot.bim	Diagnostics plots for Bayesian interval mapping
read.bim	Read samples from WinQTL output
read.bmapqtl	Read and write options for WinQTL
run.bmapqtl	Run Bmapqtl reversible jump MCMC
subset.bim	Subsetting Bayesian interval mapping data
summary.bim	Summary of Bayesian interval mapping samples
vern	Eight week vernalization data for Brassica napus