Chapter 16

eQTL Analysis in Mice and Rats

Bruno M. Tesson and Ritsert C. Jansen

Abstract

Since the introduction of genetical genomics in 2001, many studies have been published on various organisms, including mouse and rat. Genetical genomics makes use of the latest microarray profiling technologies and combines vast amounts of genotype and gene expression information, a strategy that has proven very successful in inbred line crosses. The data are analyzed using standard tools for linkage analysis to map the genetic determinants of gene expression variation. Typically, studies have singled out hundreds of genomic loci regulating the expression of nearby and distant genes (called local and distant expression quantitative trait loci, respectively; eQTLs). In this chapter, we provide a step-by-step guide to performing genome-wide linkage analysis in an eQTL mapping experiment by using the R statistical software framework.

Key words: eQTL, genetical genomics, mice, rats, microarray, ANOVA.

1. Introduction

A genetical genomics (1) study involves the perturbation of thousands of genes at the same time through genetic mechanisms of recombination and segregation to create genome-wide “mosaics” of naturally occurring gene variants. Genetical genomics experiments then correlate gene expression variation with DNA variation for tens of thousands of genes, performing tens of thousands times an analysis similar to traditional QTL analysis of a classical phenotypic trait. The analysis of variance (ANOVA) methods offer a framework well suited for such QTL analyses.

Over the past few years, a large number of mouse recombinant inbred populations (RILs, e.g., the BXD or BXA panels) and tissues have been studied in eQTL screens (2–9). The field is now expanding with the study of outbred mice (10). Many of
these data have been uploaded to the GeneNetwork database (11), which have made this the central repository for mouse and rat eQTL data. While eQTL publications on rats have been scarcer, there have been a few studies, for example, using the BXH/HXB panel of recombinant inbred strains (12, 13).

This chapter provides a computational protocol (Fig. 16.1) for eQTL analysis on RIL crosses in mice and rats. The protocol can easily be adapted to suit other genetic populations, such as backcrosses or intercrosses (14).

![Flowchart of eQTL mapping protocol.](image)

2. Materials

2.1. Hardware and Software Requirements

The protocol requires the following:

- R (www.r-project.org): R is a programming environment for statistical computing and graphics. It is available under the GNU General Public License on Windows, Linux/Unix and Mac systems. R has a command line-based interface and is widely used in the field of biostatistics thanks to the availability of multiple add-on packages designed to address specific biological analyses. All the code lines and functions presented in courier font are written in R language. Detailed knowledge of R programming is not required but the interested reader can go to the R tutorial: http://cran.r-project.org/doc/manuals/R-intro.pdf.
2.2. Dataset

The methods we describe here are showcased on Illumina Bead Array data. Illumina is an increasingly popular technology for gene expression profiling and uses arrays containing multiple beads with 50-mer probes attached. Illumina bead arrays have been developed for a number of species including humans, mice, and rats. The protocol described in this chapter is not, however, specific to Illumina data and can be applied to virtually any technology. Some adjustments will need to be made in the particular case of Affymetrix arrays (see Note 2) due to specificities of this technology (i.e., multiple probes per gene).

For this protocol, we use a small sample dataset of 100 expression traits for efficiency purposes. This dataset was extracted from a survey of hematopoietic stem cells in a population of 24 mouse recombinant inbred strains (BXD). This dataset and an electronic version of the code presented in this chapter are available at the following URL: http://gbic.biol.rug.nl/supplementary/2008/linkageGG.

The genotype data should be prepared as a tab-delimited file: each column represents one individual, each row a different marker (see Table 16.1). Values are either 1 for the first parental strain, 2 for the second parental strain, or 1.5 in the case of heterozygote individuals (these should be rare in the case of RILs). The markers are ordered by genomic location as in the marker map file (Section 2.2.5).

Table 16.1
Example of genotype data in tab-delimited format. The columns represent the different recombinant inbred lines (here from the BXD cross). The rows are different markers. RILs are homozygous 1 for the B6 allele or homozygous 2 for the DBA2 allele.

<table>
<thead>
<tr>
<th></th>
<th>BXD6</th>
<th>BXD28</th>
<th>BXD19</th>
<th>BXD15</th>
<th>BXD40</th>
<th>BXD12</th>
<th>BXD31</th>
</tr>
</thead>
<tbody>
<tr>
<td>rs6376963</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>rs6298633</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>D1Mit1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>rs3654866</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>rs3088964</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
Genetical genomics screens can be very expensive; the costs of sample preparation, microarrays, and genotyping must be multiplied by the population size. Selective genotyping is usually not a realistic option to reduce the costs in this context because the samples with the most informative genotypes depend on which gene is being considered. We therefore assume here that the genotype data are complete as is usually the case in genome-wide eQTL studies on recombinant inbred lines. The interested reader may want to refer to Note 3 for software to handle missing information or sparse marker maps.

2.2.2. Expression Data

BeadStudio, the standard Illumina software, produces probe data from bead-level intensities. The output of BeadStudio contains several columns per sample. In this chapter, we use the raw bead summary data as output by BeadStudio. From the BeadStudio output files, we extract the AVG_SIGNAL columns per sample. These columns contain raw-averaged bead intensities for each probe. The expression data are stored in a tab-delimited file, where each column refers to one individual and each row to a probe, as shown in Table 16.2.

<table>
<thead>
<tr>
<th>Probe ID</th>
<th>BXD6</th>
<th>BXD28</th>
<th>BXD19</th>
<th>BXD15</th>
<th>BXD40</th>
</tr>
</thead>
<tbody>
<tr>
<td>GL_84579826-I</td>
<td>341.668</td>
<td>349.7453</td>
<td>509.0667</td>
<td>495.4675</td>
<td>591.0002</td>
</tr>
<tr>
<td>GL_84579830-A</td>
<td>105.3439</td>
<td>113.3545</td>
<td>117.8497</td>
<td>111.6411</td>
<td>109.7728</td>
</tr>
<tr>
<td>GL_84579883-I</td>
<td>121.9119</td>
<td>126.5275</td>
<td>126.1814</td>
<td>132.6144</td>
<td>119.5611</td>
</tr>
<tr>
<td>GL_84579884-A</td>
<td>138.155</td>
<td>138.7963</td>
<td>158.4077</td>
<td>150.2157</td>
<td>133.7268</td>
</tr>
<tr>
<td>GL_84579905-A</td>
<td>189.2942</td>
<td>180.8074</td>
<td>274.7991</td>
<td>367.868</td>
<td>204.4543</td>
</tr>
<tr>
<td>GL_84662726-I</td>
<td>148.5721</td>
<td>147.1926</td>
<td>153.2858</td>
<td>145.0625</td>
<td>135.5658</td>
</tr>
<tr>
<td>GL_84662775-S</td>
<td>132.148</td>
<td>125.3375</td>
<td>139.3298</td>
<td>136.605</td>
<td>130.0435</td>
</tr>
</tbody>
</table>

Table 16.2

Example of raw expression data in tab-delimited format. The first column shows the unique probe IDs, the other columns refer to the samples denoted here by their RIL numbers.

Some authors have suggested alternative pre-processing methods for Illumina data (see Note 4 for references).

2.2.3. Marker Map

We also need a genetic map with the genomic positions of the markers in the genotype file. These positions can be specified in centimorgans (cM) or in mega base pairs (as one or both present). The marker map should be a text file in tab-delimited format (see Table 16.3).
Table 16.3
Example of marker data in tab-delimited format. The first column contains marker IDs, the second column contains chromosome numbers, the third column contains centi-morgan positions, and the last column base pair positions.

<table>
<thead>
<tr>
<th>Marker_Chm</th>
<th>Marker_cM</th>
<th>Marker_Mb</th>
</tr>
</thead>
<tbody>
<tr>
<td>rs6376963</td>
<td>1</td>
<td>0.895</td>
</tr>
<tr>
<td>rs6298633</td>
<td>1</td>
<td>2.367</td>
</tr>
<tr>
<td>D1Mit1</td>
<td>1</td>
<td>3.549</td>
</tr>
<tr>
<td>rs3654866</td>
<td>1</td>
<td>5.797</td>
</tr>
<tr>
<td>rs3088964</td>
<td>1</td>
<td>6.962</td>
</tr>
</tbody>
</table>

2.2.4. Probe and Gene Annotation Data

We finally need a file containing all the relevant probe information, including the genes targeted by the probes and genomic positions of the probes. This file should also be in tab-delimited format (Table 16.4). Annotations provided by microarray manufacturers are often not complete or up-to-date. It is sometimes necessary to re-annotate the probes based on a BLAT search of probe to genome sequence. See Note 5 for some tools which enable such re-annotation.

Table 16.4
Probe annotations. The first column contains the probe IDs, the second column contains chromosome numbers, the third column base pair positions and the last columns give gene information.

<table>
<thead>
<tr>
<th>Probe_Chm</th>
<th>Probe_Mb</th>
<th>Gene_Symbol</th>
<th>Gene_Description</th>
<th>Gene_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>GI_84579826-1</td>
<td>10</td>
<td>87.87682</td>
<td>Gnpab</td>
<td>N-Acetylglucosamine-1-phosphate transfrase, alpha and beta subunits</td>
</tr>
<tr>
<td>GI_84579830-A</td>
<td>12</td>
<td>8.560399</td>
<td>Slc7a15</td>
<td>Solute carrier family 7 (cationic amino acid transporter, y+ system), member 15</td>
</tr>
<tr>
<td>GI_84579883-1</td>
<td>11</td>
<td>120.7646</td>
<td>Slc16a3</td>
<td>Solute carrier family 16 (monocarboxylic acid transporters), member 3</td>
</tr>
</tbody>
</table>
3. Methods

3.1. Experimental Design

When profiling many samples with microarrays, it is often necessary to divide the samples into batches, which may then be profiled at different times or even dates. Attention should be paid to the random assignment of samples to batches in order to minimize the influence of confounding factors. The number of batches should be small (so that it will not take too many degrees of freedom in the analysis) and the batches should preferably be of equal size. Obviously, you should keep track of this batch organization and of any relevant information possibly associated with the profiling process. The eQTL analysis procedure can be adapted to take into account batch effects as described in Note 6.

Special considerations apply to sex, treatment, or environmental factors. If sex is not a factor of interest in the study, it is safer to limit the experiment to males or females only. Alternatively, if individuals of different sexes or different treatments or conditions are used, the model used for the eQTL analysis should account for these as additional factors (see Section 3.6.3). Strategies have been developed to optimize the power of the eQTL study with multiple conditions (e.g., see (15)).

The population size is obviously a critical choice. While more is always better in terms of statistical power, you must find the right balance between the costs incurred by microarray screens and the number of degrees of freedom necessary to fit the models you are using in your study. The relationship between population size and power in classical QTL analysis is discussed and illustrated in (16).

3.2. Loading the Data into R

The following commands will import the data into the R workspace:

```r
rawExpr <- as.matrix(read.csv(file="raw_data.txt", row.names=1, header=TRUE, sep="\t");
genotypes <- as.matrix(read.csv(file="genotypes.txt", row.names=1, header=TRUE, sep="\t");
markerMap <- as.matrix(read.csv(file="markerMap.txt", row.names=1, header=TRUE, sep="\t");
geneMap <- as.matrix(read.csv(file="geneMap.txt", row.names=1, header=TRUE, sep="\t");
```

We convert the data to logarithmic scale with the following:

```r
log2expr <- log2(rawExpr);```

3.3. Useful Checks on the Data

3.3.1. Clustering of the Expression Data

A rapid clustering of the samples can detect major correlation structure such as those caused by batch effects (Fig. 16.2, see Note 6 on how to deal with these artifacts).

```r
sample_clustering <- hclust(dist(t(log2expr)));
plot(sample_clustering);```
Fig. 16.2. Hierarchical clustering of samples using the raw expression data. From this plot, the samples may appear to be divided into three separate clusters. If these correspond to experimental batches, it would be wise to include them in the model (see Note 6).

At this stage, it is advisable to go back to the information collected during the wet lab process (see Section 3.1) to try to match that information with possible clusters.

3.3.2. Genotype Imbalance

It may happen that one of the parental genotypes is very poorly represented at some markers, especially with a small population size. Such imbalance may be caused at random or by segregation distortion, and can lead to an acute sensitivity to outliers; it should therefore be watched carefully. Fig. 16.3 illustrates the genotype distribution across the markers. The code for plotting the genotypes diagnosis graph is as follows:

```r
# The following vector contains all the chromosome lengths
# in Mb for plotting purposes
> chr.lengths <- c(198, 182, 160, 156, 152, 150, 146, 133, 125, 130, 122, 121, 124, 104, 99, 96, 91, 62, 166);

# Check of segregation distortion
> plotGenotypeBalance <- function(genotypes, markerMap, chr.lengths)
  {
    op <- par()
    mk_pos <- as.numeric(markerMap [, "Marker_Mb"])
    diffinv(chr.lengths[match(markerMap [, "MarkerChr"], names(chr.lengths))]);
  }
```
3.4. Normalization of the Expression Data

Microarray data of multiple samples need to be normalized (i.e., converted to the same scale) to allow them to be compared across samples. Normalization removes some of the between-array technical variation. A robust, simple, and efficient method is quantile normalization (17), which is widely used and has been shown to be one of the most appropriate methods in the context of eQTL mapping (18). Quantile normalization orders intensities per sample and then replaces the intensity by the mean of the measurement at that rank in all the samples. An implementation of this

```r
breaks <- c(0,
  apply(chr.locus(mk_pos$1:length(mk_pos)-1) | mk_pos$2:length(mk_pos)|1,mean),
  mk_pos[1:length(mk_pos)])
pos <- rep(mk_pos,nrow(genotypes))
genomic_fac <- factor(as.numeric(genotypes))
par(mfrow=c(1,1), mai=c(0,0,0,0))
spineplot(pos, genomic_fac, breaks=breaks, xaxlabels="",
  yaxlabels="", ylab="Genotype Balance")
chr_col <- c("gre" , "WHITE")[ match(markerMap[, "MarkerChr"], names(chr.lengths))]
par(mfrow=c(1,1))
spineplot(mk_pos, factor(chr_col), breaks=breaks,
  border="NA", xaxlabels="", yaxlabels="", ylab="Marker Positions")
par[c("cp")]

plotGenotypeBalance(genotypes, markerMap, chr.lengths)
```

![Genotype Balance Diagram](image)

**Fig. 16.3.** Genotype balance plot. This plot represents the proportions of individuals with either genotype: white and grey denote the two parental genotypes; heterozygotes appear in black in the middle.

If this “information content” plot reveals a region with such imbalance, QTLs mapped in this region should be carefully scrutinized, since the minor genotype group will be extremely sensitive to outlier samples. The superimposition of the information content on the QTL profiles can provide additional insight into local variations in the statistical power available to detect eQTLs: the regions with the best power being those where the genotypes are perfectly balanced (50% for both parental genotypes).
normalization method is available in the Bioconductor Affy package (19). Our sample dataset has already been normalized, and should therefore not be re-normalized.

The commands to normalize a complete microarray dataset are as follows:

Installing and loading Affy R library:

```r
> source("http://bioconductor.org/biocLite.R");
> biocLite("affy");
> library(affy);
```

These are the commands that apply to quantile normalization:

```r
> normExpr <- normalize.quantiles(log2Expr);
> dimnames(normExpr)<-dimnames(rawExpr);
```

It is advisable to perform similar checks to those described in **Section 3.3.1** on the normalized data to control how the normalization procedure affects the data structure.

```r
#Our sample dataset was extracted from a complete dataset which was already normalized.
> normExpr<-log2Expr;
```

The first step of the actual eQTL analysis is to define the relevant model to use. In the simplest case, namely single-marker mapping without batches and without different environments, the model only includes the genotype effect:

\[ Y_i = m_i + G_i + e_{ij} \]

where \( Y_i \) is the expression measurement for probe \( i \), \( m_i \) is the mean intensity of probe \( i \) over all samples, \( G_j \) is a factor containing the genotypes at marker \( j \), and \( e_{ij} \) is the error term.

The following commands are used to first fit the model using the `lm()` function and then retrieve significances (\( p \)-values) at each marker along the genome. It is assumed that the order of the columns (samples) in the `normExpr` matrix matches the order of the columns in the `genotypes` matrix.

```r
#Single Marker Mapping function
> singleMarkerMapping<-function(traits, genotypes)
{
  qtl_profiles <- NULL;
  for (i in 1:nrow(traits))
    {
      current_profile<-NULL;
      for (j in 1:nrow(genotypes))
        {
          model <- traits[i,j] ~ genotypes[,j];
          anova_table<-anova(lm(model));
          current_profile<-c(current_profile,
                              -log10(anova_table [1,5]));
        }
      qtl_profiles<-rbind(qtl_profiles, current_profile);
    }
  rownames(qtl_profiles) <- rownames(traits);
```
Then applying the single-marker mapping function to the expression values and the genotypes:

```r
qt1Profiles <- singleMarkerMapping(
  traits = normExpr, genotypes = genotypes)
```

Warning: this step can be computationally very intensive (see Note 1).

### 3.5.3. Processing and Visualizing the Results

Using this single-marker mapping approach, we obtain $p$-values for linkage for each gene with each of the markers on our genetic map. The $p$-value distribution across the genome for a given gene is termed the QTL profile of that gene and can be plotted as shown in Fig. 16.4 using the following function:

```r
plotQTLProfile <- function(qtl_profile, markerMap, chr.lengths)
{
  chrStrips <- seq(0, 0, length=sum(chr.lengths))
  for (i in 2:length(chr.lengths)) {
    for (j in 1:length(chr.lengths)[i]:diffinv(chr.lengths)[i+1]) {
      chrStrips[1] <- 1;
      plot(chrStrips, type = "h", col = "#ECECEC", xlab = "", ylab = "", axes = F, ylim = c(0, 1));
      par(new = TRUE);
      marker_x_positions <- as.numeric(markerMap[, "Marker_Hb"] +
                                   diffinv(chr.lengths)[match(markerMap[, "Marker_Hb"], names(chr.lengths))]);
      plot(y = qtl_profile, x = marker_x_positions, xlab = "Marker Position", ylab = "-log(p)",
           ylim = c(0, max(qtl_profile) + 1.6), type = "l", xlab = "HapMap Position", ylab = "-log(p)"
          );
    }
}
```

![Fig. 16.4. Example of a QTL profile plot. This profile shows a QTL peak for the Gnotlab gene on chromosome 10.](image-url)
We use this function to plot the QTL profile of our first probe, which targets the Gnpstab gene.

```r
plotQTLProfile(qtlProfile[,1],markerMap,chr.lengths)
```

Here we set the significance threshold for detection of an eQTL to \( \log 10(p\text{-value}) > 6 \) (see Section 3.7 on how to set significance thresholds). The following function extracts primary QTL peaks from the QTL profiles:

```r
getQTLMaxPeaks <- function(qtl_profiles,threshold)
{
  max_index <- function(v)
  {
    which(v == max(v, na.rm=T)) [1];
  }
  maxQTLs <- cbind(
    rownames(qtl_profiles),
    colnames(qtl_profiles)[
      apply(qtl_profiles,1,max_index)],
    apply(qtl_profiles,1,max,na.rm=T));
  maxQTLsThreshold <-
  matrix(
    maxQTLs [which(as.numeric(maxQTLs[,3])>=threshold),],
    ncol=3);
  colnames(maxQTLsThreshold) <- c("Probe","Marker","p");
  maxQTLsThreshold;
}
QTLPeaksThresh3 <-
getQTLMaxPeaks(qtlProfiles,threshold=3)
```

We now have a list of all the significant primary eQTLs, reported in triplets containing the probe, the marker with the smallest linkage \( p\)-value, and the largest “minus log \( p\)-value” of that linkage. It is sometimes useful to report a confidence interval too. Two approaches are commonly employed: bootstrap and 1-lod-score drop off (20, 21).

We can now generate an eQTL dot plot (Fig.16.5), which provides an informative summary of the mapping results.

```r
qtlDotPlot <-
function(QTLPeaks,markerMap,geneMap,chr.lengths)
{
  chrStrips <- seq(0,0,length=sum(chr.lengths))
  for(i in 2:0:as.integer((length(chr.lengths)-1)/2)+1)
  {
    for (j in diffinv(chr.lengths[i]:diffinv(chr.lengths)[i+1]))
    {
      chrStrips[j] <- 1;
    }
  }
  plot(chrStrips,type="h",col="#CECECE",xlab="",
ylab="",axes=F ylim=c(0,1));
  par(new=TRUE);
  QTL_Positions <-
  as.numeric(
    markerMap[QTLPeaks [,"Marker","Marker_Mb"]]
    + diffinv(chr.lengths)[
      match(
        markerMap[QTLPeaks [,"Marker","Marker_Chr"],
        names(chr.lengths)]
      ));
  Gene_Positions <-
Fig. 16.5. An example of an eQTL dot plot. Each dot represents a significant eQTL, with the gene position on the Y-axis and the QTL position on the X-axis. (This plot was obtained using the complete dataset, not just the 100 probes subset we are using as a sample for this chapter).

Locally acting eQTLs appear on the diagonal and are often over-represented. You can also sometimes the presence of vertical bands (typically between 0 and 8), which have been suggested to reflect the presence of regulation hotspots: a distant eQTL controlling or regulating many genes (22).

3.6. More Elaborate Models

3.6.1. Multiple QTL Mapping

To potentially improve statistical power for eQTL detection, it can be worthwhile fitting multiple QTL models, for example, by a stepwise procedure: correct for the first (most significant) eQTL effect found, and then map the corrected data to detect a second eQTL. The two-eQTL model is as follows:
\[ Y_i = m_i + G_k + G_j + e_{ij} \]

where \( G_k \) is the genotype vector at the first QTL position.

In this code example, we look for secondary eQTLs for the probes, for which a primary QTL has already been identified in Section 3.5. We use the list \text{maxQTLPeaksThresh3}.

```r
> secondaryMarkerMapping <- function(traits, genotypes, primaryQTLs)
  {
    if (paste(rownames(traits), collapse = '') !=
        paste(primaryQTLs [, "Marker"], collapse = ''))
      {
        print("Error: Traits submitted do not match traits with primary eQTLs.");
        return;
      }
    qtl_profiles <- NULL;
    for (i in 1:nrow(traits))
      {
        current_profile <- NULL;
        for (j in 1:nrow(genotypes))
          {
            model <- traits [i,] ~
                genotypes [primaryQTLs [i,"Marker"],] + genotypes [j,];
            anova_table <- anova(lm(model));
            current_profile <- c(current_profile,
                                  -log10(anova_table [2,5]));
          }
    }
    qtl_profiles <- rbind(qtl_profiles, current_profile);
    rownames(qtl_profiles) <- rownames(traits);
    colnames(qtl_profiles) <- rownames(genotypes);
    qtl_profiles
  }
>
> secondaryQTLProfiles <- secondaryMarkerMapping(
    normExpr [QTLPeaksThresh3 [, "Probe"],],
    genotypes, QTLPeaksThresh3);
>
> secondaryQTLPeaksThresh3 <-
  getQTLMaxPeaks(secondaryQTLProfiles, threshold=3);
```

Using the function defined in Section 3.5 to extract QTL peaks, we can now create a list of the significant secondary eQTLs for a given threshold:

```r
> secondaryQTLPeaksThresh3 <-
  getQTLMaxPeaks(secondaryQTLProfiles, threshold=3);
```

It is, of course, possible to include three or more QTLs per gene by extending the model. However, you should be cautious because of over-fitting issues. This sequential way of defining the co-factors to include in the model may not be optimal, and there are a number of more advanced strategies which address the problem of model selection (see Note 7).

In the previous step we have presented how to detect multiple QTLs per gene. However, we have not tested for interaction between the eQTLs (i.e., if the effect of one eQTL is modulated by the effect of the second one). Such complex mechanisms are common in gene regulation and are termed epistasis. Our eQTL analysis model can again be extended to take such epistasis effects into account.
\[ Y_i = m_i + G_k + G_j + G_k^* G_j + e_{ijk} \]

The code below tests for epistasis between two eQTLs that we identified for the gene in Section 3.6.1:

```r
> my_probe <- secondaryQTLPeaksThresh3[1,1];
# Probe of the gene we will test for epistasis
> G1 <- genotypes[QTLPeaksThresh3[,"Probe"] == my_probe, "Marker",];
> G2 <- genotypes[secondaryQTLPeaksThresh3[, "Marker"],];
> model.epistasis <- normExpr[my_probe] ~ G1 + G2 + G1:G2;
>anova_table <- anova(lm(model.epistasis));
>interaction_p_value <- anova_table[3,5];
```

In this example, the \( p \)-value is insignificant and there is no evidence for epistasis.

Interactions also occur between two loci whose main effects (terms G1 and G2 in the model) may not be significant on their own. It can therefore be relevant to screen for interactions for any possible pairs of loci, but this can sometimes be computationally unrealistic (a two-dimensional genome scan leads to a huge multiple-testing problem). For more guidance on strategies for epistasis testing, see Note 7.

3.6.3. Adding Environments/Treatments

Genetical genomics studies can provide insights into the way different environments or treatments affect the regulation of gene expression. When combining the genetic perturbation naturally present in inbred populations with the effect of different environments, the study of the interaction between those two causes of variation can teach us about the plasticity of eQTLs (15, 23). We can illustrate this with the example of the study of gene expression regulation across several cell types. In the example below, expression profiles were collected from four distinct cell types. The following model can be used:

\[ Y_i = m_i + CT + G_j + CT^* G_j + e_i \]

where CT is the cell type factor. For this example, we need to load new data files:

```r
> genotypes4CT <-
- as.matrix(read.csv(file="genotypes4ct.txt",
- sep="\t", row.names=1));
> expr4CT <- as.matrix(read.csv(file="expr4ct.txt",
- sep="\t", row.names=1));
# the cell types are coded as "1", "2", "3" and "4"
> CT.factor <-
- factor(c(rep(1,24), rep(2,25), rep(3,22), rep(4,25)));
```

The mapping function therefore becomes

```r
> singleMarkerMappingWithEnv <-
function (traits, genotypes, env.factor) {
  P1 <- P2 <- P3 <- P4 <- NULL;
  for (i in 1:nrow(traits)) {
    if (traits[i,] == env.factor) {
      P1 <- P2 <- P3 <- P4 <- NULL;
      # Other code...
    }
    # Other code...
  }
  # Other code...
}
```
for (j in 1:nrow(genotypes))
{
    model_environment<- traits [1,] ~ factor(env.factor)+
    genotypes [j,] + factor(env.factor):genotypes [j,];
    anova_table <- anova(lm(model_environment));
    p1 <- c(p1,log10(anova_table [[5]] [2]));
    p2 <- c(p2,log10(anova_table [[5]] [2]));
    p3 <- c(p3,log10(anova_table [[5]] [3]));
}
P1 <- rbind(P1,p1); # Env
P2 <- rbind(P2,p2); # qtl
P3 <- rbind(P3,p3); # qtlxEnv

dimnames(P1) <-
    list(rownames(traits),rownames(genotypes));
dimnames(P2) <-
    list(rownames(traits),rownames(genotypes));
dimnames(P3) <-
    list(rownames(traits),rownames(genotypes));
results<-list();
results$Profiles_Environment <- P1;
results$Profiles_QTL <- P2;
results$Profiles_QTLxEnvironment <- P3;
results;

This function outputs three p-values for each trait-marker pair: the first p-value indicates the significance level of the environment term (a low p-value indicates a clear overall influence of the environment on the trait; this p-value is not valid if the environment has not been randomly allocated to samples). The second p-value is the significance of the main genotype effect at that marker, while the third p-value reflects the significance of the genotype by environment interaction term.

>results4CT <- singleMarkersMappingWithEnv(
    expr4CT,genotypes4CT,env.factor=CT.factor);
>interactionQTLsThreshold3<-getQTLmaxPeaks(
    results4CT$Profiles_QTLxEnvironment,
    threshold=3);

A norm of reaction plot (Fig.16.6) can show how the eQTL effect is modulated by the environment. It can be obtained using the following function:

>plotNormOfReaction<-function(trait,genotype,env.factor)
{
    env.factor <- as.numeric(env.factor);
    plottingColors <- c("black","black","lightgrey","grey");
    yrange <- range(trait) +
        c((range(trait) [1] - range(trait) [2]) / 5,
            (range(trait) [2] - range(trait) [1]) / 5);
    plot(y=trait,x=env.factor,xlim=range(env.factor),
        ylim=yrange,col=plottingColors[2+"genotype"],
        xlab="Environment",
        ylab="expression for individuals");
    meanGroupValues <-
        matrix(nrow=2,ncol=length(unique(env.factor)));
    for (env in 1:length(unique(env.factor)))
    {
        meanGroupValues [1,env] <-
            mean(trait [intersect( which(genotype == 1),
                              which(env.factor == env))]);
        meanGroupValues [2,env] <-
            mean(trait [intersect( which(genotype == 0),
                              which(env.factor == env))]);
    }
}
meanGroupValues <-
mean(trait [ intersect(
which(genotype == 2),
which(env.factor == env)]));
text(env, yrange [1], env, cex=1.5, col="black");
par(new=T);
matplot(y=trait(meanGroupValues), ylim=range(env.factor),
ylim=yrange, xlab="", ylab="", xaxt="n", yaxt="n",
type="I", lty=1, lwd=4, col=c("black", "grey"));
}
>plotNormOfReaction;
expr4CT [ interactionQTLsThresh3 [1,"Probe"],
genotypes4CT [ interactionQTLsThresh3 [1,"Marker"],]
, CT.factor);

3.7. Determining the Significance Threshold

Fig. 16.6. Norm of reaction plot: eQTL by environment interaction, with the cell types on the X-axis and the gene expression on the Y-axis. Each dot is an individual sample measurement (black = B6, grey = DBA2). The lines represent mean values. The effect of the QTL is here reversed in cell type 4 compared with the other three cell types.

In eQTL analysis, the determination of the threshold for statistical significance is a critical aspect since multiple testing issues arise from both the high number of genes studied and the high number of genomic loci at which linkage is tested. The p-values yielded by the ANOVA must be adjusted to take into account these multiple testing issues. Bonferroni correction is somewhat too drastic here since the tests are not independent: firstly, the markers tested are intrinsically linked and thus correlated to their neighbors on each chromosome; and, secondly, large families of genes are known to be co-regulated, so there is also a correlation structure in that dimension.

A more appropriate approach is to estimate a false discovery rate (FDR) based on a permutation strategy (24). A carefully designed permutation procedure will make it possible to estimate
the null distribution. The principle is to apply the exact same analysis protocol to permuted datasets, calculate the average number of rejected null hypotheses for a certain p-value threshold in those permuted datasets, and then derive an FDR estimate, at that p-value threshold, as the average number of rejected hypotheses in the permuted datasets divided by the number of rejected hypotheses at the same threshold in the true data.

Different permutation strategies are possible: we advise permuting only the genotypes of the individuals, while conserving trait values (gene expression measurements). This ensures that the permutation procedure does not break the internal correlation structure of the data (within markers and within genes), but that any linkage detected between a marker and gene expression in a permuted dataset is a false-positive (25).

The following function here estimates the number of false-positives obtained with a permuted dataset for a range of p-value thresholds in the single-marker mapping case discussed in Section 3.5.

```r
>estimateFalsePositives <- function(traits, genotypes, threshold_range, nperm) {
  permuteGenotypes <- function(geno) {
    geno[, sample(1:nrow(geno), ncol(geno), replace=FALSE)];
  }
  Counters <- NULL
  for (i in 1:nperm) {
    permGenotypes <- permuteGenotypes(genotypes);
    current_profiles <- singleMarkerMapping(traits, permGenotypes);
    current_counters <- NULL;
    for (thresh in threshold_range) {
      current_counters <- c(
        current_counters,
        length(
          which(apply(current_profiles, 1, max) > thresh))
      );
    }
    Counters <- rbind(Counters, current_counters);
  }
  colnames(Counters) <- apply(matrix(threshold_range, 1, nrow(Counters)),
                             1, as.character);
  for (j in 1:nrow(Counters)) {
    rownames(Counters)[j] <-
      paste("permutation round", j);
    Counters;
  }
>}

>false_positive_estimates <-
  estimateFalsePositives(normExpr, genotypes, threshold_range = c(3, 4, 5, 6, 7), nperm = 10);
#In this example, for efficiency purpose we run only
#10 permutations round. For a reliable estimation,
#100 permutations would be a minimum.
```
We can derive an estimate of the FDR, for example, for a $p$-value threshold of 3:

```r
# number of rejected null hypothesis:
positives_thresh3 <- nrow(QTLPeaks_thresh3)
# number of false positives
false_positives_thresh3 <-
  mean(false_positive_estimates[,"3"])
# FDR
FDR_thresh3 <- false_positives_thresh3/positives_thresh3;
```

The result here gives a very high FDR (>$50\%$) which means we need to use a more stringent threshold than $\log p > 3$.

This code can easily be adapted to estimate the FDRs for other mapping procedures, the principle being that the permuted data should be analyzed with the same model and the same procedure as the real data. The cases of complex models for stratified data or interacting factors require adapted permutation procedures (26).

Another advantage of this permutation procedure is that it allows an unbiased estimation of the significance of the number and size of eQTL hotspots. There is some speculation that some hotspots may be the result of false-positive linkage of groups of correlated genes to random genome positions (with no regulatory connection) (27, 28). Calculating the size and the number of the hotspots obtained with permuted datasets that have retained the correlation between genes is a straightforward manner of testing the significance of hotspots (25).

Some authors have suggested using different thresholds for local and distant eQTLs: detecting local effects does not involve genome-wide testing of loci and can therefore be controlled with relaxed thresholds (29). Finally, it is important to take into account the fact that sex chromosomes have specific properties which require different thresholds for sex and autosomal chromosomes (see Note 8).

### 3.8. Interpretation of the Results

A typical eQTL analysis will yield hundreds or thousands of genetic linkages. Extracting meaningful biological information from the results can prove challenging.

Local eQTLs typically offer insights into possible cis-regulatory differences between the two alleles. Inspection of the polymorphisms in the regulatory regions of the gene can provide insight into the possible molecular mechanism (e.g., an SNP located in a transcription factor binding site located in the promoter region of the gene). Polymorphisms located within the probe target regions can also create a technically false-positive eQTL (see Note 9).

A distant eQTL indicates the presence of a distant regulator (e.g., a transcription factor or an miRNA gene) at the QTL location. This regulator may either be locally regulated or contain a non-synonymous polymorphism affecting its function. It is,
however, usually difficult to directly pinpoint the regulator because of the relatively poor mapping resolution (a QTL typically spans several Mb and contains tens to hundreds of candidate genes).

Hotspots, which are large groups of genes having co-localizing eQTLs, may reveal the action of master regulators (i.e., genes controlling many others). It is possible to design strategies to reduce the large number of candidate regulators (typically hundreds) that fall into the hotspot QTL region. We present here one small sample hotspot, illustrated in Fig. 16.7. Sixteen genes were found to share a QTL. We investigate the possibility of a common regulator located in that QTL region. Different candidates (genes physically located within the QTL) are ranked according to their correlation with each of the hotspot genes (Table 16.5). Using Rank Product (30) it is then possible to prioritize the candidates.

![Graph showing -log(p) vs Marker Position]

![UCSC Known Genes Based on UniProt, RefSeq, and GenBank mRNA]

**Fig. 16.7.** Multiple possible candidate regulators: on the top panel the QTL profiles of 16 genes show a common peak. A large number of genes, illustrated by a UCSC genome browser screenshot (lower panel), lie within the confidence interval of that eQTL.
Table 16.5
Prioritization of candidate regulators based on Rank Product of correlation with hotspot genes. The genes with the lowest p-values are those that correlate best with the hotspot genes and are therefore given top priority. *Mxi1* and *Add3* are the most likely candidates according to this correlation criterion.

<table>
<thead>
<tr>
<th>Candidate regulators</th>
<th>Rank Product of correlation with hotspot genes</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Mxi1</em></td>
<td>2.412555</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td><em>Add3</em></td>
<td>2.641117</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td><em>Smncl1</em></td>
<td>3.808562</td>
<td>0.0012</td>
</tr>
<tr>
<td><em>Shoc2</em></td>
<td>3.883223</td>
<td>0.00135</td>
</tr>
<tr>
<td><em>Gpam</em></td>
<td>4.179971</td>
<td>0.0027</td>
</tr>
<tr>
<td><em>Sores1</em></td>
<td>4.571672</td>
<td>0.00825</td>
</tr>
<tr>
<td><em>5830416P10Rik</em></td>
<td>5.282127</td>
<td>0.03055</td>
</tr>
<tr>
<td><em>Adra2a</em></td>
<td>7.849839</td>
<td>0.38</td>
</tr>
<tr>
<td><em>1700001K23Rik</em></td>
<td>9.409524</td>
<td>0.69665</td>
</tr>
<tr>
<td><em>Fdeg4</em></td>
<td>10.60708</td>
<td>0.8643</td>
</tr>
<tr>
<td><em>Guty2g</em></td>
<td>10.74565</td>
<td>0.878475</td>
</tr>
<tr>
<td><em>Dusp5</em></td>
<td>11.54332</td>
<td>0.93845</td>
</tr>
<tr>
<td><em>Tcf7l2</em></td>
<td>11.95649</td>
<td>0.9578</td>
</tr>
<tr>
<td><em>Tectb</em></td>
<td>13.5352</td>
<td>0.99195</td>
</tr>
<tr>
<td><em>Zdhhc6</em></td>
<td>14.56703</td>
<td>0.99825</td>
</tr>
<tr>
<td><em>Ins1</em></td>
<td>15.25569</td>
<td>0.99965</td>
</tr>
<tr>
<td><em>Vit1a</em></td>
<td>15.5439</td>
<td>0.99975</td>
</tr>
<tr>
<td><em>Rbm20</em></td>
<td>16.02744</td>
<td>0.99995</td>
</tr>
<tr>
<td><em>Actl5</em></td>
<td>16.25449</td>
<td>0.99995</td>
</tr>
<tr>
<td><em>Xpnpepl</em></td>
<td>18.07143</td>
<td>1</td>
</tr>
</tbody>
</table>

Hotspot elucidation and more generally QTL gene candidate search are data-driven research processes which integrate heterogeneous types of information (31): we have illustrated the use of correlation measurements. Other data types can include Transcription Factor Binding Site (TFBS) modules investigation, gene
4. Notes

1. **Computational capacity issues**

   Some of the protocol steps (mapping, permutation procedure) can be computationally very intensive. If available, it is advisable to use a multi-core machine or a cluster of computers to perform these steps. The jobs can easily be separated by groups of probes, since every probe is here mapped separately. R/Parallel (33) is a useful R package, which allows R to run iterative tasks in parallel on multiple processors. Another trick that can be used to reduce the computing time is to drop redundant markers (neighboring markers with identical genotypes for all samples) at the start of the analysis.

   Memory issues can also arise when huge matrices are created within R. The amount of memory used by R is, for example, limited to 1 GB in the default windows setup of the program. This memory limit can be extended using the command `memory.limit(size_MB)`. However, the maximum memory cannot exceed the physical memory available in the computer. A possible workaround is to divide the traits into smaller entities and to write intermediary results to files.

2. **Affymetrix-related issues**

   Affymetrix arrays differ from alternative expression profiling technologies by their use of probe sets made of multiple (10–16) probes targeting one gene. While a number of studies have focused on probe set-summarized data to perform eQTL mapping, we suggest using a more subtle approach taking into account probe-level intensities. This approach has been extensively described in (34).

3. **Alternative software solutions**

   R/QTL (35) is an R package which includes many functions for mapping, including an algorithm to infer missing genotype data using Hidden Markov Models. GeneNetwork (www.genenetwork.org (11)) also offers eQTL analysis for user uploaded data, one trait at a time, and genome-wide analysis tools for a number of published datasets.

4. **Alternative Illumina data pre-processing**

   Compared with Affymetrix, for example, Illumina is a relatively new technology and standard analysis guidelines have yet to emerge. While in this chapter we illustrate our eQTL
analysis with raw probe-summarized data as output by BeadStudio, there are alternative possibilities which make use of bead-level intensity data. See, for example, work by Dunning et al. (36, 37) for some methods and software.

5. **Probe (re-)annotations**

The correct annotation of probes is a critical aspect of any microarray analysis. It is especially crucial in the case of eQTL studies since the presence of subtle differences in the probe target sequences (SNPs) between parental lines can produce technical false-positive eQTLs (see Note 9). Annotation files provided by array manufacturers tend to be incomplete and outdated and do not include genetic variation information across different strains.

The strategy of probe re-annotation should therefore comprise a systematic BLAT search of each of the probe sequences against the latest genome assembly build, combined with a mining of polymorphism databases. This is obviously a gruesome task for which there are a few software tools available (38, 39, 40).

6. **Batches and hidden factors**

Microarray data are known to be very sensitive to the effect of batches, which can create artificial correlation. In the context of eQTL mapping, these effects can act as confounding factors and cause multiple spurious genetic linkages, often forming apparent hotspots: if the confounding factor influences many genes (as is the case with microarray batches) and if there is, by chance, correlation of that factor with the genotypes at a certain genomic locus, then all genes will artificially map to that region, misleadingly suggesting the presence of a master regulator (34). If the confounding factor is known, it is possible to correct for its effect by adapting the mapping model. In the single-marker mapping case:

\[
Y_i = m_i + B + G_i + e_i
\]

where \( B \) is the batch factor.

Caution: if multiple environments are used, it is usually required to account for the batch effects in an environment-specific fashion. In this case, an appropriate model would be

\[
Y_i = m_i + B + T + B^T + G_i + G_j^T + e_i
\]

The \( B \times T \) allows for a more careful batch effect correction: for example, if one gene was only expressed in one of the environments (in our example one tissue), then the batch
effect could affect only the gene in that tissue, and should not be corrected in the samples belonging to other environments.

7. Advanced model selection procedures

The selection of relevant co-factors and interaction terms in generalized models, and particularly in the context of QTL mapping, has been widely discussed in the scientific literature. Mapping of multiple QTLs and epistasis testing can be seen as model selection problems. For examples, see (41, 42).

8. The case of sex chromosomes

While most of the Y chromosome does not undergo recombination, the recombination rate of the X chromosome is slower than that of the autosomes. This has important consequences on the detection of significant QTLs. For a comprehensive view of these issues, see (43).

9. Probe hybridization artifacts

When several probes are available for the same gene, it is not uncommon to observe a difference in the mapping results of those probes: “the probes tell different stories” or statistically there is eQTL by probe interaction (34). This can be explained either by biological mechanisms (alternative splicing) or by technical artifacts. Such technical artifacts may arise when a polymorphism is present in the sequence targeted by a probe (44). If the probe was designed specifically based on the genome sequence of one of the parental strains, it is possible that some polymorphism causes the other genotype mRNA products to have a weaker binding affinity and thus a lower signal. Such effects will yield spurious local eQTL linkages. If the probes have been designed specifically based on the sequence of one of the two parental strains (say, strain A, and not strain B), it is possible to estimate roughly the number of local eQTLs affected by this issue. For example, if 65% of local eQTLs are linked with a higher expression of the gene for the A allele, while for the other 35% local eQTLs the B allele is more highly expressed. This contrasts with the 50–50% expected without hybridization effect. In this case, we would expect 65–35 = 30% of eQTLs to be caused by this hybridization difference rather than by a real differential expression effect.

References


