# Comparing Responders with Nonresponders

Keith A. Baggerly, E. Shannon Neeley and Kevin R. Coombes

October 19, 2007

## 1 Introduction

One of the major results of Dressman et al is that it is possible to use microarray signatures to distinguish responders (CR) from nonresponders (NR). This was initially established using a training set of 83 samples and validated on a testing set of 36 samples. We do not know how the samples were initially divided into training and test sets, but we will assume that signatures found to work on a randomly chosen subset of the data should also be visible in the complete set. They also used a routine for shotgun stochastic search (SSS), which, as its name implies, returns answers that are subject to random variation.

We pose the question of whether responders differ from nonresponders in a different, and hopefully simpler, fashion. Specifically, if we start with all of the data and perform two-sample t-tests comparing CRs and NRs gene by gene,

- Does a plot of the p-values overall show an overabundance of small values, suggesting that a difference exists?
- Does a plot of the p-values for the specific genes that they report show that most of them are small, suggesting that these genes are particularly effective?

## 2 Options and Libraries

```
> options(width = 80)
```

In order to perform many of these comparisons in parallel, we make use of the ClassComparison and ClassDiscovery libraries, which are available from our website, `http://bioinformatics.mdanderson.org`

```
> library(affy)
> library(ClassComparison)
> library(ClassDiscovery)
```

## 3 Load and Sort Raw Data

```
> load(file.path("RDataObjects", "ovcaRMAFromCEL.Rda"))
> load(file.path("RDataObjects", "ovcaRMAFromXLS.Rda"))
> load(file.path("RDataObjects", "reportedGenesGEO.Rda"))
> load(file.path("RDataObjects", "clinicalInfo.Rda"))
> load(file.path("RDataObjects", "celRunDate.Rda"))
```

Before using these later, let's make sure that all of the sample IDs are in the same order throughout. We will take the clinicalInfo ordering to be canonical.

```
> ovcaRMAFromCEL <- ovcaRMAFromCEL[, rownames(clinicalInfo)]
> ovcaRMAFromXLS <- ovcaRMAFromXLS[, rownames(clinicalInfo)]
> celRunDate <- celRunDate[rownames(clinicalInfo)]
```

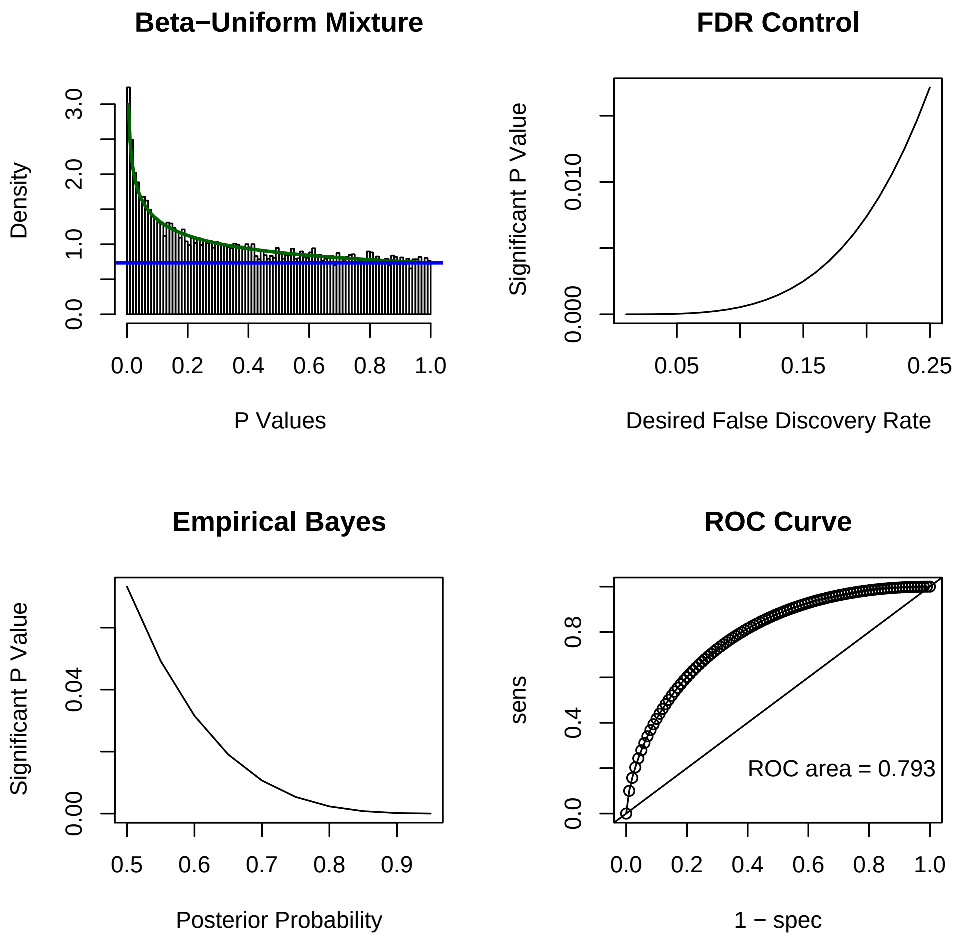
## 4 Comparing All Genes

We begin by computing t-statistics for all genes.

```
> tFromCEL <- MultiTtest(ovcaRMAFromCEL, clinicalInfo$Response)
> tFromXLS <- MultiTtest(ovcaRMAFromXLS, clinicalInfo$Response)
```

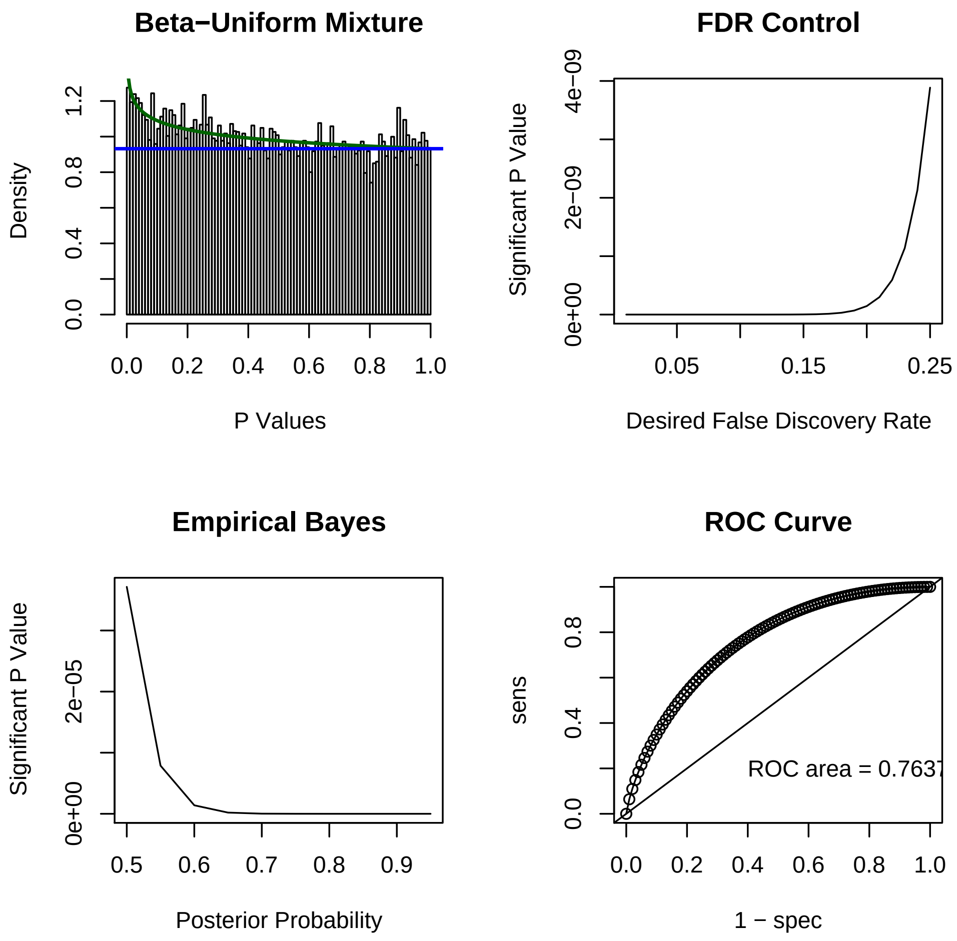
Next, we look at the distribution of p-values for each contrast to see if there are lots of small p-values. We begin with the quantifications derived from the CEL files. We fit the distribution of p-values using a beta-uniform mixture (BUM) model.

```
> bumFromCEL <- Bum(tFromCEL@p.values)
> image(bumFromCEL)
```



This looks somewhat promising; there do indeed seem to be more small p-values than we might expect just by chance. Now we try the same approach using the quantifications derived from the XLS file.

```
> bumFromXLS <- Bum(tFromXLS@p.values)
> image(bumFromXLS)
```

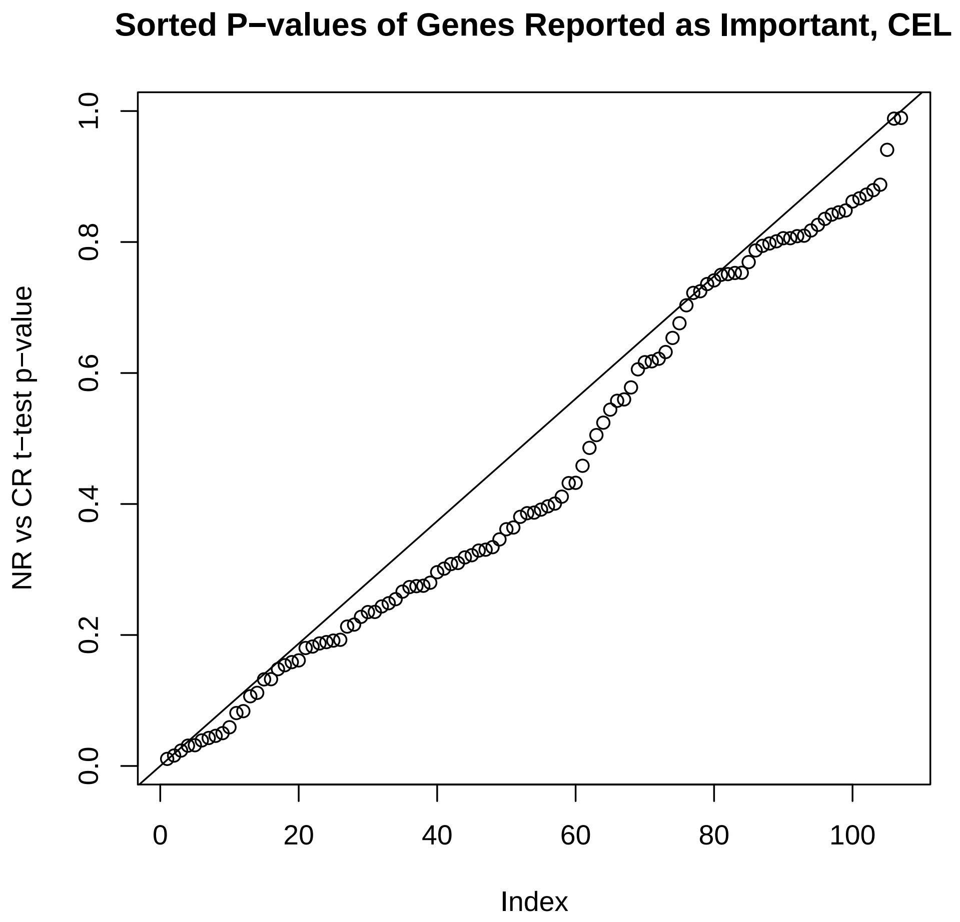


We still see some enrichment, but nowhere near as much. To the extent that this is a random relabeling of the samples, this suggests that we may need to use permutation tests to assess enrichment as there are more than we might expect using the "null" hypothesis.

## 5 Looking at the Reported Genes

Here, we are hoping to see many small p-values in the reported set of genes, as this might indicate that these genes are particularly important for examining the responder/nonresponder contrast. Again, we try this first using the quantifications derived from the CEL files.

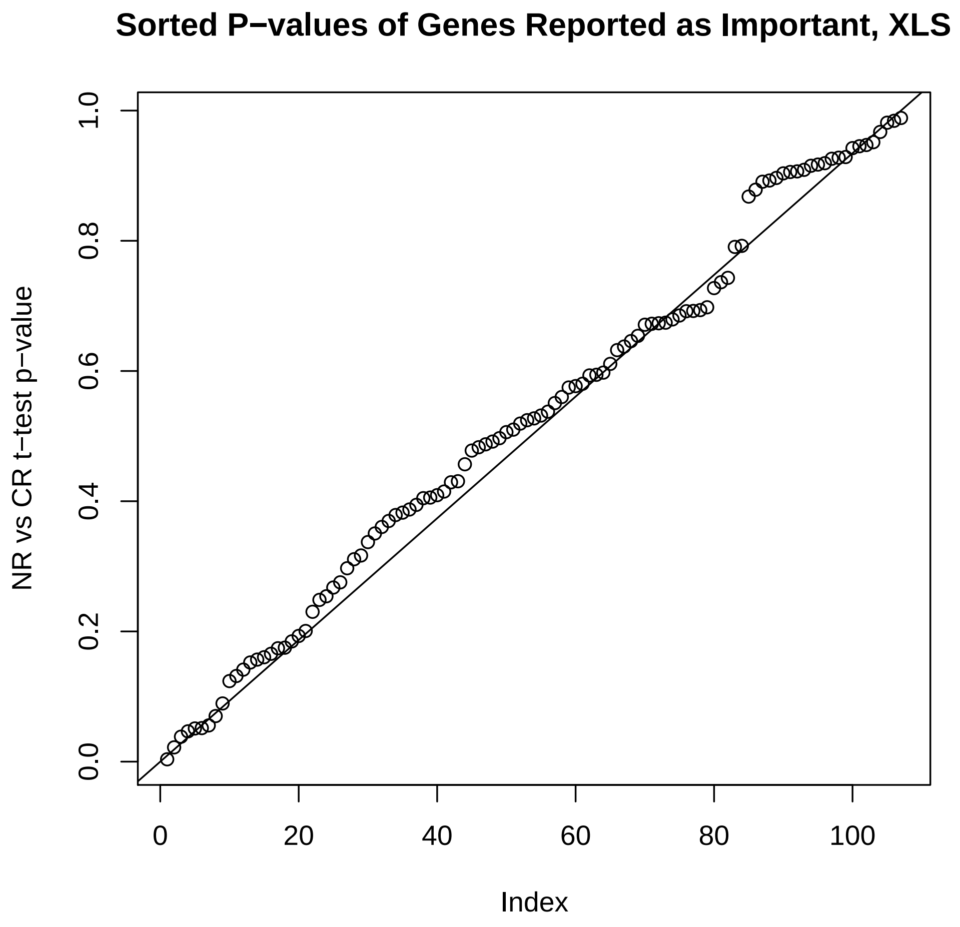
```
> plot(sort(tFromCEL@p.values[match(rownames(reportedGenesGEO),
+     rownames(ovcaRMAFromCEL))]), xlab = "Index", ylab = "NR vs CR t-test p-value",
+     main = "Sorted P-values of Genes Reported as Important, CEL")
> abline(0, 1/dim(reportedGenesGEO)[1])
```



This list of genes shows no particular enrichment for very small p-values; the distribution is very close to uniform. This test provides no evidence that these genes are particularly important.

Next, we repeat the above contrast using the quantifications derived from the XLS file.

```
> plot(sort(tFromXLS@p.values[match(rownames(reportedGenesGEO),
+     rownames(ovcaRMAFromXLS))]), xlab = "Index", ylab = "NR vs CR t-test p-value",
+     main = "Sorted P-values of Genes Reported as Important, XLS")
> abline(0, 1/dim(reportedGenesGEO)[1])
```



Again, we find no evidence that the genes reported are particularly relevant for characterizing the NR/CR split.

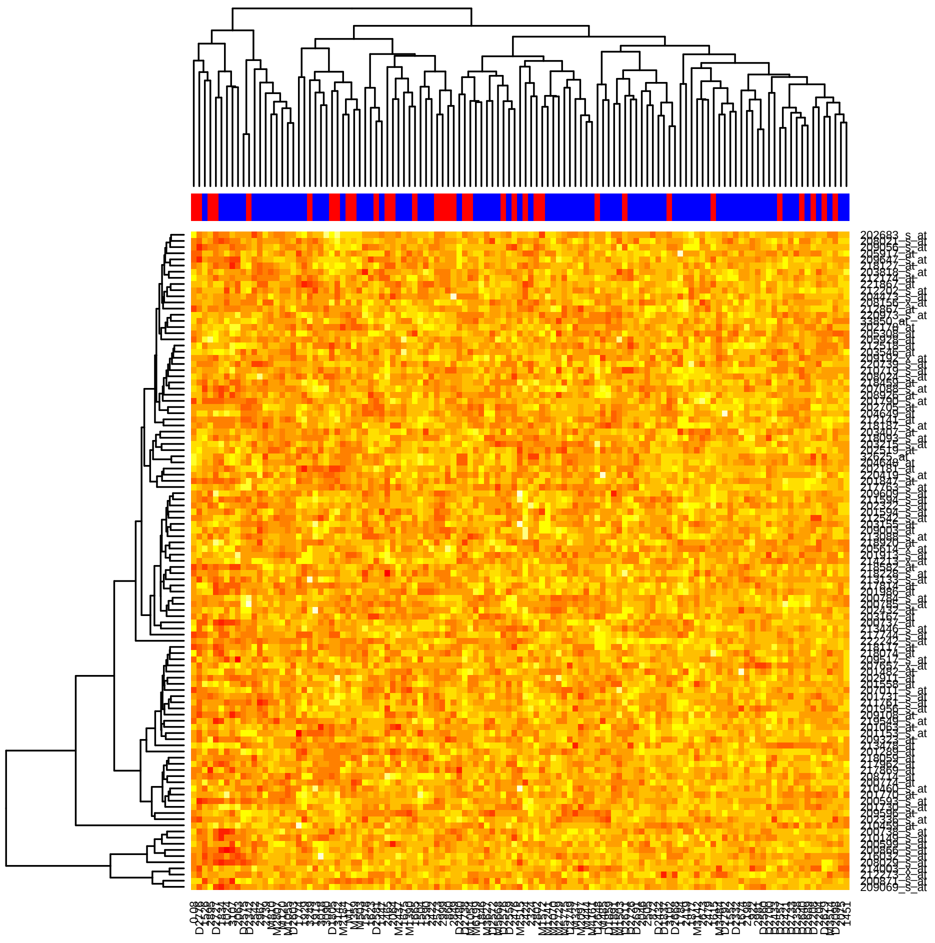
Now, it may be that the structure being found is inherently multivariate in nature, but we don't see a very good way to test this. What we can do is try something simply like hierarchical clustering with a heatmap to see if some structure that we can discern visually presents itself.

Let's define a vector of colors to represent CR/NR status.

```
> respColors <- rep("blue", 119)
> respColors[clinicalInfo$Response == "NR"] <- "red"
```

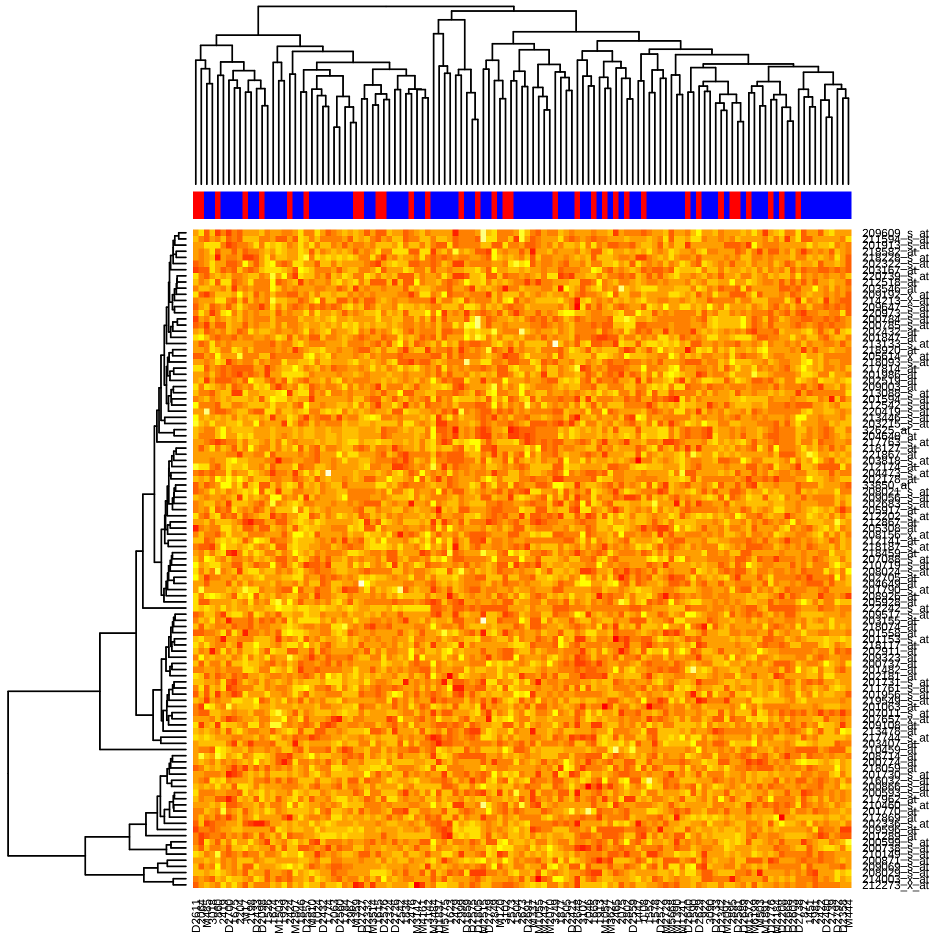
Now let's construct a heatmap using the CEL numbers.

```
> heatmap(ovcaRMAFromCEL[rownames(reportedGenesGEO), ], ColSideColors = respColors)
```



We do not see any real structure, and the clustering does not separate CRs from NRs. We now repeat this using the XLS numbers.

```
> heatmap(as.matrix(ovcaRMAFromXLS[rownames(reportedGenesGEO),
+     ]), ColSideColors = respColors)
```



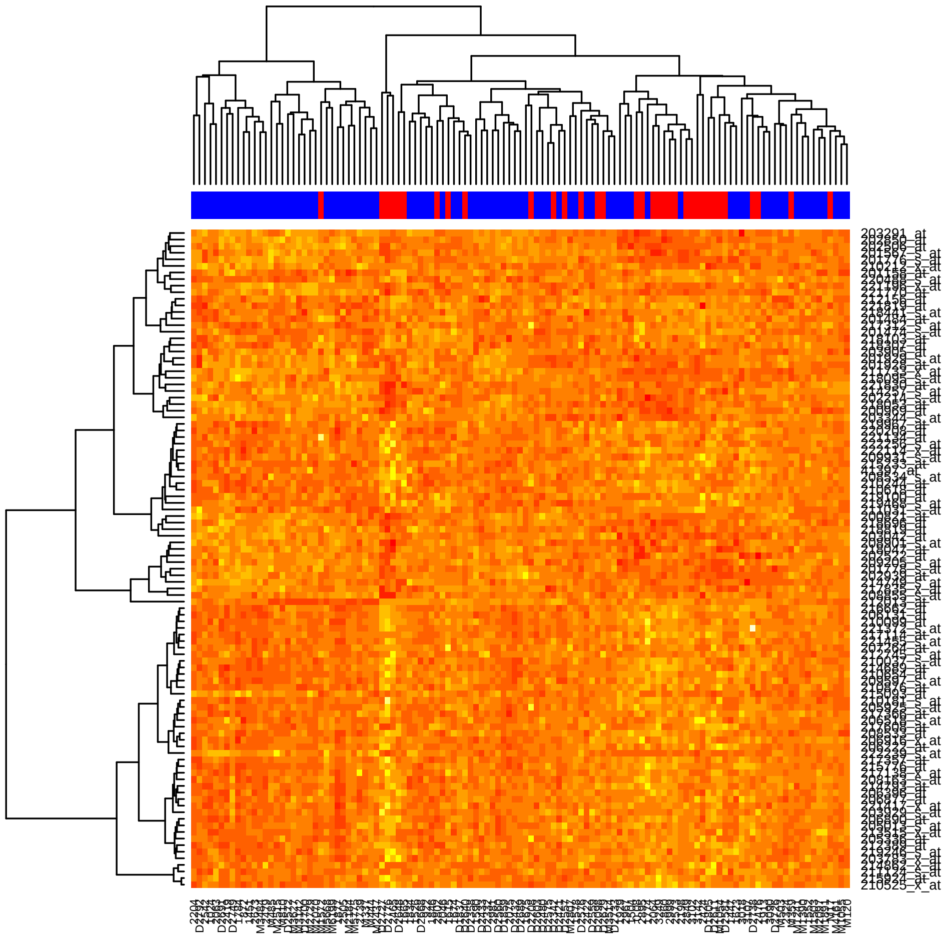
The story is the same here as it was above. We see no stark structure or separation of CRs from NRs.

## 6 More Focused Contrasts

### 6.1 A Heatmap We Think We Can Predict

We'd like to understand the CR/NR boundary better. To explore this boundary, we'll first try something where we think we'll be able to predict the outcome – we'll produce a heatmap using the top 100 genes as selected by t-test p-values. We expect this to split the CRs and NRs pretty well.

```
> heatmap(ovcaRMAFromCEL[tFromCEL@p.values <= sort(tFromCEL@p.values)[100],
+     ], ColSideColors = respColors)
```



Actually, what we get isn't quite what we expected. While there is a pretty clear split, the smaller cluster contains 33 CRs and 1 NR, not the 34 NRs we had expected at first. While this does suggest that there may be a story, it also suggests that it may not be quite as simple as we thought.

## 6.2 Trying to explain Group Membership

Let's take a closer look at the 34 patients in the smaller cluster. We want to see if there is simple explanation to account for the observed groups.

First we find which patients belong to which group.

```
> hc <- hclust(dist(t(ovcaRMAFromCEL[tFromCEL@p.values <= sort(tFromCEL@p.values)[100],
+     ])))
> group <- cutree(hc, 2)
```

Here we double check that the groups we defined are in the same order as the clinicalInfo.

```
> sum(rownames(clinicalInfo) != names(group))
```

```
[1] 0
```

It looks good. Is group membership associated with any of the clinical variables? Here group 1 is the larger group (85 members) and group 2 is the smaller (34 members).

```
> table(group, clinicalInfo[, "Response"])

group CR NR
    1 52 33
    2 33  1

> table(group, clinicalInfo[, "Stage"])

group  2  3  4
    1  1 68 16
    2  0 30  3

> table(group, clinicalInfo[, "Grade"])

group     2/3  ?  1  2  3  4 UNK
    1  1    2  1  4 41 34  1   1
    2  0    0  0  0 13 21  0   0

> table(group, clinicalInfo[, "Debulk"])

group  O  S
    1 43 42
    2 21 13
```

Actually it doesn't appear that group membership is associated with any of the clinical variable except response, which we've already noted. However, there is a significant association with run date.

```
> x <- table(group, celRunDate)
> x

     celRunDate
group 2002-09-20 2002-10-23 2002-11-12 2002-12-16 2002-12-21 2003-01-03
    1          5          9          9          1          3          3
    2          5          0          0          0          0          8
     celRunDate
group 2003-05-30 2003-06-26 2004-03-09 2004-03-16 2004-04-20 2004-05-18
    1         10          1          9          2          0         11
    2          0          0          7          4          5          4
     celRunDate
group 2004-05-21 2004-05-27 2004-06-22 2004-06-23
    1          7          6          1          8
    2          0          1          0          0

> fisher.test(x, simulate.p.value = T, B = 5000)
```

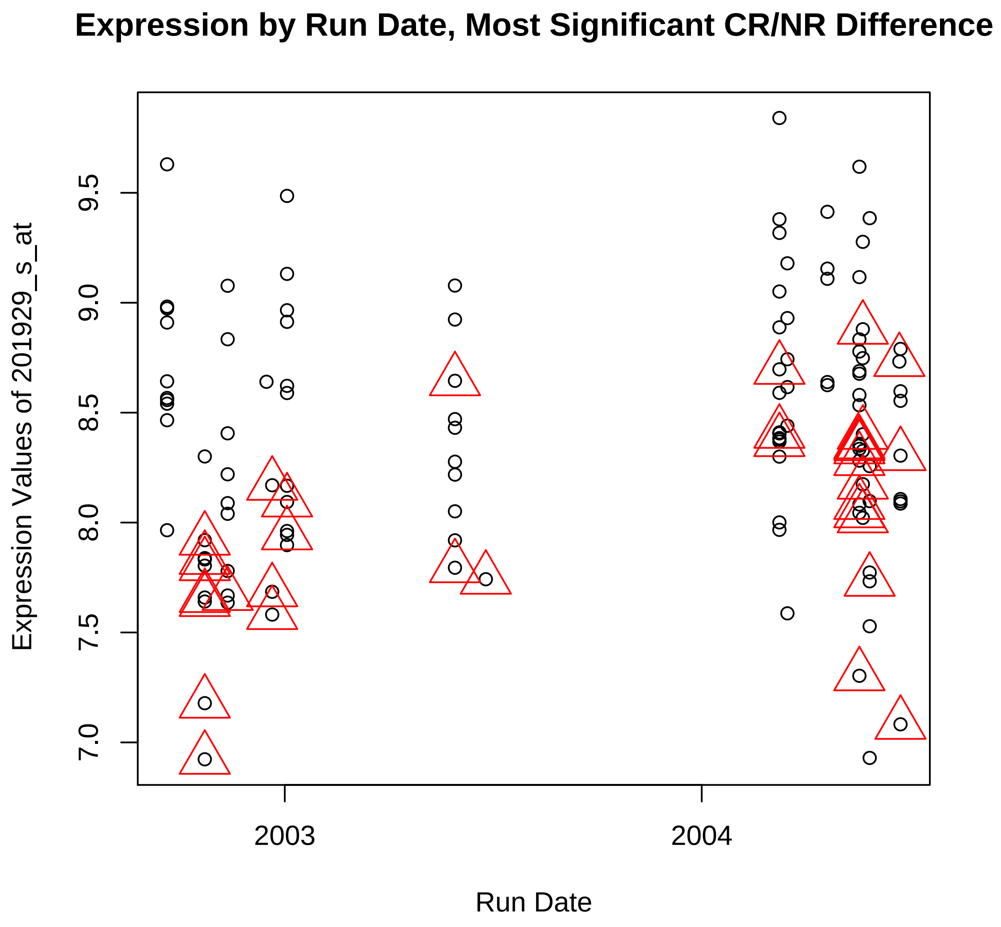
```
        Fisher's Exact Test for Count Data with simulated p-value (based on
        5000 replicates)

data:  x
p-value = 0.0002000
alternative hypothesis: two.sided
```

As in ovca04, the p-value is equal to 1/5000 which is the smallest p-value possible with 5000 simulations. The split observed is more extreme than any of our simulations. Most of the stucture observed in the heat map above seems to be driven by run date.

## 6.3 The Single "Best" Gene

We'll take a look at the gene with the smallest p-value using the CEL data, and we plot the expression values as a function of run date.

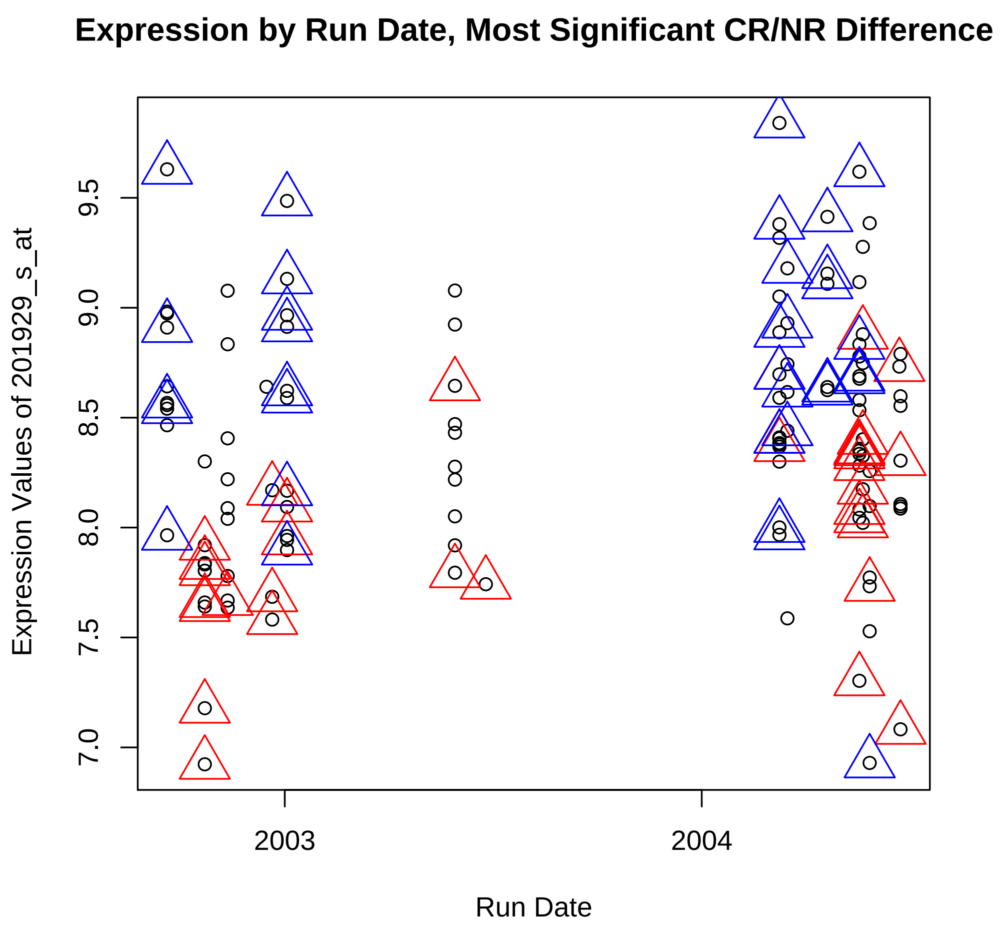
```
> plot(celRunDate, ovcaRMAFromCEL[which.min(tFromCEL@p.values),
+     ], xlab = "Run Date", ylab = paste("Expression Values of",
+     rownames(ovcaRMAFromCEL)[which.min(tFromCEL@p.values)]),
+     main = "Expression by Run Date, Most Significant CR/NR Difference")
> points(celRunDate[clinicalInfo$Response == "NR"], ovcaRMAFromCEL[which.min(tFromCEL@p.values),
+     clinicalInfo$Response == "NR"], col = "red", pch = 2, cex = 3)
```



This gene does indeed appear to have lower expression in the NRs (marked red), but it also points out a problem – some of the biggest differences are due to the first two run dates, where survival and response status are largely confounded with batch.

The following plot also shows members from the smaller group from the t-test marked in blue. There are no samples from group 2 run on the second date.

```
> plot(celRunDate, ovcaRMAFromCEL[which.min(tFromCEL@p.values),
+     ], xlab = "Run Date", ylab = paste("Expression Values of",
+     rownames(ovcaRMAFromCEL)[which.min(tFromCEL@p.values)]),
+     main = "Expression by Run Date, Most Significant CR/NR Difference")
> points(celRunDate[clinicalInfo$Response == "NR"], ovcaRMAFromCEL[which.min(tFromCEL@p.values),
+     clinicalInfo$Response == "NR"], col = "red", pch = 2, cex = 3)
> points(celRunDate[group == 2], ovcaRMAFromCEL[which.min(tFromCEL@p.values),
+     group == 2], col = "blue", pch = 2, cex = 3)
```



## 6.4 Fitting ovcaRMAFromCEL with Run Date

We'd like to try correcting for run date and then assessing the significance of the split. There are 16 distinct run dates.

We will fit the gene expression values as a function of run date, and then fit expression as a function of clinical response after correcting for run date.

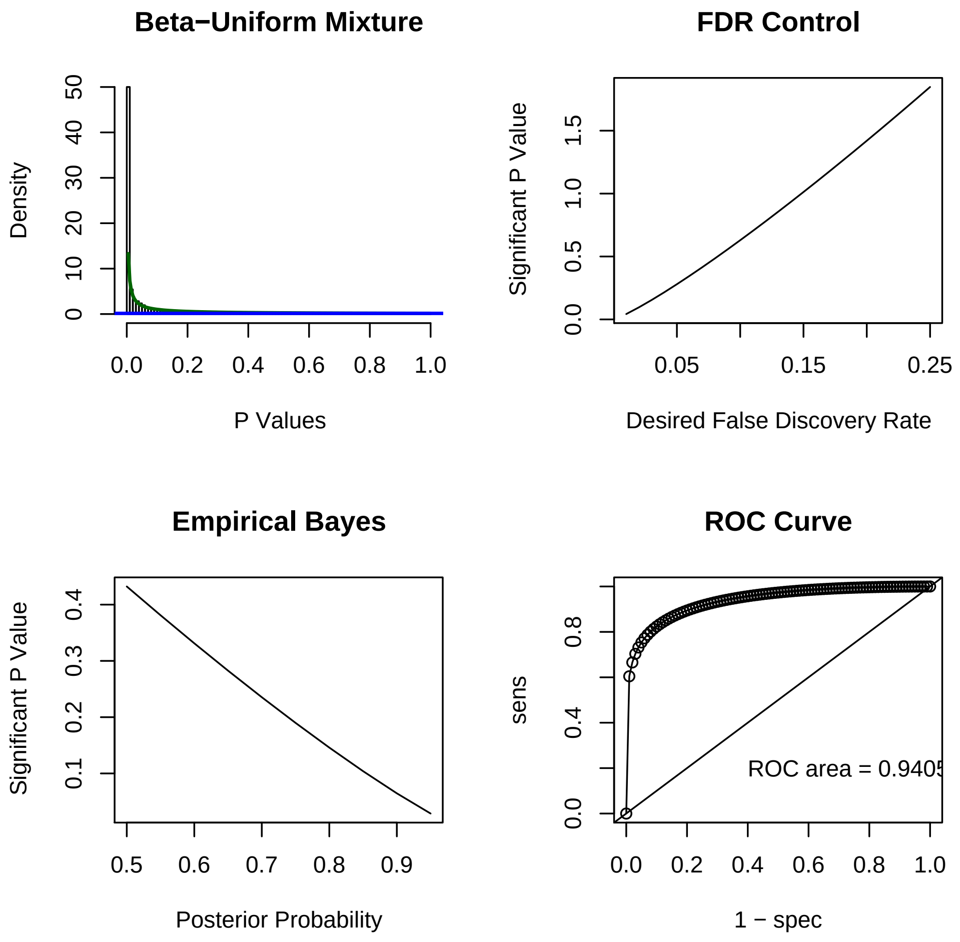
```
> celRunDatef <- as.factor(celRunDate)
> runDateModelForm <- Y ~ celRunDatef
> runDateModelLMAll <- MultiLinearModel(runDateModelForm, data.frame(celRunDatef = celRunDatef,
+     Response = clinicalInfo$Response), ovcaRMAFromCEL)
> ovcaRMAFromCELResids <- ovcaRMAFromCEL - t(runDateModelLMAll@predictions)
> runDatefullModelForm <- Y ~ celRunDatef + Response
> runDatefullModelLMAll <- MultiLinearModel(runDatefullModelForm,
+     data.frame(celRunDatef = celRunDatef, Response = clinicalInfo$Response),
+     ovcaRMAFromCEL)
> runDateReducedModelLMAll <- anova(runDatefullModelLMAll, runDateModelLMAll)
```

Having fit models, let's see what things look like for run date and response status. First, we build BUM models.

```
> bumRunDate <- Bum(runDateModelLMAll@p.values)
> bumRespRunDate <- Bum(runDateReducedModelLMAll$p.values)
```

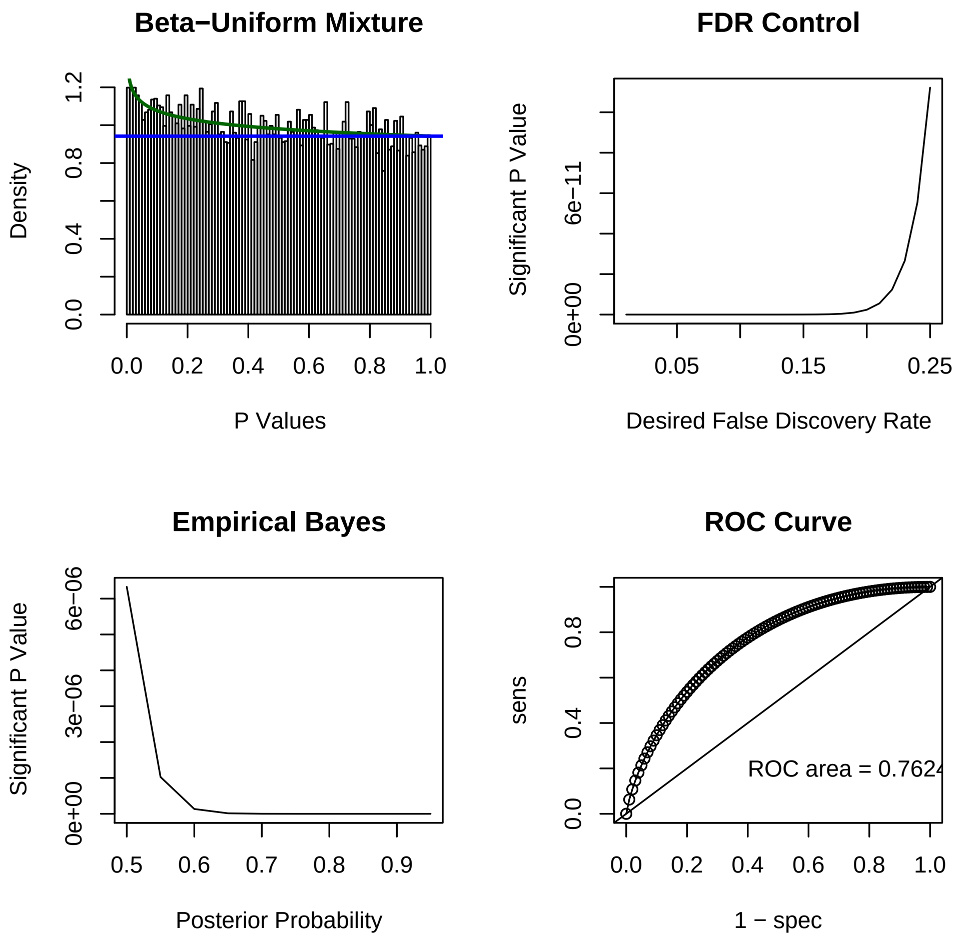
Now, we look at the results for run date.

```
> image(bumRunDate)
```



This plot is off the scale; almost all of the genes show significant differences associated with run date. Then, we look at things after accounting for run date.

```
> image(bumRespRunDate)
```



Almost all of the enrichment has vanished, and the plot looks similar to what we obtained for the XLS quantifications where the structure was attenuated due to scrambling. There could be a some small differences, but there isn't much that remains significant after controlling for run date.

## 6.5 Fitting ovcaRMAFromXLS with Run Date

We can also see if trying to account for run date effects markedly alters the results when the XLS quantifications are used. As the scrambling is not wholly random, some of the batch structure may be preserved.

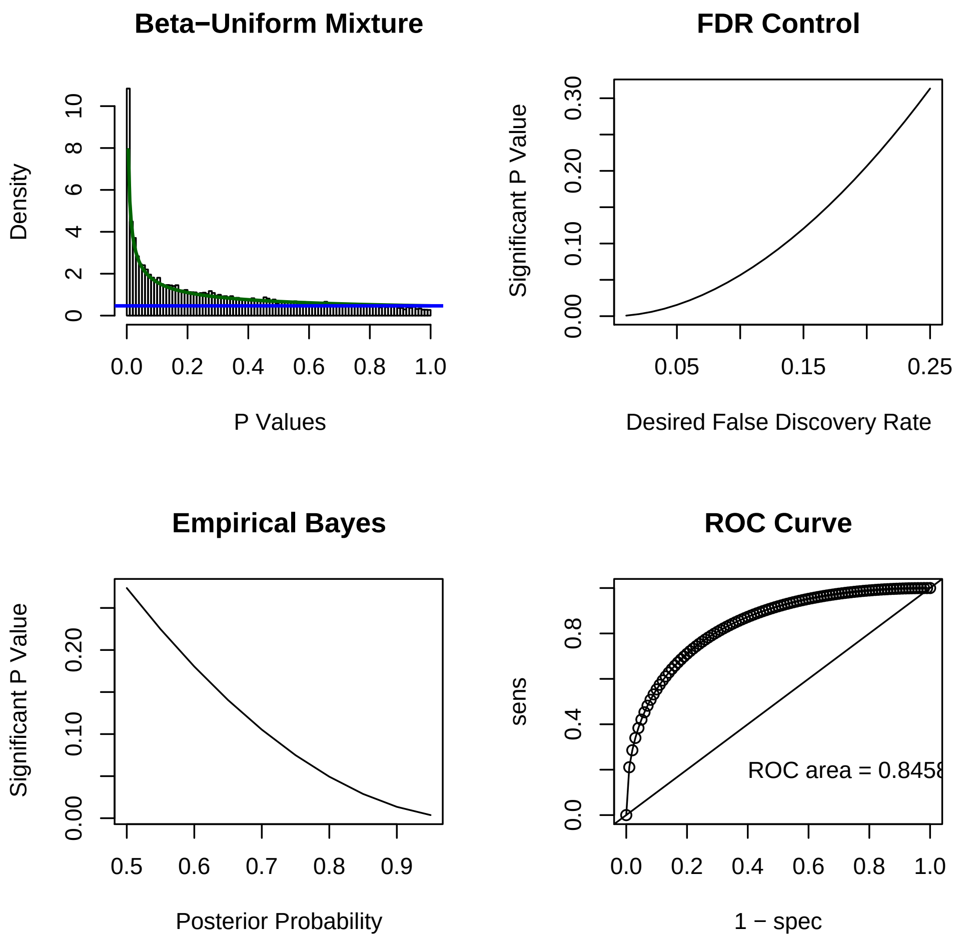
```
> runDateModelXLSForm <- Y ~ celRunDatef
> runDateModelXLSLMAll <- MultiLinearModel(runDateModelXLSForm,
+     data.frame(celRunDatef = celRunDatef, Response = clinicalInfo$Response),
+     ovcaRMAFromXLS)
> ovcaRMAFromXLSResids <- ovcaRMAFromXLS - t(runDateModelXLSLMAll@predictions)
> runDatefullModelXLSForm <- Y ~ celRunDatef + Response
> runDatefullModelXLSLMAll <- MultiLinearModel(runDatefullModelXLSForm,
+     data.frame(celRunDatef = celRunDatef, Response = clinicalInfo$Response),
+     ovcaRMAFromXLS)
> runDateReducedModelXLSLMAll <- anova(runDatefullModelXLSLMAll,
+     runDateModelXLSLMAll)
```

Having assembled the models, let's assemble the BUMs.

```
> bumXLSRunDate <- Bum(runDateModelXLSLMAll@p.values)
> bumRespXLSRunDate <- Bum(runDateReducedModelXLSLMAll$p.values)
```

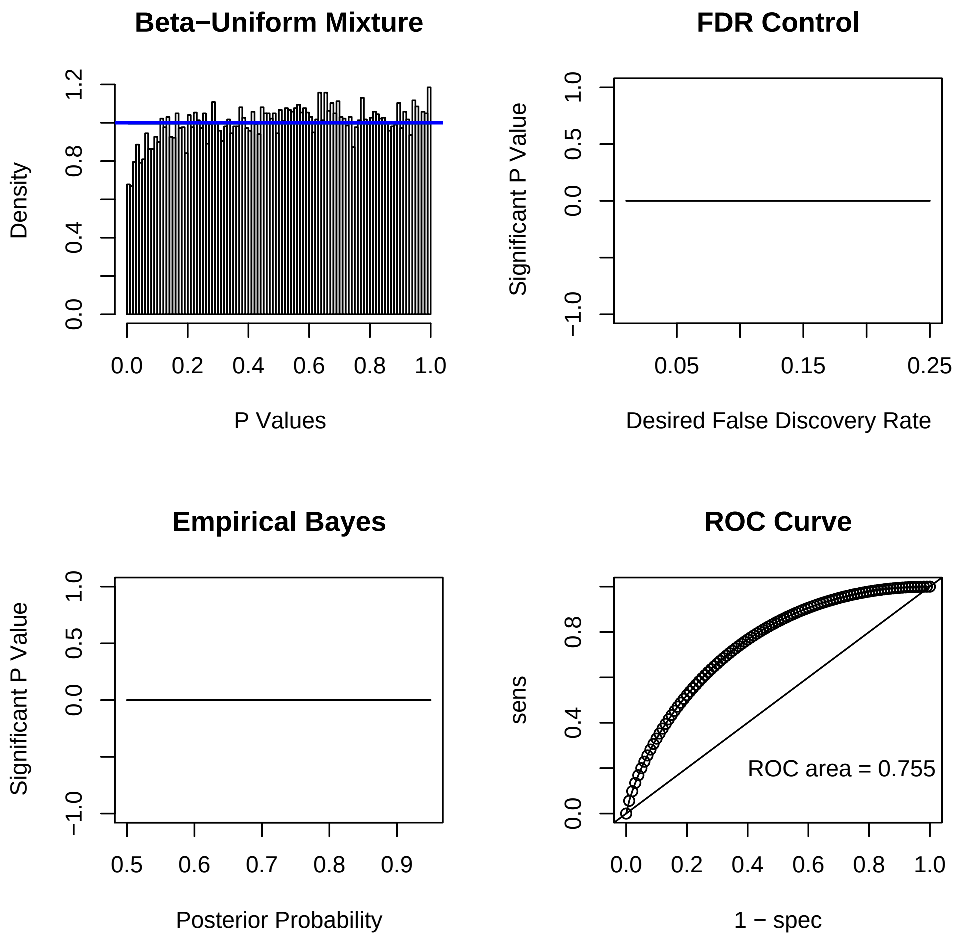
Now let's see what the plots look like.

```
> image(bumXLSRunDate)
```



Even here, there are much larger effects associated with batch than with the CR/NR divide. Finally, let's look for signs of response after correcting for run date.

```
> image(bumRespXLSRunDate)
```



There is nothing remains significant after controlling for run date; if anything, we're now overcorrecting a bit.

## 6.6 Grouping Dates into Batches

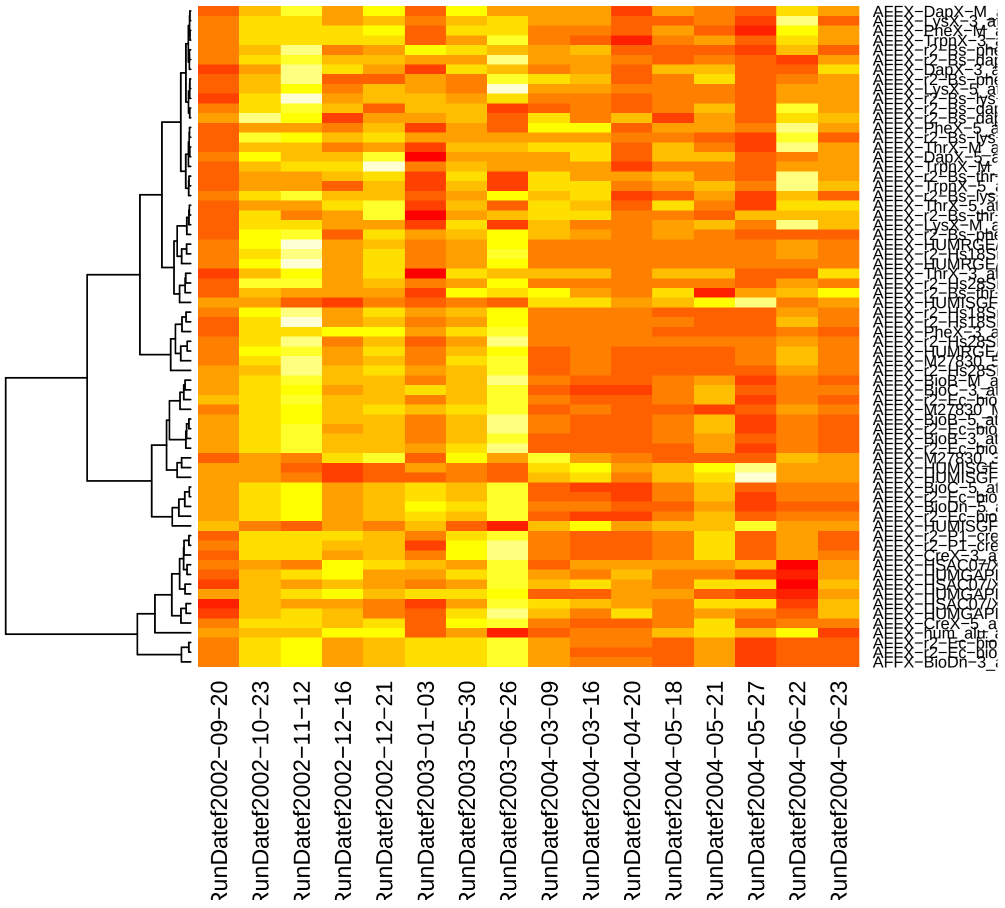
As noted earlier, there are 16 distinct run dates, which is probably a bit much. We would like to see if there is a way to combine some of the dates into batches without losing information.

As a first pass, we look at just the Affymetrix control probesets. We compute the mean expression levels for these probesets across run dates and look for obvious clusters.

```
> probeNames <- rownames(ovcaRMAFromCEL)
> shortNames <- substr(probeNames, 1, 4)
> controls <- shortNames == "AFFX"
> ovcaRMAControls <- ovcaRMAFromCEL[controls, ]
> runDateModelForm <- Y ~ -1 + celRunDatef
> runDateModelControls <- MultiLinearModel(runDateModelForm, data.frame(celRunDatef = celRunDatef),
+     ovcaRMAControls)
> coeffsControl <- runDateModelControls@coefficients
```

We can see clusters by looking at a heat map of the the mean expression values.

```
> heatmap(t(coeffsControl), Colv = NA)
```



The heatmap suggests that there are 7 run batches. The dates in these batches are:

1. 9-20-2002
2. 10-23-2002, 11-12-2002
3. 12-16-2002, 12-21-2002
4. 01-03-2003
5. 05-30-2003

6. 06-26-2003

7. 03-09-2004, 03-16-2004, 04-20-2004, 05-18-2004, 05-21-2004, 05-27-2004, 06-22-2004, 06-23-2004

Note that the sharpest cluster distinction is between those with run dates before 2004 and those in 2004. The early data is more variable, with 8 run dates grouping into 6 batches. All of the more recent data groups into one batch. As a side note, all of the 2004 arrays appear to have been run by the same operator on the same machine (all of the DatHeaders have the name "Robyn" imbedded).

We can also look at the probesets with the smallest ANOVA p-value for a test comparing the mean expression across run dates.

```
> celRunDatef <- as.factor(celRunDate)
> runDateModelForm <- Y ~ -1 + celRunDatef
> runDateModel <- MultiLinearModel(runDateModelForm, data.frame(celRunDatef = celRunDatef),
+     ovcaRMAFromCEL)
> coeffs <- runDateModel@coefficients
> pvals <- runDateModel@p.values
> sum(pvals == 0)
```

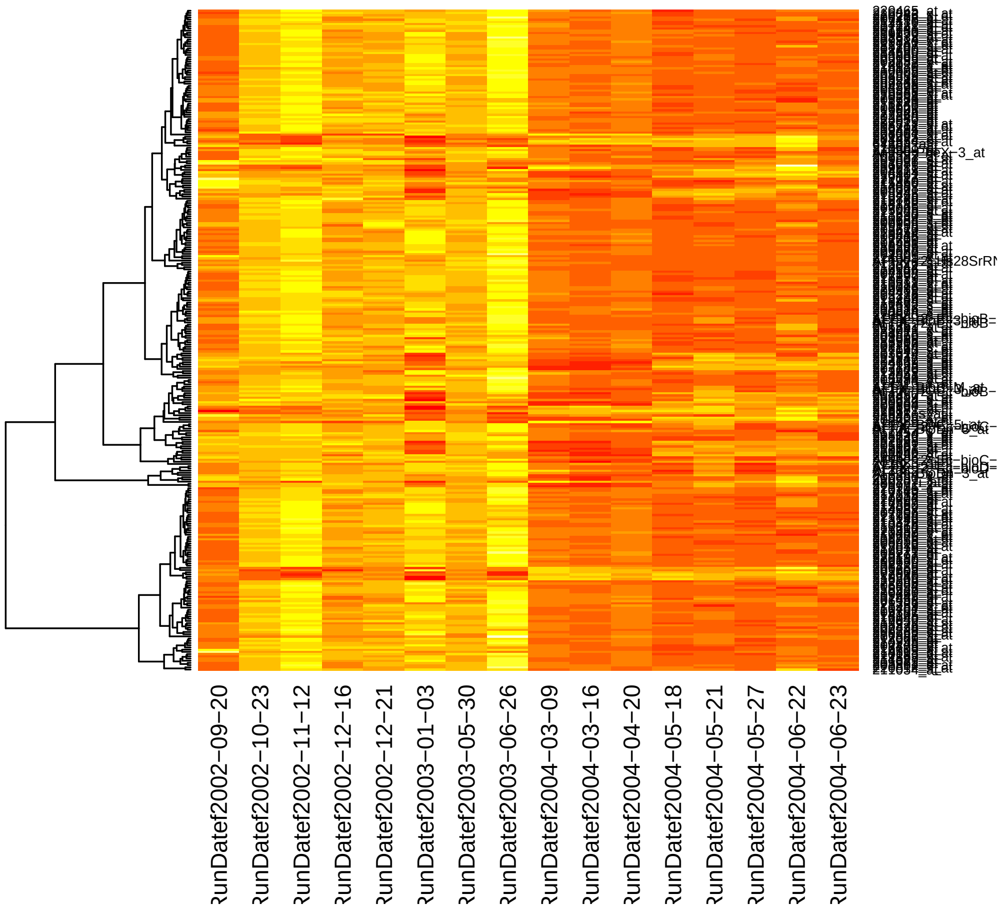
```
[1] 299
```

There are 299 probesets with p-values so small they were machine-rounded to zero. We will look at the mean expression levels across run dates for these probesets.

```
> smallPval <- pvals == 0
> smallCoefs <- coeffs[, smallPval]
```

A heatmap shows the clusters.

```
> heatmap(t(coeffs2), Colv = NA)
```



The visible clusters match those we saw using the Affy controls. Next, we define these batches explicitly and see what the NR/CR split looks like after controlling for batch.

```
> runBatch <- rep(7, 119)
> runBatch[celRunDate == "2002-09-20"] <- 1
> runBatch[celRunDate == "2002-10-23"] <- 2
> runBatch[celRunDate == "2002-11-12"] <- 2
> runBatch[celRunDate == "2002-12-16"] <- 3
> runBatch[celRunDate == "2002-12-21"] <- 3
> runBatch[celRunDate == "2003-01-03"] <- 4
> runBatch[celRunDate == "2003-05-30"] <- 5
> runBatch[celRunDate == "2003-06-26"] <- 6
> runBatch[celRunDate == "2004-03-09"] <- 7
> runBatch[celRunDate == "2004-03-16"] <- 7
> runBatch[celRunDate == "2004-04-20"] <- 7
> runBatch[celRunDate == "2004-05-18"] <- 7
> runBatch[celRunDate == "2004-05-21"] <- 7
> runBatch[celRunDate == "2004-05-27"] <- 7
> runBatch[celRunDate == "2004-06-22"] <- 7
> runBatch[celRunDate == "2004-06-23"] <- 7
> runBatch <- as.factor(runBatch)
> names(runBatch) <- names(celRunDate)
```

## 6.7 Fitting ovcaRMAFromCEL with Batches

We fit the gene expression values as a function of runBatch, and then fit expression as a function of clinical response after correcting for batch.

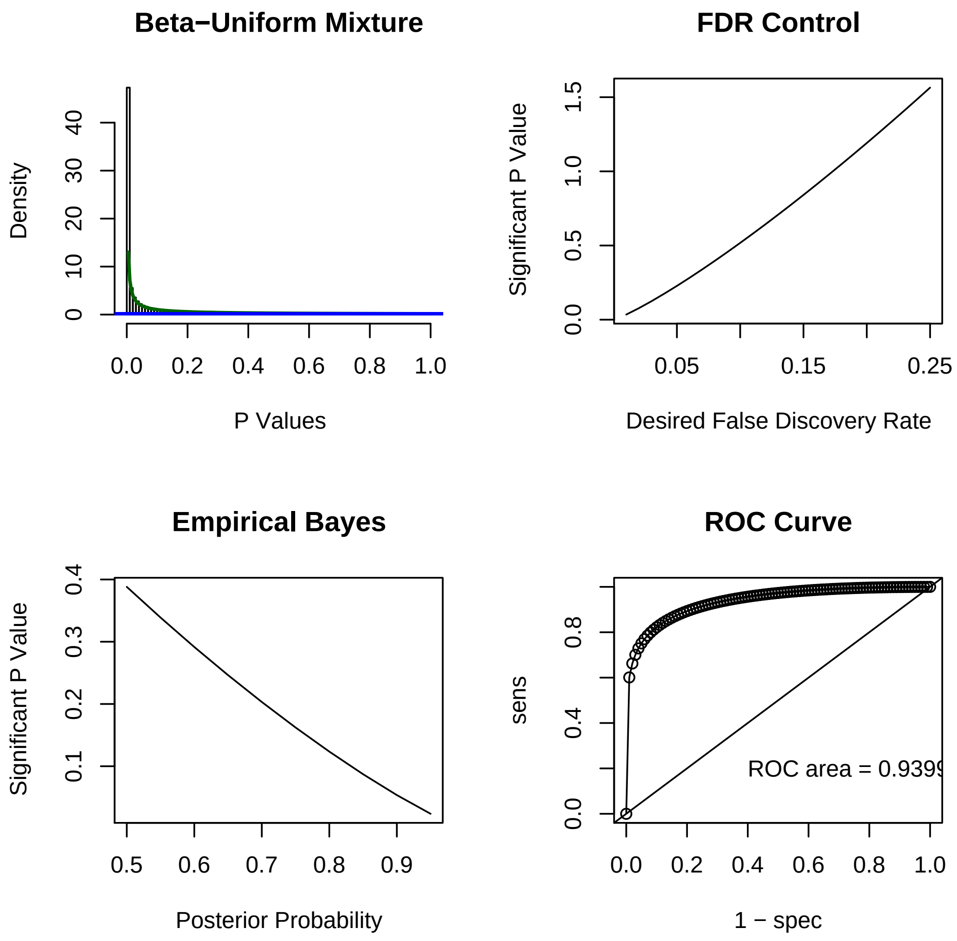
```
> batchModelForm <- Y ~ runBatch
> batchModelLMAll <- MultiLinearModel(batchModelForm, data.frame(runBatch = runBatch,
+     Response = clinicalInfo$Response), ovcaRMAFromCEL)
> ovcaRMAFromCELResids <- ovcaRMAFromCEL - t(batchModelLMAll@predictions)
> fullModelForm <- Y ~ runBatch + Response
> fullModelLMAll <- MultiLinearModel(fullModelForm, data.frame(runBatch = runBatch,
+     Response = clinicalInfo$Response), ovcaRMAFromCEL)
> reducedModelLMAll <- anova(fullModelLMAll, batchModelLMAll)
```

Having fit models, let's see what things look like for batch and response status. First, we build BUM models.

```
> bumBatch <- Bum(batchModelLMAll@p.values)
> bumRespBatch <- Bum(reducedModelLMAll$p.values)
```

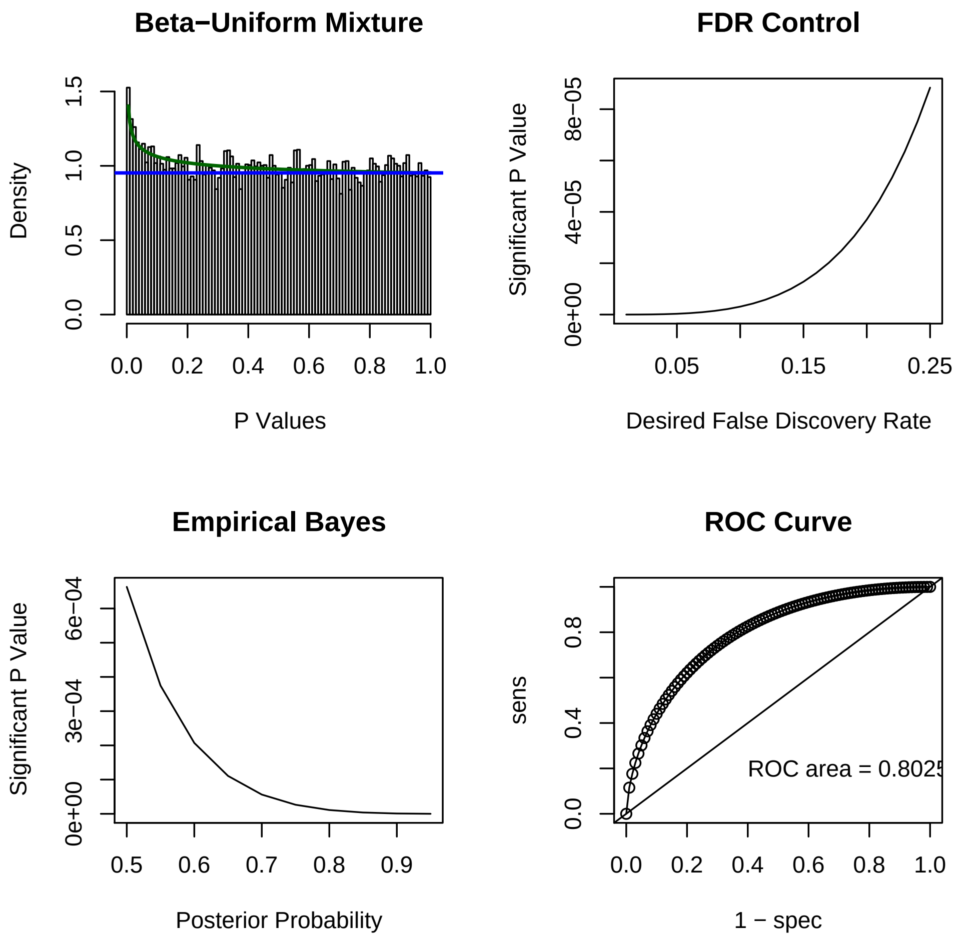
Now, we look at the results for batch.

```
> image(bumBatch)
```



Just as with run date, this plot is off the scale; almost all of the genes show significant differences associated with batch. Then, we look at things after accounting for batch.

```
> image(bumRespBatch)
```



There appear to be some more differences than when we corrected for run date, but still far fewer than we saw before applying a correction. Not many would emerge with a reasonable FDR.

## 6.8 Fitting ovcaRMAFromXLS with Batch

We can also see if trying to account for batch effects markedly alters the results when the XLS quantifications are used. As the scrambling is not wholly random, some of the batch structure may be preserved.

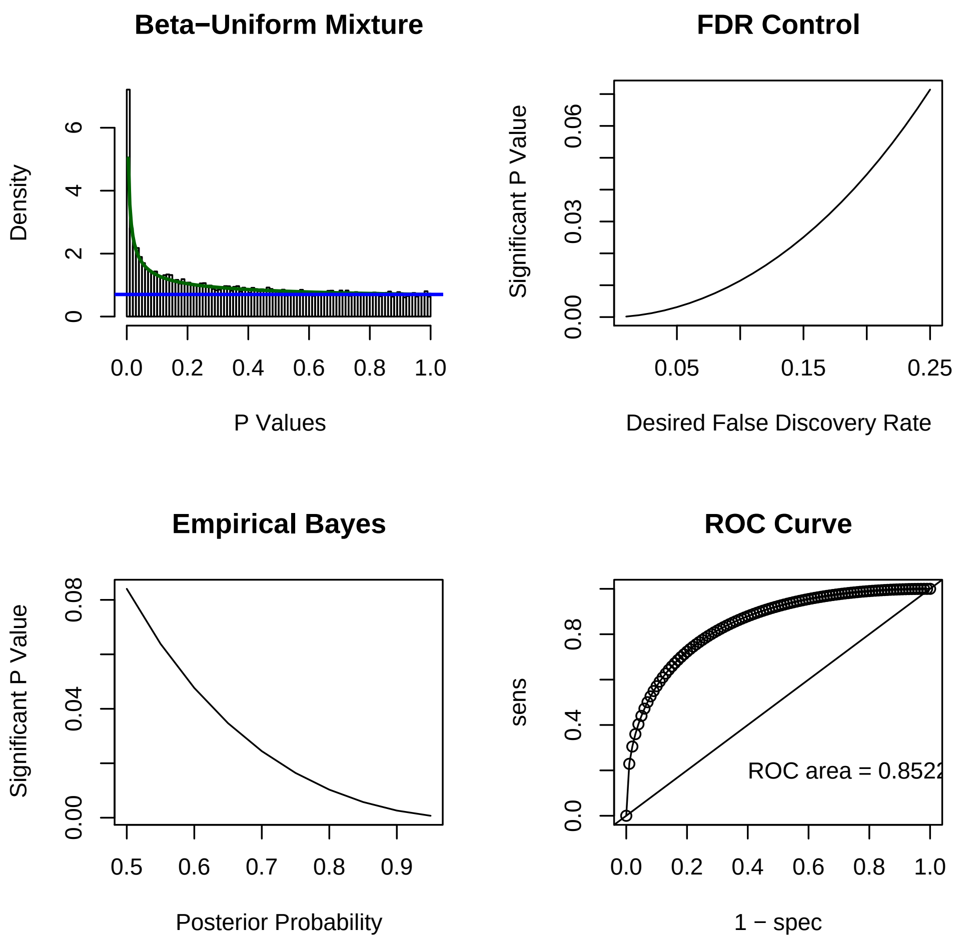
```
> batchModelXLSForm <- Y ~ runBatch
> batchModelXLSLMAll <- MultiLinearModel(batchModelForm, data.frame(runBatch = runBatch,
+     Response = clinicalInfo$Response), ovcaRMAFromXLS)
> ovcaRMAFromXLSResids <- ovcaRMAFromXLS - t(batchModelXLSLMAll@predictions)
> fullModelXLSForm <- Y ~ runBatch + Response
> fullModelXLSLMAll <- MultiLinearModel(fullModelXLSForm, data.frame(runBatch = runBatch,
+     Response = clinicalInfo$Response), ovcaRMAFromXLS)
> reducedModelXLSLMAll <- anova(fullModelXLSLMAll, batchModelXLSLMAll)
```

Having assembled the models, let's assemble the BUMs.

```
> bumXLSBatch <- Bum(batchModelXLSLMAll@p.values)
> bumRespXLSBatch <- Bum(reducedModelXLSLMAll$p.values)
```

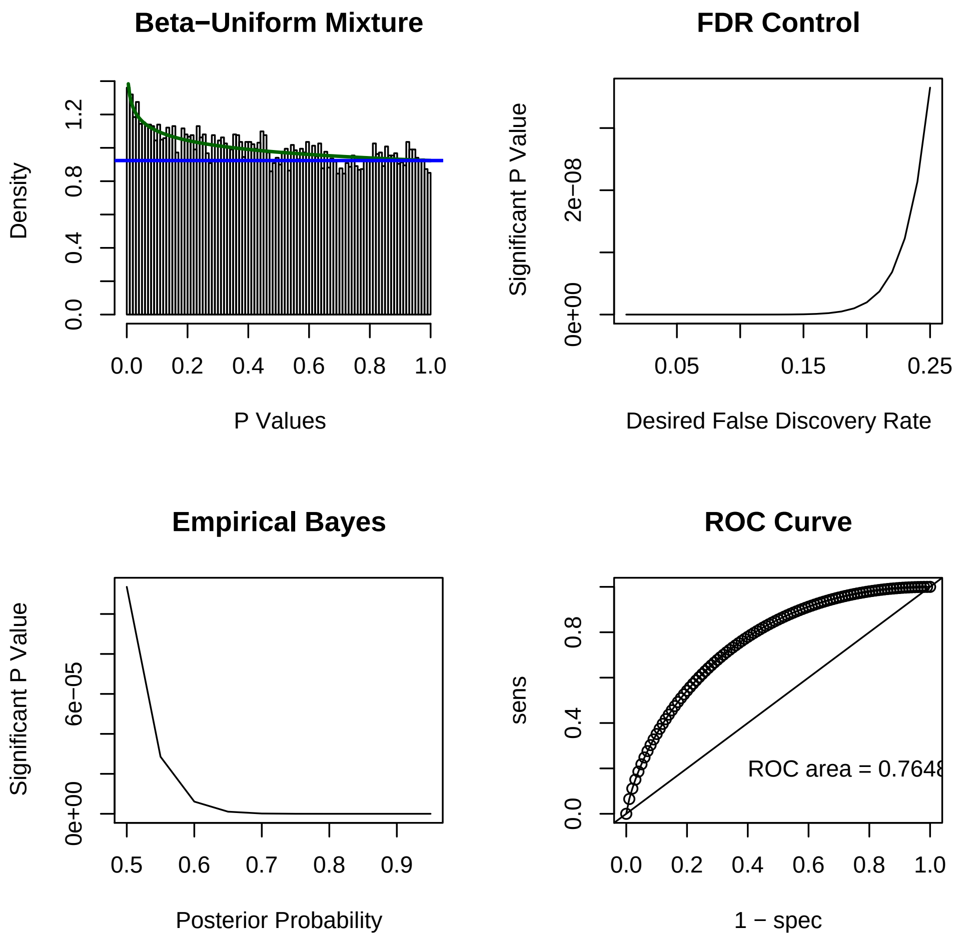
Now let's see what the plots look like.

```
> image(bumXLSBatch)
```



Even here, there are much larger effects associated with batch than with the CR/NR divide. Finally, let's look for signs of response after correcting for batch.

```
> image(bumRespXLSBatch)
```



This actually doesn't look very different from the CR/NR split before correcting for batch. However, since this is a random relabeling there may not be any actual biological enrichment.

## 7 Summary

1. When we perform two-sample t-tests comparing response (CR vs. NR) on the data computed from the CEL files, the BUM plot gives clear evidence of differential expression.
2. The same analysis applied to the data from the Excel file gives very little evidence of differential expression.
3. In both cases, the *p*-values from the reported list of 100 genes are essentially uniformly distributed.
4. Hierarchical clustering using the 100 reported genes from either the CEL data or the Excel data does not separate NRs from CRs.

5. Hierarchical clustering using genes selected directly by a t-test from the CEL data does somewhat better, but not as well as we would have predicted; it shows evidence of a larger effect that is imperfectly correlated with response.
6. When we look at the gene with the smallest p-value, it looks as though its effect on response is confounded with run date.
7. Almost every gene exhibits a strong run date effect in the CEL data. After correcting for run date effect, the differences between response categories are very slight.
8. On the Excel data, differences between response categories disappear completely after accounting for run date effects.
9. Mean expression levels of the Affymetrix controls across run dates suggest 7 run “batches”. Correcting for these run batches removes most of the differences between response categories for both the CEL and XLS data.

## 8 Appendix

### 8.1 Saves

```
> save(runBatch, file = paste("RDataObjects", "runBatch.Rda", sep = .Platform$file.sep))
> save(ovcaRMAFromCELResids, file = paste("RDataObjects", "ovcaRMAFromCELResids.Rda",
+     sep = .Platform$file.sep))
> save(ovcaRMAFromXLSResids, file = paste("RDataObjects", "ovcaRMAFromXLSResids.Rda",
+     sep = .Platform$file.sep))
```

### 8.2 SessionInfo

```
> sessionInfo()
```

```
R version 2.5.1 (2007-06-27)
i386-pc-mingw32

locale:
LC_COLLATE=English_United States.1252;LC_CTYPE=English_United States.1252;LC_MONETARY=English_United Sta

attached base packages:
[1] "splines"   "tools"     "stats"     "graphics"  "grDevices" "utils"
[7] "datasets"  "methods"   "base"

other attached packages:
       survival      colorspace  ClassDiscovery         cluster ClassComparison
         "2.32"          "0.95"         "2.5.0"        "1.11.7"         "2.5.0"
     PreProcess       oompaBase            affy          affyio         Biobase
        "2.5.0"         "2.5.0"        "1.14.2"         "1.4.1"        "1.14.1"
```