Riborex Manual

Wenzheng Li, Weili Wang, Philip J. Uren, Luiz OF Penalva, Andrew D. Smith

18 September, 2017

Introduction

Riborex is a R package for identifying differentially translated genes from Ribo-seq data. Riborex integrates both RNA- and Ribo-seq read count data into a single generalized linear model (GLM) and generates a modified design matrix reflecting the integration. At its core, Riborex applies existing RNA-seq analysis tools such as edgeR, DESeq2 and Voom to this modified design matrix and identifies differential translation across conditions.

Detailed example

First, we need to load Riborex library.

```
library(riborex)
```

The input for Riborex are two read count tables summarized from RNA-seq and Ribo-seq data respectively. The read count table should be organized as a data frame with rows correspond to genes and columns correspond to samples as shown below.

```
data(riborexdata)
RNACntTable <- rna
RiboCntTable <- ribo
```

We can check the first five lines of the table:

```
head(RNACntTable,5)
```

```
##                     BN_336 BN_337 BN_338 BN_339
## ENSRNOG00000000017      7     11      4      4
## ENSRNOG00000000024   2467   2478   3258   2316
## ENSRNOG00000000033    206    282    330    244
## ENSRNOG00000000034    758    672   1335    767
## ENSRNOG00000000036    237    163    211    189
```

```
head(RiboCntTable,5)
```

```
##                     BN_341 BN_342 BN_343 BN_344
## ENSRNOG00000000017     15      5     10      2
## ENSRNOG00000000024   5206   5921   2864   1985
## ENSRNOG00000000033     30     30     23     13
## ENSRNOG00000000034    943    775    842    311
## ENSRNOG00000000036     80     49     30      7
```

Then we need to prepare two vectors to indicate the treatments of samples in RNA- and Ribo-seq data. Both RNA-seq and Ribo-seq can have different number of samples in control and treated condtions, and RNA-seq and Ribo-seq data can have different number of samples.

```
rnaCond <- c("control", "control", "treated", "treated")
riboCond <- c("control", "control", "treated", "treated")
```

After the two read count table and two condition vectors are ready, we can use riborex (), and we can choose which engine to use. By default, DESeq2 is used as the engine if you don't specify the engine option. Use help(riborex) in R to see more details about this function.

```
res.deseq2 <- riborex(RNACntTable, RiboCntTable, rnaCond, riboCond)
```

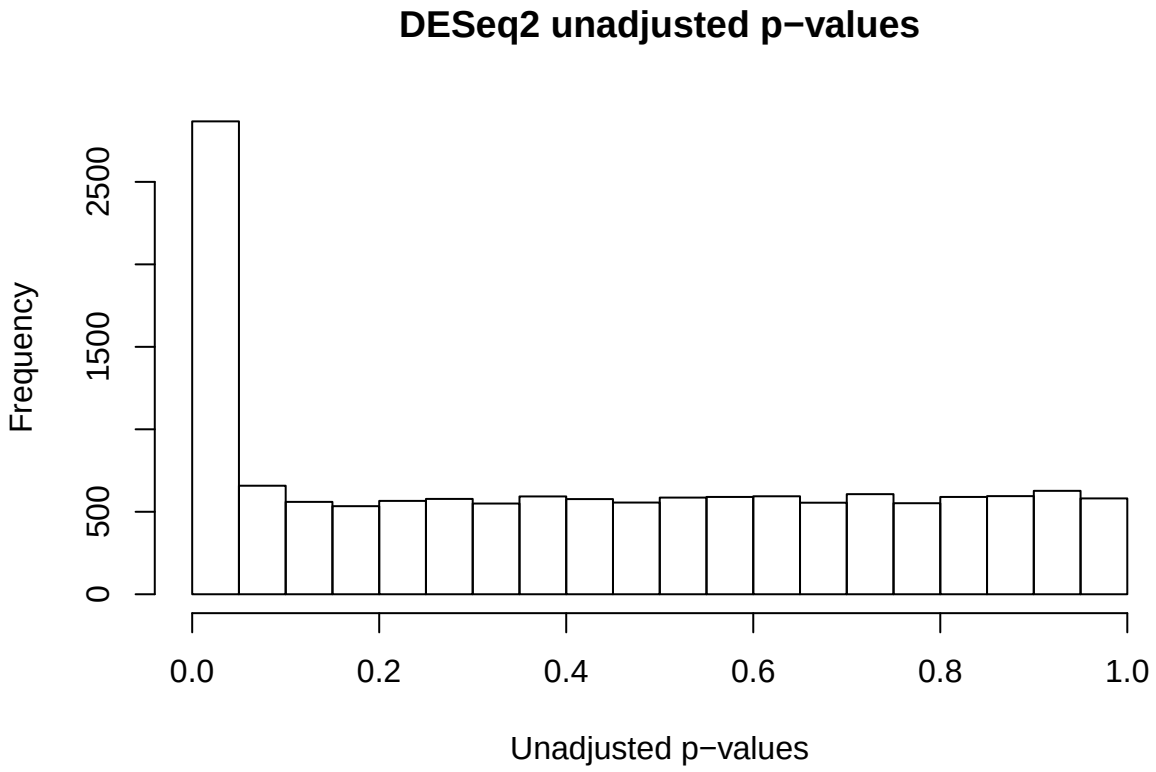
The format of the result is the same when DESeq2 is used in RNA-seq analysis.

```
res.deseq2
```

```
## log2 fold change (MLE): EXTRA1 treated vs control
## Wald test p-value: EXTRA1 treated vs control
## DataFrame with 13916 rows and 6 columns
##                       baseMean log2FoldChange     lfcSE       stat
##                      <numeric>      <numeric> <numeric>  <numeric>
## ENSRNOG00000000017    7.694934     1.53323724 1.5662846  0.9789008
## ENSRNOG00000000024 3544.238071    -0.10215095 0.2020511 -0.5055699
## ENSRNOG00000000033  111.439451     0.26525296 0.5033680  0.5269564
## ENSRNOG00000000034  783.974916     0.06513995 0.3506098  0.1857904
## ENSRNOG00000000036   99.151141    -0.63425877 0.5910827 -1.0730457
## ...                        ...            ...       ...        ...
## ENSRNOG00000061895   105.24110     0.52804379 0.4846633  1.0895065
## ENSRNOG00000061899    40.04057    -0.49721003 0.7802896 -0.6372122
## ENSRNOG00000061910  4237.53481     0.30904740 0.2460783  1.2558905
## ENSRNOG00000061928  1651.61680    -0.08099135 0.2465675 -0.3284753
## ENSRNOG00000061989   107.36281    -0.10482910 0.4554566 -0.2301627
##                        pvalue      padj
##                     <numeric> <numeric>
## ENSRNOG00000000017 0.3276290 0.7366558
## ENSRNOG00000000024 0.6131586 0.9019205
## ENSRNOG00000000033 0.5982239 0.8963865
## ENSRNOG00000000034 0.8526091 0.9748558
## ENSRNOG00000000036 0.2832506 0.6901202
## ...                      ...       ...
## ENSRNOG00000061895 0.2759306 0.6824115
## ENSRNOG00000061899 0.5239866 0.8635147
## ENSRNOG00000061910 0.2091557 0.5984207
## ENSRNOG00000061928 0.7425523 0.9428604
## ENSRNOG00000061989 0.8179654 0.9676247
```

You can check the distribution p-values.

```
hist(res.deseq2$pvalue, main = 'DESeq2 unadjusted p-values', xlab='Unadjusted p-values')
```



We can see for this dataset, the p-value distribution is as expected based on DESeq2 manual which is uniformly distribution with differentially expressed genes enriched with small p-values. We will show another dataset later for which the p-value distribution is skew to the right and how it can be fixed with fdrtool.

Also, you can use summary () for your results.

```
summary(res.deseq2)
```

```
##
## out of 13916 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)     : 1107, 8%
## LFC < 0 (down)   : 1217, 8.7%
## outliers [1]     : 0, 0%
## low counts [2]   : 540, 3.9%
## (mean count < 6)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

And results can be saved by:

```
write.table(res.deseq2, "riborex_res_deseq2.txt", quote=FALSE)
```

If you want to use edgeR as your engine, you can use riborex () as:

```
res.edgeR <- riborex(RNACntTable, RiboCntTable, rnaCond, riboCond, "edgeR")
```

The format of the result is the same when edgeR is used in RNA-seq analysis.

```
head(res.edgeR$table)
```

```
##                          logFC    logCPM         LR    PValue       FDR
## ENSRNOG00000000017  1.36290149 -2.456968 1.80649827 0.1789289 0.4627345
```

```
## ENSRNOG00000000024 -0.30127172  6.212404 2.13670654 0.1438103 0.4037250
## ENSRNOG00000000033  0.07178854  1.313235 0.02631769 0.8711269 0.9636097
## ENSRNOG00000000034 -0.13430329  4.029136 0.12541035 0.7232390 0.9111612
## ENSRNOG00000000036 -0.82540899  1.132478 2.15554356 0.1420562 0.4008219
## ENSRNOG00000000040 -0.19057283 -1.555003 0.07537378 0.7836675 0.9338658
```

For edgeR engine, you can also choose to estimate dispersion of RNA-seq and Ribo-seq data separately by specifying engine as "edgeRD".

```
res.edgeRD <- riborex(RNACntTable, RiboCntTable, rnaCond, riboCond, "edgeRD")
```

If you want to use Voom as the engine, you can run riborex () as:

```
res.voom <- riborex(RNACntTable, RiboCntTable, rnaCond, riboCond, "Voom")
```

The format of the result is the same when Voom is used in RNA-seq analysis.

```
head(res.voom)
```

```
##                          logFC    AveExpr          t   P.Value adj.P.Val
## ENSRNOG00000000017  1.39674189 -2.8437969  1.2847048 0.2268559 0.5243467
## ENSRNOG00000000024 -0.30047073  6.0075571 -1.8737990 0.0893910 0.2991861
## ENSRNOG00000000033  0.07661457  0.7070877  0.1687028 0.8692764 0.9540355
## ENSRNOG00000000034 -0.13777833  3.9701893 -0.3933633 0.7020190 0.8871606
## ENSRNOG00000000036 -0.89165405  0.7106061 -1.3890222 0.1939374 0.4826701
## ENSRNOG00000000040 -0.20967655 -1.7042630 -0.2991046 0.7707701 0.9157225
##                            B
## ENSRNOG00000000017 -4.928058
## ENSRNOG00000000024 -5.682138
## ENSRNOG00000000033 -6.320171
## ENSRNOG00000000034 -7.130133
## ENSRNOG00000000036 -5.425369
## ENSRNOG00000000040 -5.854482
```

Case-study with "incorrect" p-value distribution

```
RNACntTable.corrected <- rna.null
RiboCntTable.corrected <- ribo.null
```

We can check the first five lines of the table:

```
head(RNACntTable.corrected)
```

```
##                    rna_T0_r1 rna_T0_r2 rna_T0_r3 rna_T24_r1 rna_T24_r2
## ENSG00000116032.5          0         0         0          0          2
## ENSG00000188026.12       803       691       763       1118        973
## ENSG00000171174.13       198       149       236        227        193
## ENSG00000166257.8          0         0         0          0          0
## ENSG00000149136.8       9301      7705     12490       4795       4842
## ENSG00000136938.8       6325      5433      8880       3163       4074
##                    rna_T24_r3
## ENSG00000116032.5           0
## ENSG00000188026.12        656
## ENSG00000171174.13        198
## ENSG00000166257.8           0
## ENSG00000149136.8        4151
```

```
## ENSG00000136938.8        2996
```

```
head(RiboCntTable.corrected)
```

```
##                     ribo_T0_r1 ribo_T0_r2 ribo_T0_r3 ribo_T24_r1
## ENSG00000116032.5            2          0          0           2
## ENSG00000188026.12         382        432        360         784
## ENSG00000171174.13          75        113         77         161
## ENSG00000166257.8            0          0          0           0
## ENSG00000149136.8         2536       3546       2702        2743
## ENSG00000136938.8          892       1473       1060         979
##                     ribo_T24_r2 ribo_T24_r3
## ENSG00000116032.5             2           5
## ENSG00000188026.12          880         890
## ENSG00000171174.13          220         164
## ENSG00000166257.8             3           2
## ENSG00000149136.8          3678        2765
## ENSG00000136938.8          1473         947
```

The condition vectors can be created as:

```
rnaCond.corrected <- c(rep('T0', 3), rep('T24',3))
riboCond.corrected <- rnaCond.corrected
rnaCond.corrected
```

```
## [1] "T0"  "T0"  "T0"  "T24" "T24" "T24"
```

```
riboCond.corrected
```

```
## [1] "T0"  "T0"  "T0"  "T24" "T24" "T24"
```

The results from DESeq2 can be obtained as:

```
res.deseq2.corrected <- riborex(RNACntTable.corrected, RiboCntTable.corrected, rnaCond.corrected, riboC
```

```
## DESeq2 mode selected
```

```
## combining design matrix
```

```
## applying DESeq2 to modified design matrix
```

```
## estimating size factors
```

```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

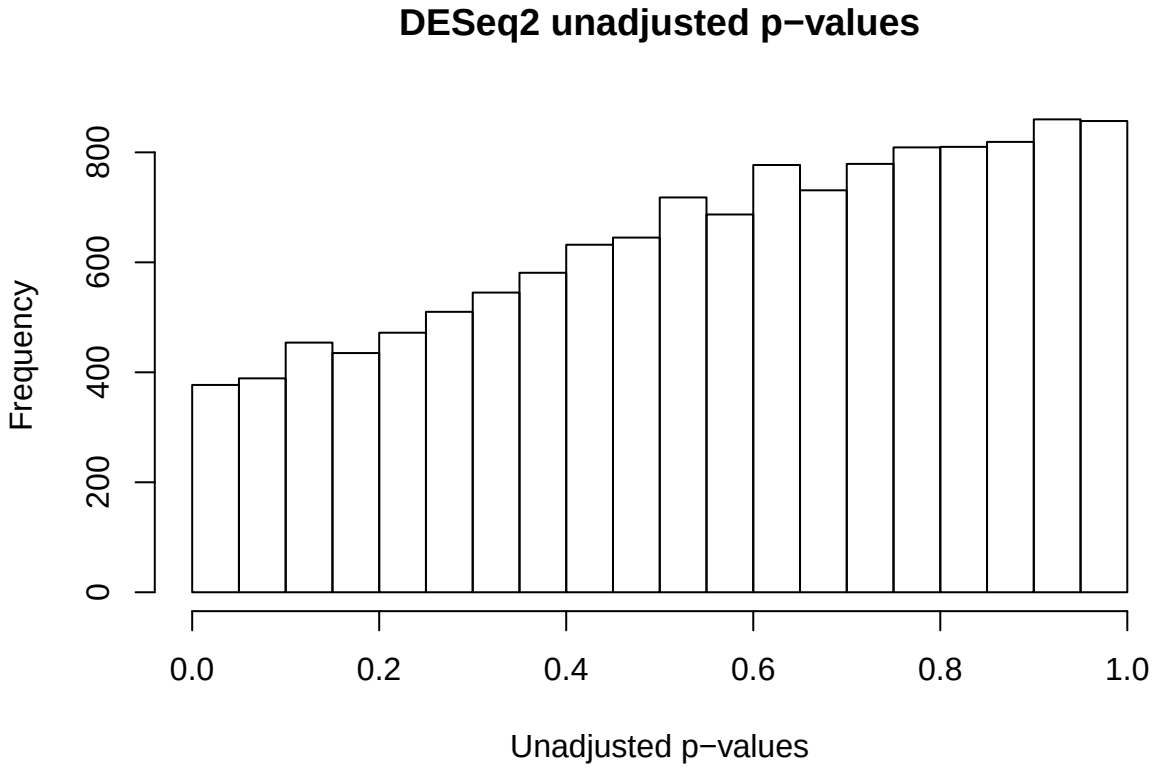
```
## mean-dispersion relationship
```

```
## final dispersion estimates
```

```
## fitting model and testing
```

We can check the p-value distribution as:

```
hist(res.deseq2.corrected$pvalue, main = 'DESeq2 unadjusted p-values', xlab='Unadjusted p-values')
```



We can see from the histogram that the distribution of p-values is skew to the right, that means the null distribution is not "correct", we can fix it by reestimating the p-values using fdrtool:

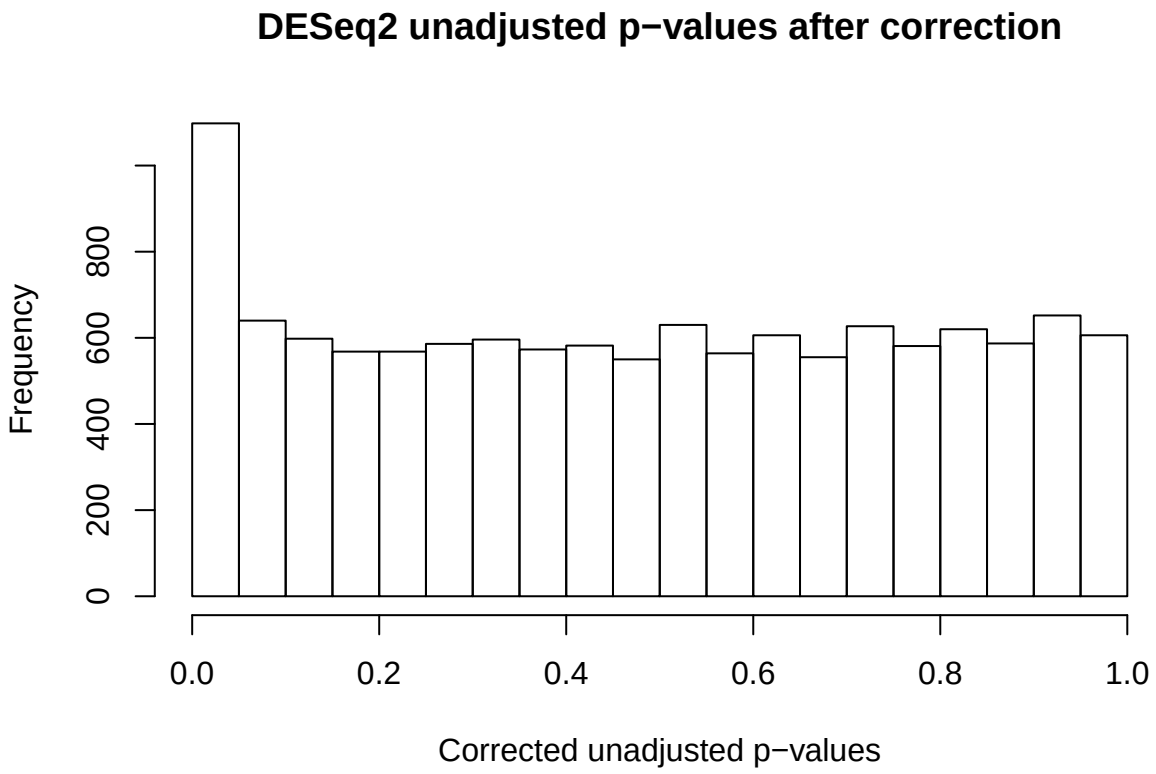
```
results.corrected <- correctNullDistribution(res.deseq2.corrected)
```

```
## correcting null distribution by reestimating pvalues
```

```
## Step 1... determine cutoff point
## Step 2... estimate parameters of null distribution and eta0
## Step 3... compute p-values and estimate empirical PDF/CDF
## Step 4... compute q-values and local fdr
```

We can see the p-value distribution after correction:

```
hist(results.corrected$pvalue, main = 'DESeq2 unadjusted p-values after correction',
     xlab='Corrected unadjusted p-values')
```



We can see after the correction, the distribution of p-values is as expected. And the adjusted pvalues are corrected also.

Multi-factor experiment

Since we don't find any available ribosome profiling data generated in a multi-factor experiement, here we generate a pseudo dataset to demonstrate the usage of riborex in a multi-factor experiment. The pseudo dataset have 8 samples in RNA-seq and Ribo-seq, and two factors are included.

```
rna <- RNACntTable[,c(1,2,3,4,1,2,3,4)]
ribo <- RiboCntTable[,c(1,2,3,4,1,2,3,4)]
```

For multi-factor experiment, we prepare two data frames to indicate the treatment under each factor. Here for the 8 samples in both RNA- and Ribo-seq experiement, the 3rd and 4th samples are treated with drug1 and the 7th and 8th samples are treated with drug2.

```
rnaCond <- data.frame(factor1=(c("control1", "control1", "treated1", "treated1",
                                 "control1", "control1", "control1", "control1")),
                      factor2=(c("control2", "control2", "control2", "control2",
                                 "control2", "control2", "treated2", "treated2")))

riboCond <- data.frame(factor1=(c("control1", "control1", "treated1", "treated1",
                                  "control1", "control1", "control1", "control1")),
                       factor2=(c("control2", "control2", "control2", "control2",
                                  "control2", "control2", "treated2", "treated2")))
```

Also we need to prepare a contrast to specify the comparison we want to perform, for example, if we want to compare the influence of the usage of drug2. The contrast can be constructed as:

```
contrast = c("factor2", "control2", "treated2")
```

Then riborex () is used with contrast specified.

```
res.deseq2 <- riborex(rna, ribo, rnaCond, riboCond, "DESeq2", contrast = contrast)
```

We can see the summary of the result:

```
summary(res.deseq2)
```

```
##
## out of 13916 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)     : 1887, 14%
## LFC < 0 (down)   : 1987, 14%
## outliers [1]     : 0, 0%
## low counts [2]   : 270, 1.9%
## (mean count < 3)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

edgeR and edgeRD can be used in a similar way.

```
res.edgeR <- riborex(rna, ribo, rnaCond, riboCond, "edgeR", contrast = contrast)
```

```
res.edgeRD <- riborex(rna, ribo, rnaCond, riboCond, "edgeRD", contrast = contrast)
```

Currently, you can't choose Voom as the engine in a multi-factor experiment yet.

Setup

This analysis was conducted on

```
sessionInfo()
```

```
## R version 3.3.2 (2016-10-31)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 17.04
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] riborex_2.3.4              fdrtool_1.2.15
##  [3] edgeR_3.16.5               limma_3.30.13
##  [5] DESeq2_1.16.1              SummarizedExperiment_1.4.0
##  [7] Biobase_2.34.0             GenomicRanges_1.26.4
```

```
##  [9] GenomeInfoDb_1.10.3        IRanges_2.8.2
## [11] S4Vectors_0.12.2           BiocGenerics_0.22.0
##
## loaded via a namespace (and not attached):
##  [1] genefilter_1.58.1    locfit_1.5-9.1       splines_3.3.2
##  [4] lattice_0.20-35      colorspace_1.3-2     htmltools_0.3.6
##  [7] yaml_2.1.14          base64enc_0.1-3      blob_1.1.0
## [10] survival_2.41-3      XML_3.98-1.6         rlang_0.1.2
## [13] foreign_0.8-69       DBI_0.7              BiocParallel_1.8.2
## [16] bit64_0.9-7          RColorBrewer_1.1-2   plyr_1.8.4
## [19] stringr_1.2.0        zlibbioc_1.20.0      munsell_0.4.3
## [22] gtable_0.2.0         htmlwidgets_0.9      memoise_1.1.0
## [25] evaluate_0.10.1      latticeExtra_0.6-28  knitr_1.17
## [28] geneplotter_1.52.0   AnnotationDbi_1.38.0 htmlTable_1.9
## [31] Rcpp_0.12.12         acepack_1.4.1        xtable_1.8-2
## [34] scales_0.5.0         backports_1.1.0      checkmate_1.8.3
## [37] Hmisc_4.0-3          annotate_1.52.1      XVector_0.14.1
## [40] bit_1.1-12           gridExtra_2.3        ggplot2_2.2.1
## [43] digest_0.6.12        stringi_1.1.5        grid_3.3.2
## [46] rprojroot_1.2        tools_3.3.2          bitops_1.0-6
## [49] magrittr_1.5         lazyeval_0.2.0       RCurl_1.95-4.8
## [52] tibble_1.3.4         RSQLite_2.0          Formula_1.2-2
## [55] cluster_2.0.6        Matrix_1.2-11        data.table_1.10.4
## [58] rmarkdown_1.6        rpart_4.1-11         nnet_7.3-12
```