recount (gene and exon analyses)

Kai Kammers and Shannon Ellis

July 22, 2016

Contents

Included is an example of how to download and analyze expression data from SRA study SRP032798. The data come from human breast cancer samples, and we compare the transcriptomes of TNBC samples (triple negative breast cancer) and HER2-positive breast cancer samples (breast cancer type that tests positive for a protein called human epidermal growth factor receptor 2). Code here demonstrates how to carry out differential expression analyses on gene, exon, junction, and differential expressed region (DER) levels within a single study using `limma` and `voom`. We test for concordance among the results of each analysis and demonstrate how to carry out gene ontology analysis using `topGO` to characterize top hits from differential expression analyses.

We first load the required packages.

```
## load libraries
library('recount')
```

```
##
## This data.table install has not detected OpenMP support. It will work but slower in single threaded r
```

```
library('SummarizedExperiment')
library('limma')
library('edgeR')
library('qvalue')
library('topGO')
library('matrixStats')
library('RSkittleBrewer')
library('derfinder')
```

Gene level analysis

We first download the project of interest (SRP032798), obtaining expression data for the study of interest. We obtain summaries of the number of samples and genes included using `colData()` and `rowData()`, respectively.

```
## Find the project of interest (SRP032789), e.g. with parts of the abstract
project_info <- abstract_search('To define the digital transcriptome of three breast cancer')
```

```
## Explore information
project_info
```

```
##     number_samples species
## 865             20   human
##
## 865 Goal: To define the digital transcriptome of three breast cancer subtypes (TNBC, Non-TNBC, and HI
##       project
## 865 SRP032789
```

```
## Browse the project at SRA
browse_study(project_info$project)

## Download the gene level RangedSummarizedExperiment data
if(!file.exists(file.path('SRP032789', 'rse_gene.Rdata'))) {
    download_study(project_info$project)
}

## Load the data
load(file.path(project_info$project, 'rse_gene.Rdata'))
rse_gene
```

```
## class: RangedSummarizedExperiment
## dim: 23779 20
## metadata(0):
## assays(1): counts
## rownames(23779): 1 10 ... 9994 9997
## rowData names(2): gene_id bp_length
## colnames(20): SRR1027171 SRR1027173 ... SRR1027190 SRR1027172
## colData names(18): project sample ... avg_read_length bigwig_file
```

```
## This is the phenotype data provided by the recount project
colData(rse_gene)
```

```
## DataFrame with 20 rows and 18 columns
##                project      sample  experiment         run
##            <character> <character> <character> <character>
## SRR1027171   SRP032789   SRS500214   SRX374850  SRR1027171
## SRR1027173   SRP032789   SRS500216   SRX374852  SRR1027173
## SRR1027174   SRP032789   SRS500217   SRX374853  SRR1027174
## SRR1027175   SRP032789   SRS500218   SRX374854  SRR1027175
## SRR1027176   SRP032789   SRS500219   SRX374855  SRR1027176
## ...                ...         ...         ...         ...
## SRR1027187   SRP032789   SRS500230   SRX374866  SRR1027187
## SRR1027188   SRP032789   SRS500231   SRX374867  SRR1027188
## SRR1027189   SRP032789   SRS500232   SRX374868  SRR1027189
## SRR1027190   SRP032789   SRS500233   SRX374869  SRR1027190
## SRR1027172   SRP032789   SRS500215   SRX374851  SRR1027172
##            read_count_as_reported_by_sra reads_aligned
##                                <integer>     <integer>
## SRR1027171                      88869444      88869444
## SRR1027173                     107812596     107812596
## SRR1027174                      98563260      98563260
## SRR1027175                      91327892      91327892
## SRR1027176                      96513572      96513572
```

```
## ...                                ...          ...
## SRR1027187                     75260678      75260678
## SRR1027188                     65709192      65709192
## SRR1027189                     65801392      65801392
## SRR1027190                     74356276      74356276
## SRR1027172                     80986440      58902122
##            proportion_of_reads_reported_by_sra_aligned paired_end
##                                              <numeric>  <logical>
## SRR1027171                                           1       TRUE
## SRR1027173                                           1       TRUE
## SRR1027174                                           1       TRUE
## SRR1027175                                           1       TRUE
## SRR1027176                                           1       TRUE
## ...                                                ...        ...
## SRR1027187                                   1.0000000       TRUE
## SRR1027188                                   1.0000000       TRUE
## SRR1027189                                   1.0000000       TRUE
## SRR1027190                                   1.0000000       TRUE
## SRR1027172                                   0.7273084       TRUE
##            sra_misreported_paired_end mapped_read_count        auc
##                             <logical>         <integer>  <numeric>
## SRR1027171                      FALSE          86949307 5082692127
## SRR1027173                      FALSE         104337779 6077034329
## SRR1027174                      FALSE          95271238 5504462845
## SRR1027175                      FALSE          88820239 5150234117
## SRR1027176                      FALSE          93464650 5416681912
## ...                               ...               ...        ...
## SRR1027187                      FALSE          64697612 3567078255
## SRR1027188                      FALSE          65278500 4856453823
## SRR1027189                      FALSE          65328289 4858587600
## SRR1027190                      FALSE          73911898 5501089036
## SRR1027172                      FALSE          57523391 3351013968
##            sharq_tissue sharq_cell_type biosample_submission_date
##             <character>     <character>                <character>
## SRR1027171       breast             esc   2013-11-07T12:40:22.203
## SRR1027173       breast             esc   2013-11-07T12:40:32.283
## SRR1027174       breast             esc   2013-11-07T12:40:28.283
## SRR1027175       breast             esc   2013-11-07T12:40:34.343
## SRR1027176       breast             esc   2013-11-07T12:40:36.303
## ...                 ...             ...                       ...
## SRR1027187       breast             esc   2013-11-07T12:40:56.180
## SRR1027188       breast             esc   2013-11-07T12:40:58.170
## SRR1027189       breast             esc   2013-11-07T12:40:20.227
## SRR1027190       breast             esc   2013-11-07T12:40:18.090
## SRR1027172       breast             esc   2013-11-07T12:40:26.217
##            biosample_publication_date   biosample_update_date
##                           <character>             <character>
## SRR1027171    2013-11-08T01:11:17.160 2014-03-07T16:09:38.542
## SRR1027173    2013-11-08T01:11:14.827 2014-03-07T16:09:38.698
## SRR1027174    2013-11-08T01:11:52.283 2014-03-07T16:09:38.637
## SRR1027175    2013-11-08T01:11:15.963 2014-03-07T16:09:38.731
## SRR1027176    2013-11-08T01:11:46.430 2014-03-07T16:09:38.768
## ...                               ...                     ...
## SRR1027187    2013-11-08T01:11:29.587 2014-03-07T16:09:39.093
```

```
## SRR1027188    2013-11-08T01:12:06.660 2014-03-07T16:09:39.130
## SRR1027189    2013-11-08T01:11:33.080 2014-03-07T16:09:38.498
## SRR1027190    2013-11-08T01:12:11.320 2014-03-07T16:09:38.469
## SRR1027172    2013-11-08T01:11:45.250 2014-03-07T16:09:38.604
##            avg_read_length   bigwig_file
##                  <integer>   <character>
## SRR1027171             120 SRR1027171.bw
## SRR1027173             120 SRR1027173.bw
## SRR1027174             120 SRR1027174.bw
## SRR1027175             120 SRR1027175.bw
## SRR1027176             120 SRR1027176.bw
## ...                    ...           ...
## SRR1027187             120 SRR1027187.bw
## SRR1027188             150 SRR1027188.bw
## SRR1027189             150 SRR1027189.bw
## SRR1027190             150 SRR1027190.bw
## SRR1027172              87 SRR1027172.bw
```

```
## At the gene level, the row data includes the names of the genes and
## the sum of the reduced exons widths, which can be used for taking into
## account the gene length.
rowData(rse_gene)
```

```
## DataFrame with 23779 rows and 2 columns
##           gene_id bp_length
##       <character> <integer>
## 1               1      4027
## 2              10      1317
## 3             100      1532
## 4            1000      4473
## 5       100008589      5071
## ...           ...       ...
## 23775        9991      8234
## 23776        9992       803
## 23777        9993      4882
## 23778        9994      6763
## 23779        9997      1393
```

Downloaded count data are first scaled to take into account differing coverage between samples. Phenotype data (`pheno`) are obtained and ordered to match the sample order of the gene expression data (`rse_gene`). Only those samples that are HER2-positive or TNBC are included for analysis. Prior to differential gene expression analysis, count data are obtained in matrix format and then filtered to only include those genes with greater than five average normalized counts across all samples.

```
## Scale counts by taking into account the total coverage per sample
rse <- scale_counts(rse_gene)

## Download additional phenotype data from
## http://trace.ncbi.nlm.nih.gov/Traces/study/?acc=SRP032789
pheno <- read.table('SraRunTable_SRP032789.txt', sep = '\t',
    header = TRUE,
    stringsAsFactors = FALSE)

## Obtain correct order for pheno data
pheno <- pheno[match(rse$run, pheno$Run_s), ]
identical(pheno$Run_s, rse$run)
```

```
## [1] TRUE
```

```
head(cbind(pheno$Run_s, rse$run))
```

```
##      [,1]         [,2]
## [1,] "SRR1027171" "SRR1027171"
## [2,] "SRR1027173" "SRR1027173"
## [3,] "SRR1027174" "SRR1027174"
## [4,] "SRR1027175" "SRR1027175"
## [5,] "SRR1027176" "SRR1027176"
## [6,] "SRR1027177" "SRR1027177"
```

```
## Obtain grouping information
colData(rse)$group <- pheno$tumor_type_s
table(colData(rse)$group)
```

```
##
## HER2 Positive Breast Tumor      Non-TNBC Breast Tumor
##                          5                          6
##    Normal Breast Organoids          TNBC Breast Tumor
##                          3                          6
```

```
## Subset data to HER2 and TNBC types
rse <- rse[, rse$group %in% c('HER2 Positive Breast Tumor',
                              'TNBC Breast Tumor')]

## Save filtered rse object
rse_gene_filt <- rse

## Obtain count matrix
counts <- assays(rse_gene_filt)$counts

## Filter count matrix
filter <- apply(counts, 1, function(x) mean(x) > 5)
counts <- counts[filter, ]
dim(counts)
```

```
## [1] 17874    11
```

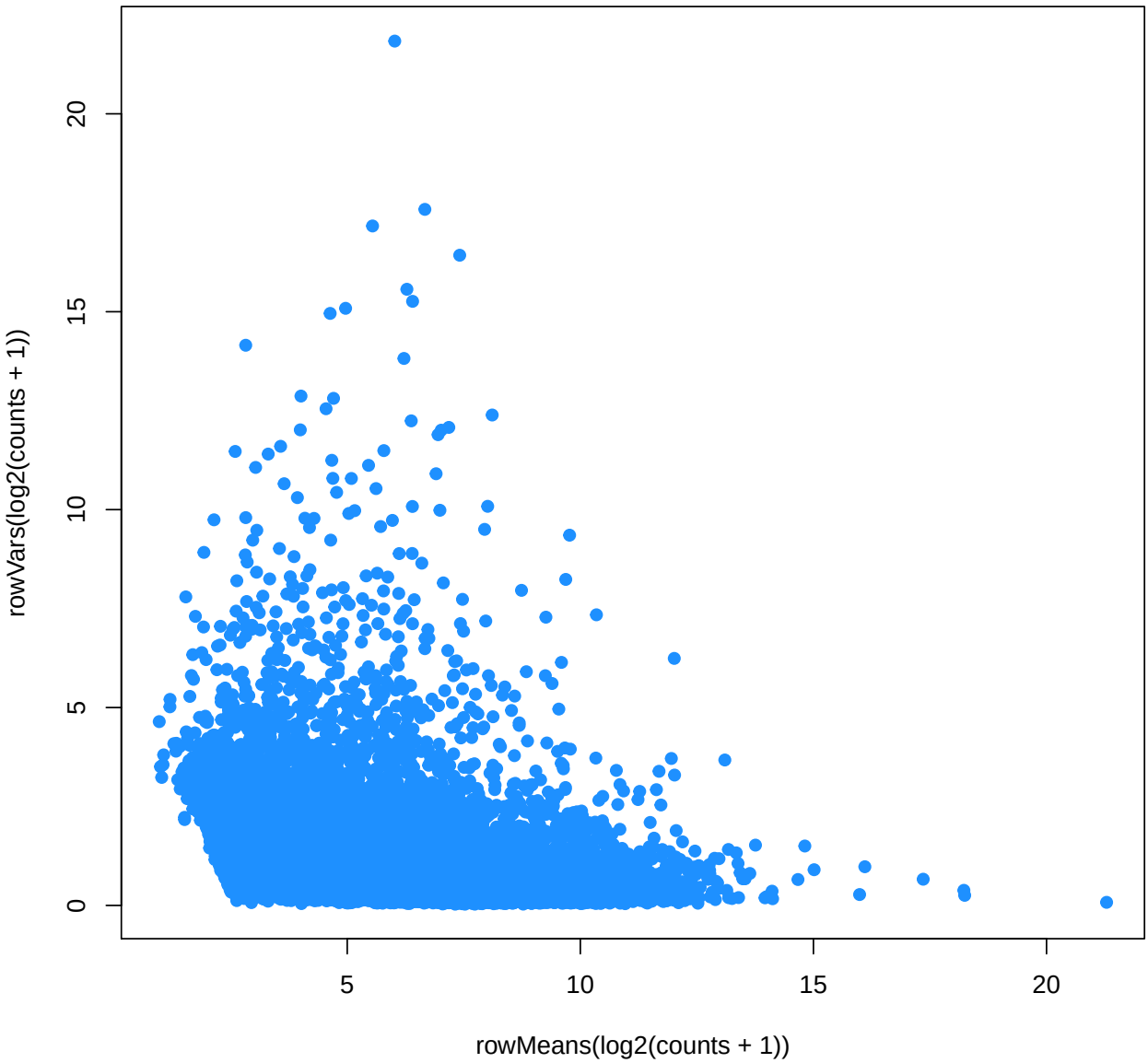
```
## Save for gene, exon and junction comparisons
counts_gene <- counts
counts_gene[1:5, 1:5]
```

```
##           SRR1027171 SRR1027173 SRR1027174 SRR1027175 SRR1027176
## 1                 66         46         42         64         65
## 100               97         22         73         90        109
## 1000              29        183        450        160        153
## 100008589    1842832    2085085    3122352    3082593    2862316
## 100009676         92         50        123        119         99
```

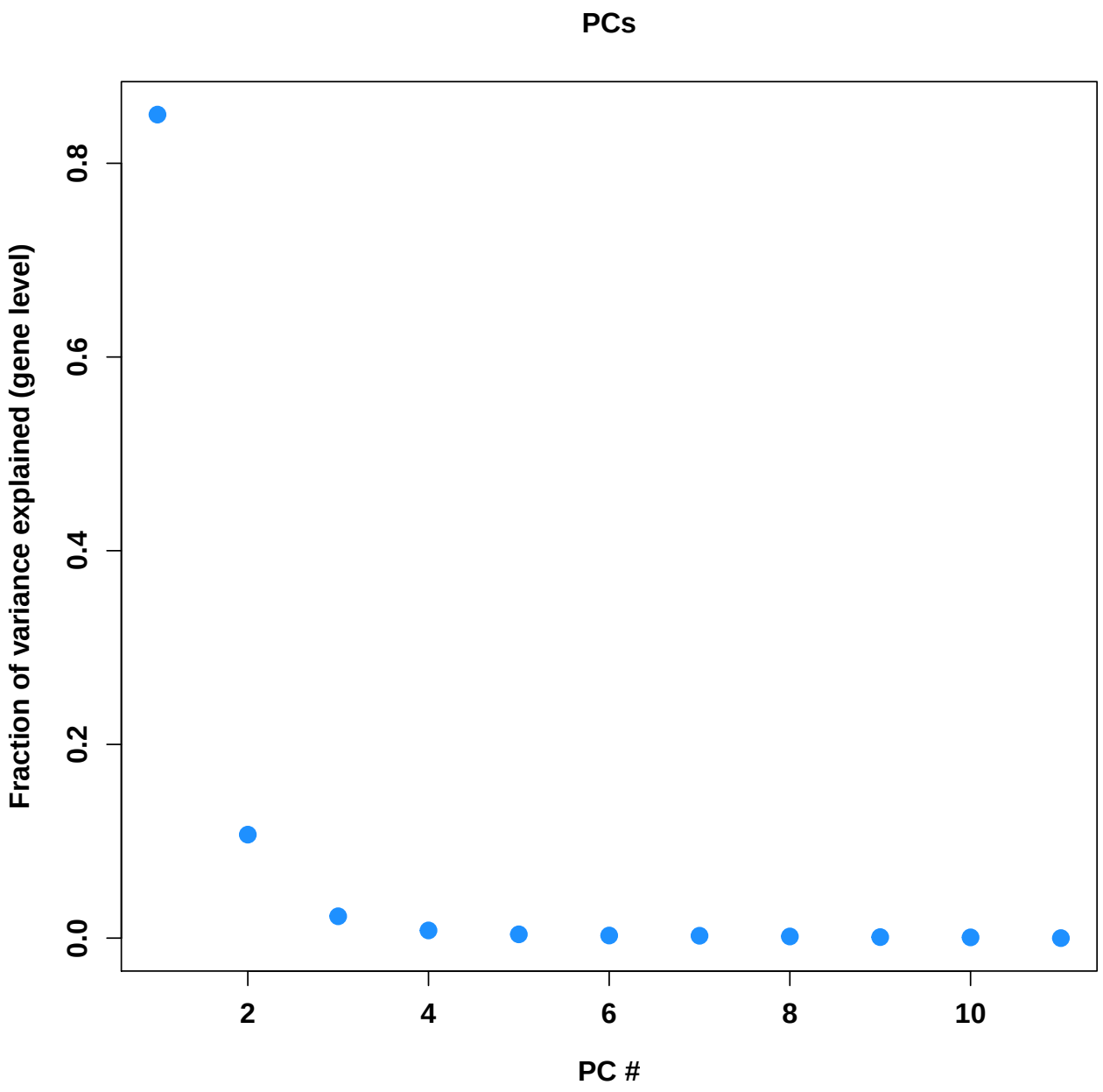
To get a better sense of the data, we plot the mean-variance relationship for each gene. Similarly, we run principal component analysis (PCA) to identify any sample outliers within the data. We assess the variance explained by each of the first 11 PCs as well as visualize the relationship of each sample in the first two PCs.

```
## Set colors
trop <- RSkittleBrewer('tropical')[c(1, 2)]
cols <- as.numeric(as.factor(rse$group))
```

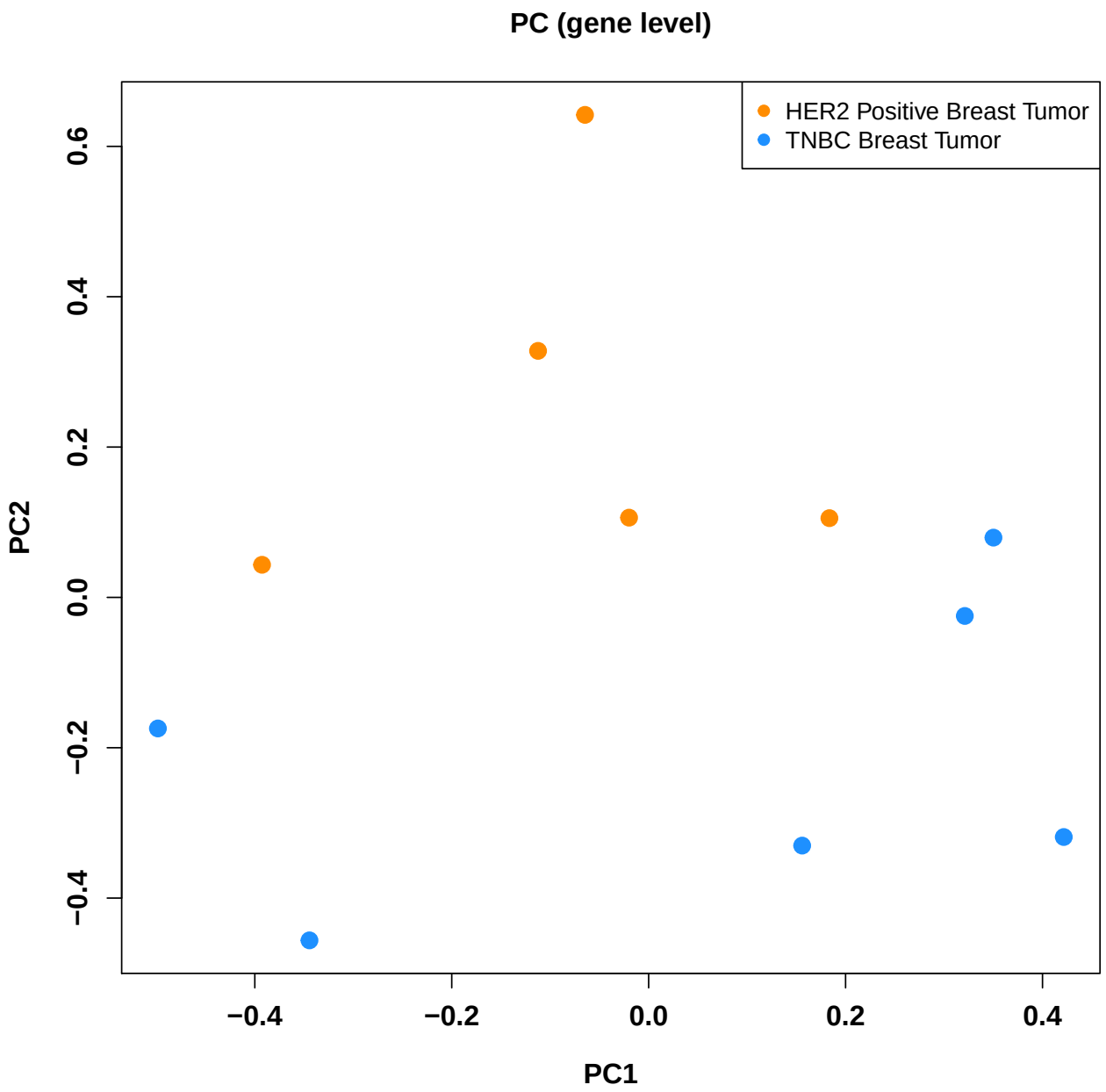
```
## Look at mean variance relationship
plot(rowMeans(log2(counts + 1)), rowVars(log2(counts + 1)),
     pch = 19, col = trop[2])
```




```
## Calculate PCs with svd function
expr.pca <- svd(counts - rowMeans(counts))

## Plot PCs
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
plot(expr.pca$d^2 / sum(expr.pca$d^2), pch = 19, col = trop[2], cex = 1.5,
     ylab = 'Fraction of variance explained (gene level)', xlab = 'PC #',
     main = 'PCs')
```




```
## Plot PC1 vs. PC2
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
plot(expr.pca$v[, 1], expr.pca$v[, 2], pch = 19, col = trop[cols], cex = 1.5,
     xlab = 'PC1', ylab = 'PC2',
     main = 'PC (gene level)')
legend('topright', pch = 19, col = trop[c(1, 2)],
       names(summary(as.factor(rse$group))), bg="white")
```



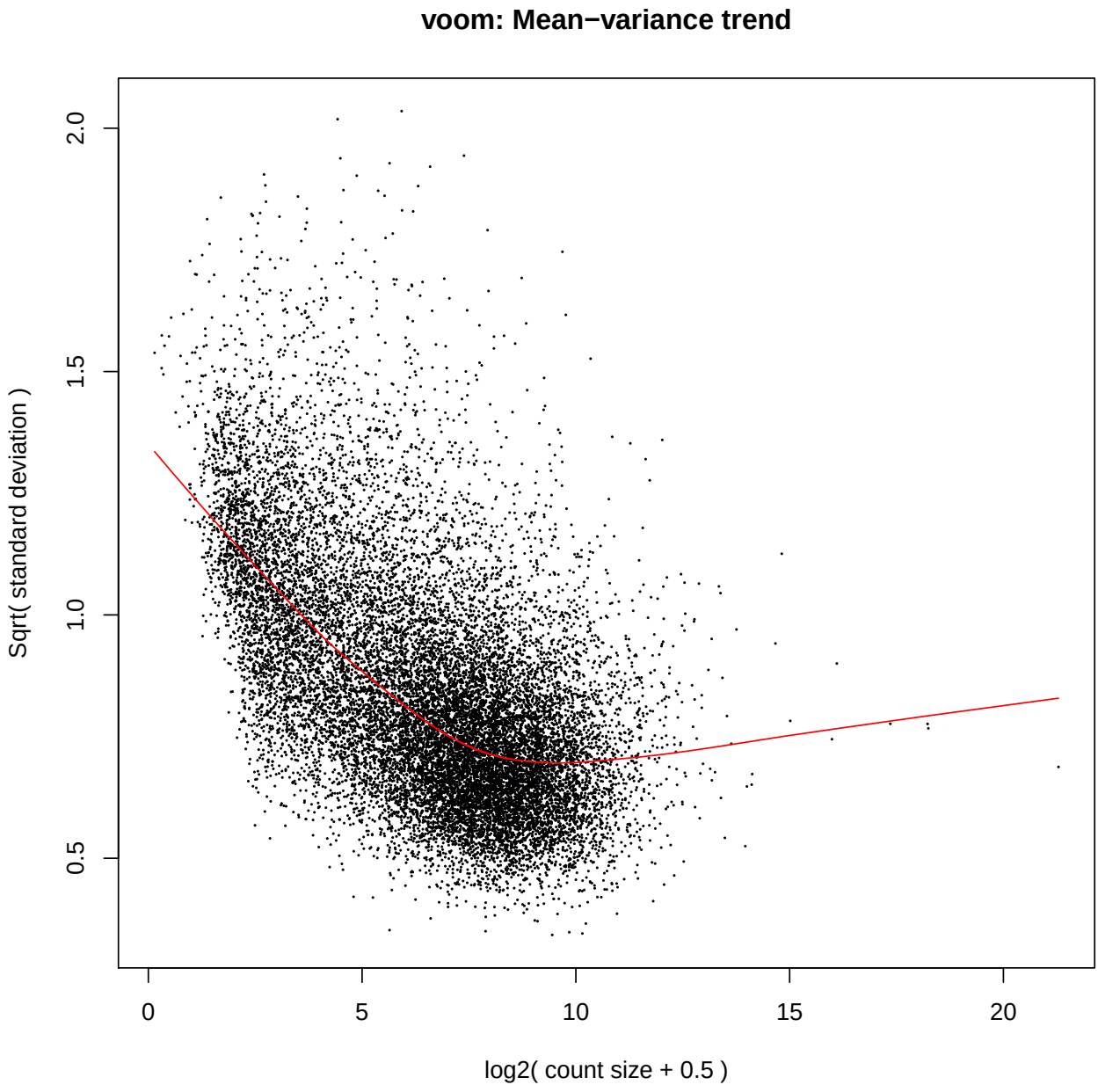
Having determined there are no sample outliers in these data, we carry out differential gene expression analysis. Differential gene expression between TNBC and HER2-positive samples are determined using `limma` and `voom`. Differentially expressed genes are visualized using a volcano plot to compare the effect size of the differential expression [as measured by the $log_2(fold - change)$ in expression] and its significance [$-log_10(p - value)$].

```
## Perform differential expression analysis with limma-voom
design <- model.matrix(~ rse$group)
design
```

```
##    (Intercept) rse$groupTNBC Breast Tumor
## 1            1                          1
## 2            1                          1
## 3            1                          1
## 4            1                          1
## 5            1                          1
```

```
## 6            1                          0
## 7            1                          0
## 8            1                          0
## 9            1                          0
## 10           1                          0
## 11           1                          1
## attr(,"assign")
## [1] 0 1
## attr(,"contrasts")
## attr(,"contrasts")$`rse$group`
## [1] "contr.treatment"
```

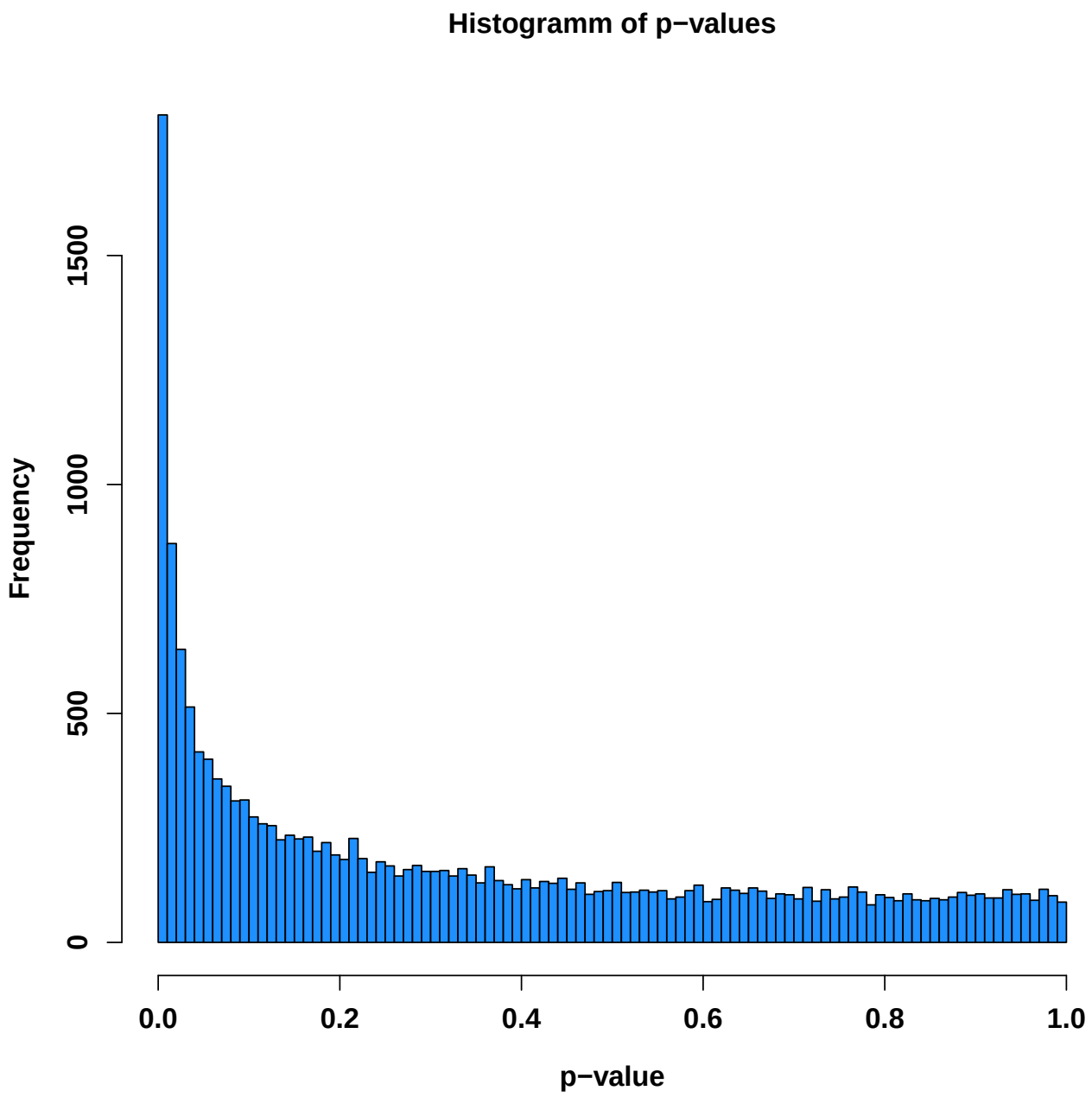
```
dge <- DGEList(counts = counts)
dge <- calcNormFactors(dge)
v <- voom(dge, design, plot = TRUE)
```



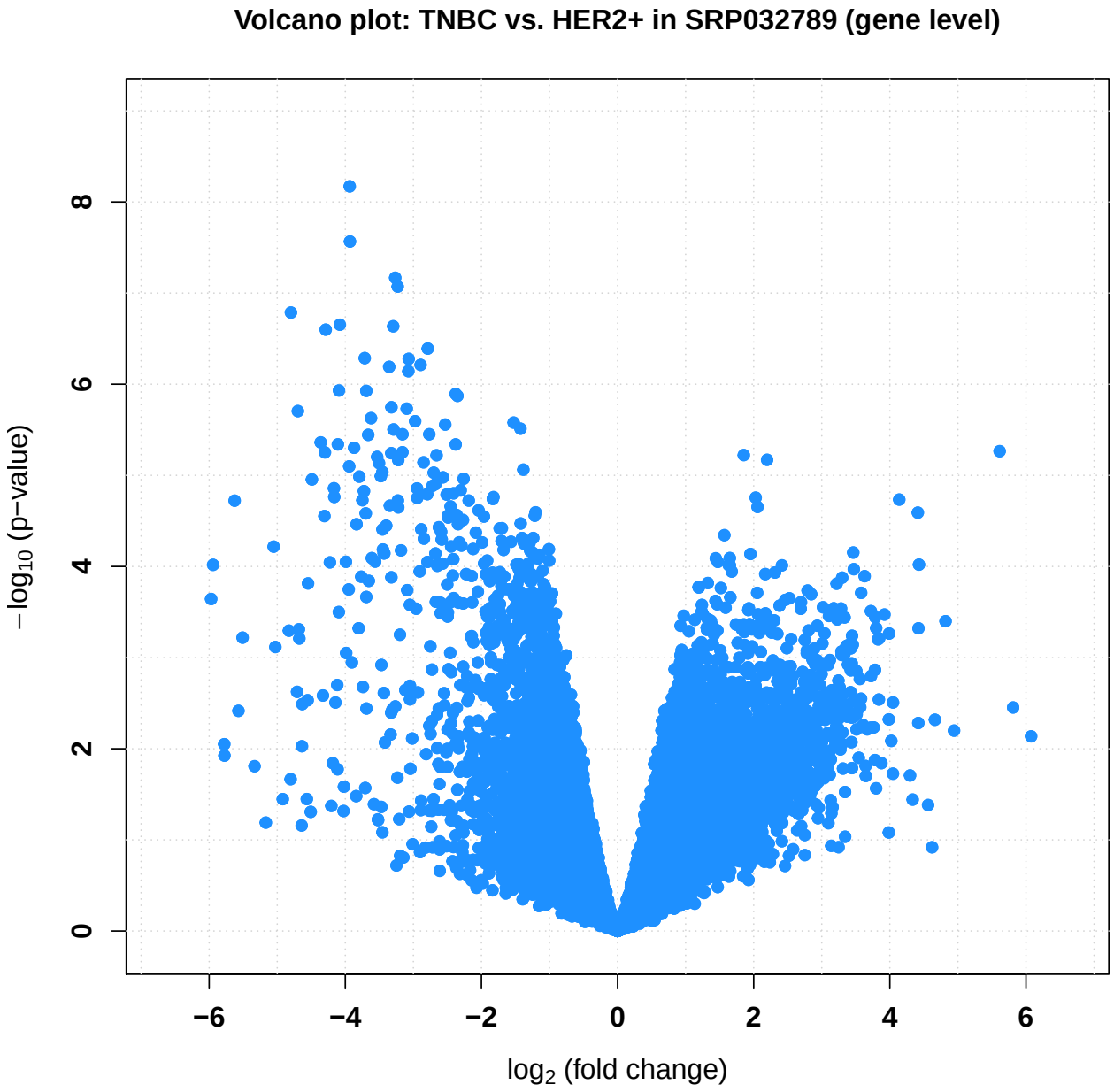
```
fit <- lmFit(v, design)
fit <- eBayes(fit)
log2FC <- fit$coefficients[, 2]
p.mod <- fit$p.value[, 2]
q.mod <- qvalue(p.mod)$q
res_gene <- data.frame(log2FC, p.mod, q.mod)
rownames(res_gene) <- rownames(counts)

## Determine the number of genes differentially expressed at q<0.05
sum(res_gene$q.mod < 0.05)
```

```
## [1] 1611
```

```
## Histogram of p-values
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
hist(p.mod, col = trop[2], xlab = 'p-value',
     main = 'Histogramm of p-values', breaks = 100)
```




```
## Volcano plot
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
rx2 <- c(-1, 1) * 1.1 * max(abs(log2FC))
ry2 <- c(-0.1, max(-log10(p.mod))) * 1.1
plot(log2FC, -log10(p.mod),
     pch = 19, xlim = rx2, ylim = ry2, col = trop[2],
     xlab = bquote(paste(log[2], ' (fold change)')),
     ylab = bquote(paste(-log[10], ' (p-value)')))
abline(v = seq(-10, 10, 1), col = 'lightgray', lty = 'dotted')
abline(h = seq(0, 23, 1), col = 'lightgray', lty = 'dotted')
points(log2FC, -log10(p.mod), pch = 19, col = trop[2])
title('Volcano plot: TNBC vs. HER2+ in SRP032789 (gene level)')
```



Gene set enrichment analysis

To get a better understanding of those genes showing differential gene expression, we utilize `topGO`, a gene set analysis library. Genes included in this analysis are those reaching a q-value cutoff less than 0.05.

```
names(q.mod) <- rownames(counts)
interesting <- function(x) x < 0.05
```

After determining which genes to include for analysis, `topGO` objects are generated and the enrichment tests are run. The Kolomogorov-Smirnov (`ks`) test is used to test for distributional differences. Here, we ask whether each GO group is "enriched" for differentially expressed (`q.mod < 0.05`) genes. Equivalently, we are testing whether the p-value distributions are the same for genes in and outside of each gene ontology. We run tests on the "biological processes" ontology.

```
topgoobjBP <- new('topGOdata',
    description = 'biological process',
    ontology = 'BP', allGenes = q.mod, geneSelectionFun = interesting,
    annotationFun = annFUN.org, mapping = 'org.Hs.eg.db', ID = 'entrez')
```

```
##
## Building most specific GOs .....
## Loading required package: org.Hs.eg.db
##
##  ( 10729 GO terms found. )
##
## Build GO DAG topology ..........
##  ( 14588 GO terms and 34554 relations. )
##
## Annotating nodes ...............
##  ( 13784 genes annotated to the GO terms. )
```

```
bptest <- runTest(topgoobjBP, algorithm = 'weight01', statistic = 'ks')
```

```
##
##           -- Weight01 Algorithm --
##
##       the algorithm is scoring 14588 nontrivial nodes
##       parameters:
##           test statistic: ks
##           score order: increasing
##
##   Level 20:  1 nodes to be scored    (0 eliminated genes)
##
##   Level 19:  7 nodes to be scored    (0 eliminated genes)
##
##   Level 18:  18 nodes to be scored   (1 eliminated genes)
##
##   Level 17:  41 nodes to be scored   (29 eliminated genes)
##
##   Level 16:  119 nodes to be scored  (82 eliminated genes)
##
##   Level 15:  238 nodes to be scored  (170 eliminated genes)
##
##   Level 14:  477 nodes to be scored  (495 eliminated genes)
##
##   Level 13:  827 nodes to be scored  (1140 eliminated genes)
##
##   Level 12:  1202 nodes to be scored (2246 eliminated genes)
##
##   Level 11:  1541 nodes to be scored (4239 eliminated genes)
```

```
##
##   Level 10:  1930 nodes to be scored (5909 eliminated genes)
##
##   Level 9:   2043 nodes to be scored (8114 eliminated genes)
##
##   Level 8:   1940 nodes to be scored (9773 eliminated genes)
##
##   Level 7:   1776 nodes to be scored (10997 eliminated genes)
##
##   Level 6:   1304 nodes to be scored (11911 eliminated genes)
##
##   Level 5:   723 nodes to be scored  (12580 eliminated genes)
##
##   Level 4:   302 nodes to be scored  (13126 eliminated genes)
##
##   Level 3:   77 nodes to be scored   (13351 eliminated genes)
##
##   Level 2:   21 nodes to be scored   (13513 eliminated genes)
##
##   Level 1:   1 nodes to be scored    (13586 eliminated genes)
```

```
bptest
```

```
##
## Description: biological process
## Ontology: BP
## 'weight01' algorithm with the 'ks' test
## 14588 GO terms scored: 48 terms with p < 0.01
## Annotation data:
##     Annotated genes: 13784
##     Significant genes: 1130
##     Min. no. of genes annotated to a GO: 1
##     Nontrivial nodes: 14588
```

```
bpres_gene <- GenTable(topgoobjBP, pval = bptest,
                       topNodes = length(bptest@score), numChar = 100)
head(bpres_gene, n = 10)
```

```
##         GO.ID                                          Term Annotated
## 1  GO:0016579                    protein deubiquitination       107
## 2  GO:0016569                      chromatin modification       491
## 3  GO:0030049                     muscle filament sliding        30
## 4  GO:0007050                           cell cycle arrest       223
## 5  GO:0071557                histone H3-K27 demethylation         4
## 6  GO:0071363 cellular response to growth factor stimulus       528
## 7  GO:0006606                  protein import into nucleus       250
## 8  GO:0008589  regulation of smoothened signaling pathway        59
## 9  GO:0043547      positive regulation of GTPase activity       557
## 10 GO:0006338                          chromatin remodeling       149
##    Significant Expected    pval
## 1           13     8.77 9.4e-05
```

```
## 2           56    40.25 0.00031
## 3            7     2.46 0.00041
## 4           29    18.28 0.00050
## 5            3     0.33 0.00124
## 6           57    43.28 0.00167
## 7           22    20.49 0.00172
## 8            8     4.84 0.00188
## 9           58    45.66 0.00188
## 10          18    12.21 0.00194
```

Exon level analysis

As above, we are interested here in differential expression. However, rather than summarizing across genes, this analysis will look for differential expression at the exon level. In this analysis, we include all exons that map to the previous filtered genes and again carry out differential expression analysis using `limma` and `voom`.

Here, we download data from the same project as above (SRP032798); however, this time, we are interested in obtaining the exon level data.

```
## Find a project of interest (SRP032789)
project_info <- abstract_search('To define the digital transcriptome of three breast cancer')
project_info
```

```
##     number_samples species
## 865             20   human
##
## 865 Goal: To define the digital transcriptome of three breast cancer subtypes (TNBC, Non-TNBC, and H
##       project
## 865 SRP032789
```

```
## Browse the project at SRA
browse_study(project_info$project)

## Download the exon level RangedSummarizedExperiment data
if(!file.exists(file.path('SRP032789', 'rse_exon.Rdata'))) {
    download_study(project_info$project, type = 'rse-exon')
}

## Load the data
load(file.path(project_info$project, 'rse_exon.Rdata'))
rse_exon
```

```
## class: RangedSummarizedExperiment
## dim: 226117 20
## metadata(0):
## assays(1): counts
## rownames(226117): 1 1 ... 9997 9997
## rowData names(0):
## colnames(20): SRR1027171 SRR1027173 ... SRR1027190 SRR1027172
## colData names(18): project sample ... avg_read_length bigwig_file
```

```
## This is the sample phenotype data provided by the recount project
colData(rse_exon)
```

```
## DataFrame with 20 rows and 18 columns
```

```
##               project      sample  experiment         run
##           <character> <character> <character> <character>
## SRR1027171   SRP032789   SRS500214   SRX374850  SRR1027171
## SRR1027173   SRP032789   SRS500216   SRX374852  SRR1027173
## SRR1027174   SRP032789   SRS500217   SRX374853  SRR1027174
## SRR1027175   SRP032789   SRS500218   SRX374854  SRR1027175
## SRR1027176   SRP032789   SRS500219   SRX374855  SRR1027176
## ...               ...         ...         ...         ...
## SRR1027187   SRP032789   SRS500230   SRX374866  SRR1027187
## SRR1027188   SRP032789   SRS500231   SRX374867  SRR1027188
## SRR1027189   SRP032789   SRS500232   SRX374868  SRR1027189
## SRR1027190   SRP032789   SRS500233   SRX374869  SRR1027190
## SRR1027172   SRP032789   SRS500215   SRX374851  SRR1027172
##           read_count_as_reported_by_sra reads_aligned
##                               <integer>     <integer>
## SRR1027171                      88869444      88869444
## SRR1027173                     107812596     107812596
## SRR1027174                      98563260      98563260
## SRR1027175                      91327892      91327892
## SRR1027176                      96513572      96513572
## ...                                  ...           ...
## SRR1027187                      75260678      75260678
## SRR1027188                      65709192      65709192
## SRR1027189                      65801392      65801392
## SRR1027190                      74356276      74356276
## SRR1027172                      80986440      58902122
##           proportion_of_reads_reported_by_sra_aligned paired_end
##                                             <numeric>  <logical>
## SRR1027171                                          1       TRUE
## SRR1027173                                          1       TRUE
## SRR1027174                                          1       TRUE
## SRR1027175                                          1       TRUE
## SRR1027176                                          1       TRUE
## ...                                               ...        ...
## SRR1027187                                  1.0000000       TRUE
## SRR1027188                                  1.0000000       TRUE
## SRR1027189                                  1.0000000       TRUE
## SRR1027190                                  1.0000000       TRUE
## SRR1027172                                  0.7273084       TRUE
##           sra_misreported_paired_end mapped_read_count        auc
##                            <logical>         <integer>  <numeric>
## SRR1027171                      FALSE          86949307 5082692127
## SRR1027173                      FALSE         104337779 6077034329
## SRR1027174                      FALSE          95271238 5504462845
## SRR1027175                      FALSE          88820239 5150234117
## SRR1027176                      FALSE          93464650 5416681912
## ...                               ...               ...        ...
## SRR1027187                      FALSE          64697612 3567078255
## SRR1027188                      FALSE          65278500 4856453823
## SRR1027189                      FALSE          65328289 4858587600
## SRR1027190                      FALSE          73911898 5501089036
## SRR1027172                      FALSE          57523391 3351013968
##           sharq_tissue sharq_cell_type biosample_submission_date
##            <character>     <character>                <character>
```

```
## SRR1027171       breast             esc   2013-11-07T12:40:22.203
## SRR1027173       breast             esc   2013-11-07T12:40:32.283
## SRR1027174       breast             esc   2013-11-07T12:40:28.283
## SRR1027175       breast             esc   2013-11-07T12:40:34.343
## SRR1027176       breast             esc   2013-11-07T12:40:36.303
## ...                 ...             ...                       ...
## SRR1027187       breast             esc   2013-11-07T12:40:56.180
## SRR1027188       breast             esc   2013-11-07T12:40:58.170
## SRR1027189       breast             esc   2013-11-07T12:40:20.227
## SRR1027190       breast             esc   2013-11-07T12:40:18.090
## SRR1027172       breast             esc   2013-11-07T12:40:26.217
##            biosample_publication_date   biosample_update_date
##                             <character>             <character>
## SRR1027171    2013-11-08T01:11:17.160 2014-03-07T16:09:38.542
## SRR1027173    2013-11-08T01:11:14.827 2014-03-07T16:09:38.698
## SRR1027174    2013-11-08T01:11:52.283 2014-03-07T16:09:38.637
## SRR1027175    2013-11-08T01:11:15.963 2014-03-07T16:09:38.731
## SRR1027176    2013-11-08T01:11:46.430 2014-03-07T16:09:38.768
## ...                               ...                     ...
## SRR1027187    2013-11-08T01:11:29.587 2014-03-07T16:09:39.093
## SRR1027188    2013-11-08T01:12:06.660 2014-03-07T16:09:39.130
## SRR1027189    2013-11-08T01:11:33.080 2014-03-07T16:09:38.498
## SRR1027190    2013-11-08T01:12:11.320 2014-03-07T16:09:38.469
## SRR1027172    2013-11-08T01:11:45.250 2014-03-07T16:09:38.604
##            avg_read_length   bigwig_file
##                  <integer>   <character>
## SRR1027171             120 SRR1027171.bw
## SRR1027173             120 SRR1027173.bw
## SRR1027174             120 SRR1027174.bw
## SRR1027175             120 SRR1027175.bw
## SRR1027176             120 SRR1027176.bw
## ...                    ...           ...
## SRR1027187             120 SRR1027187.bw
## SRR1027188             150 SRR1027188.bw
## SRR1027189             150 SRR1027189.bw
## SRR1027190             150 SRR1027190.bw
## SRR1027172              87 SRR1027172.bw
```

As above, downloaded count data are first scaled to take into account differing coverage between samples. The same phenotype data (`pheno`) are used and again ordered to match the sample order of the expression data (`rse_exon`). Only those samples that are HER2-positive or TNBC are included for analysis. Prior to differential exon expression analysis, count data are obtained in matrix format and then filtered to only include exons within genes that had been analyzed previously.

```
## Scale counts by taking into account the total coverage per sample
rse <- scale_counts(rse_exon)

## Download pheno data from
## http://trace.ncbi.nlm.nih.gov/Traces/study/?acc=SRP032789
pheno <- read.table('SraRunTable_SRP032789.txt', sep = '\t',
                    header = TRUE,
                    stringsAsFactors = FALSE)

## Obtain correct order for pheno data
pheno <- pheno[match(rse$run, pheno$Run_s), ]
```

```
identical(pheno$Run_s, rse$run)
```

```
## [1] TRUE
```

```
head(cbind(pheno$Run_s, rse$run))
```

```
##      [,1]         [,2]
## [1,] "SRR1027171" "SRR1027171"
## [2,] "SRR1027173" "SRR1027173"
## [3,] "SRR1027174" "SRR1027174"
## [4,] "SRR1027175" "SRR1027175"
## [5,] "SRR1027176" "SRR1027176"
## [6,] "SRR1027177" "SRR1027177"
```

```
## Obtain grouping information
colData(rse)$group <- pheno$tumor_type_s
table(colData(rse)$group)
```

```
##
## HER2 Positive Breast Tumor      Non-TNBC Breast Tumor
##                          5                          6
##    Normal Breast Organoids          TNBC Breast Tumor
##                          3                          6
```

```
## Subset data to HER2 and TNBC types
rse <- rse[, rse$group %in% c('HER2 Positive Breast Tumor',
                              'TNBC Breast Tumor')]

## Save filtered rse object
rse_exon_filt <- rse
rse_exon_filt
```

```
## class: RangedSummarizedExperiment
## dim: 226117 11
## metadata(0):
## assays(1): counts
## rownames(226117): 1 1 ... 9997 9997
## rowData names(0):
## colnames(11): SRR1027171 SRR1027173 ... SRR1027187 SRR1027172
## colData names(19): project sample ... bigwig_file group
```

```
## Obtain count matrix
counts <- assays(rse_exon_filt)$counts
dim(counts)
```

```
## [1] 226117     11
```

```
## Filter count matrix (keep exons that are in filtered gene counts matrix)
filter <- rownames(counts) %in% rownames(counts_gene)
counts <- counts[filter, ]
dim(counts)
```

```
## [1] 204559     11
```

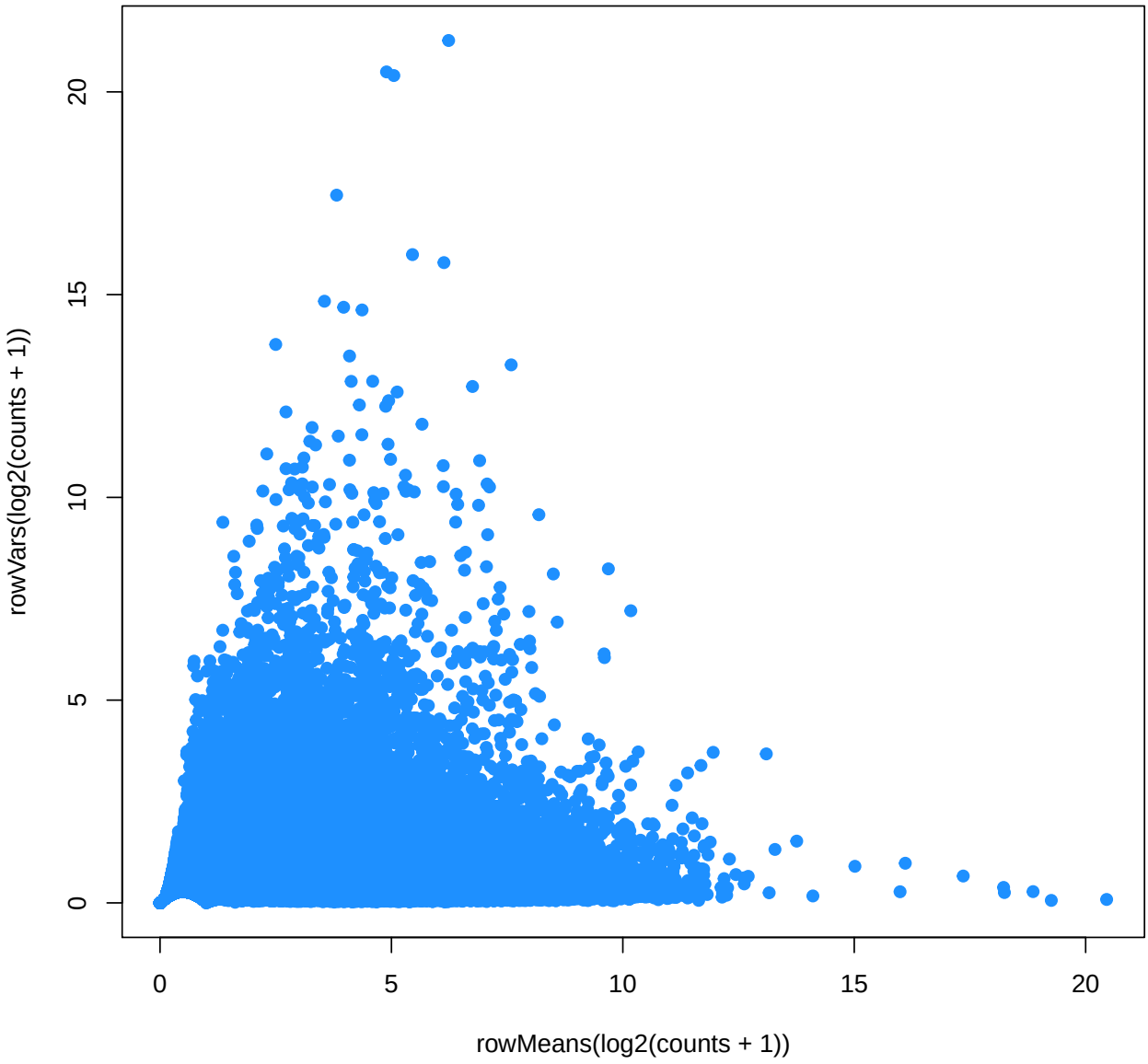
```
## Save for gene, exon and junction comparisons
counts_exon <- counts
counts_exon[1:5, 1:5]
```

```
##   SRR1027171 SRR1027173 SRR1027174 SRR1027175 SRR1027176
```

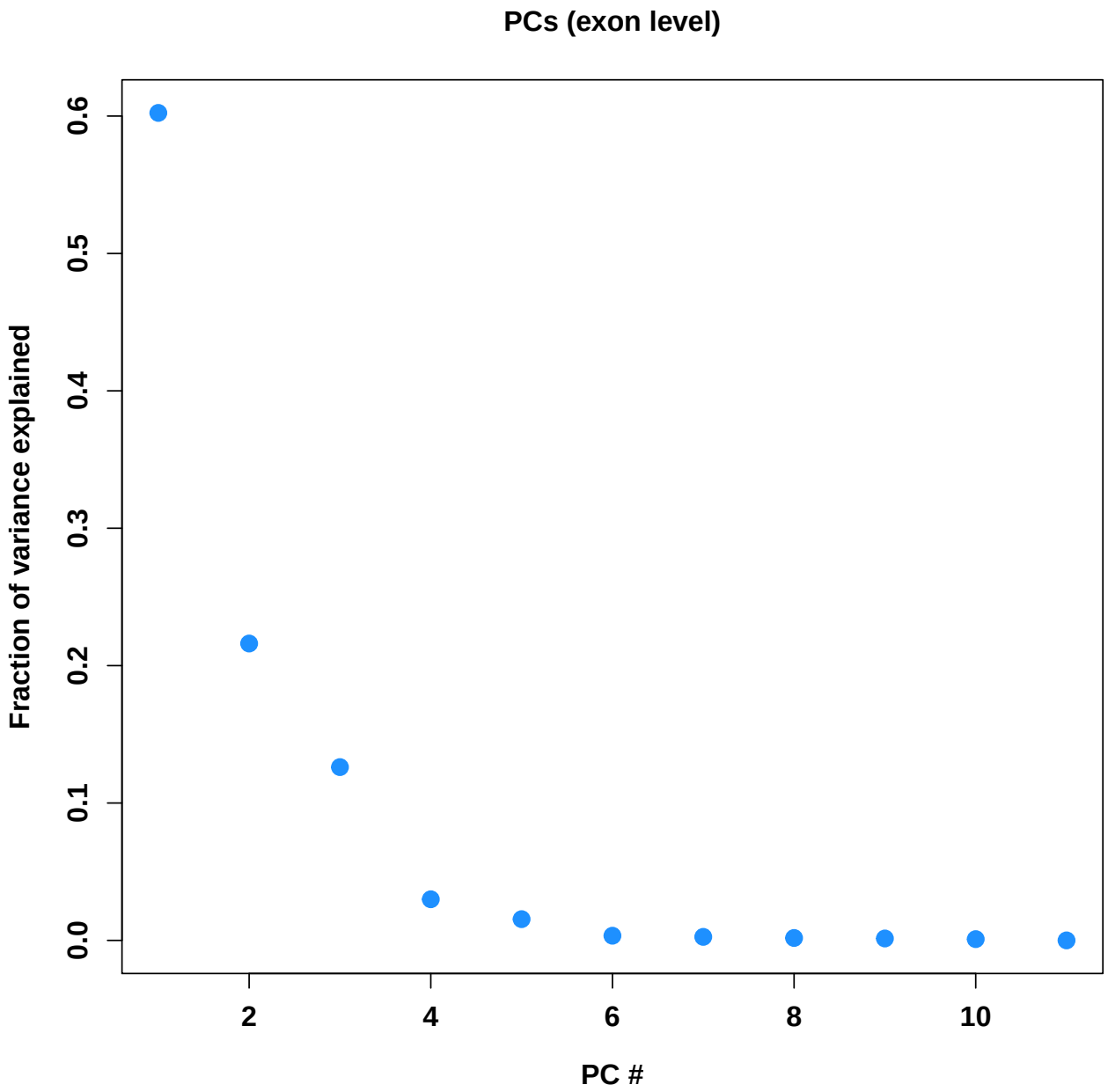
```
## 1         10          6          8         15         10
## 1         10          9         17         23         19
## 1          7          3          1          3          6
## 1         14         13          4          6         15
## 1          3          2          1          1          2
```

As above, to get a better sense of the data, we assess the mean-variance relationship for each exon. Similarly, we run principal component analysis (PCA) to identify any sample outliers within the data. We assess the variance explained by each of the first 11 PCs as well as visualize the relationship of each sample in the first two PCs.

```
## Set colors
trop <- RSkittleBrewer('tropical')[c(1, 2)]
cols <- as.numeric(as.factor(rse$group))

## Look at mean variance relationship
plot(rowMeans(log2(counts + 1)), rowVars(log2(counts + 1)),
     pch = 19, col = trop[2])
```

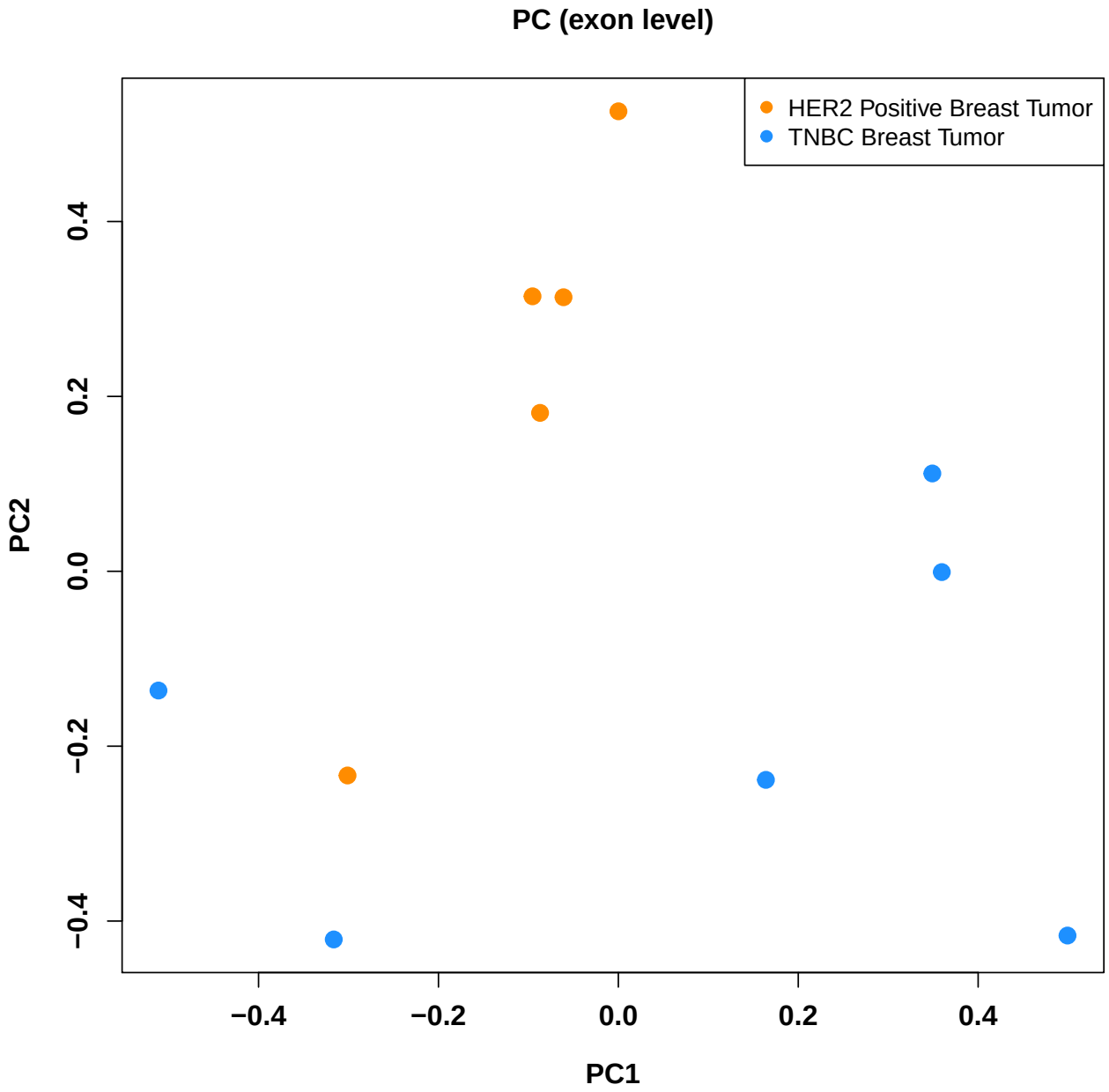


```
## Calculate PCs with svd function
expr.pca <- svd(counts - rowMeans(counts))

## Plot PCs
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
plot(expr.pca$d^2 / sum(expr.pca$d^2), pch = 19, col = trop[2], cex = 1.5,
     ylab = 'Fraction of variance explained', xlab = 'PC #',
     main = 'PCs (exon level)')
```




```
## Plot PC1 vs. PC2
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
plot(expr.pca$v[, 1], expr.pca$v[, 2], pch = 19, col = trop[cols], cex = 1.5,
     xlab = 'PC1', ylab = 'PC2',
     main = 'PC (exon level)')
legend('topright', pch = 19, col = trop[c(1, 2)],
```

```
      names(summary(as.factor(rse$group))), bg="white")
```



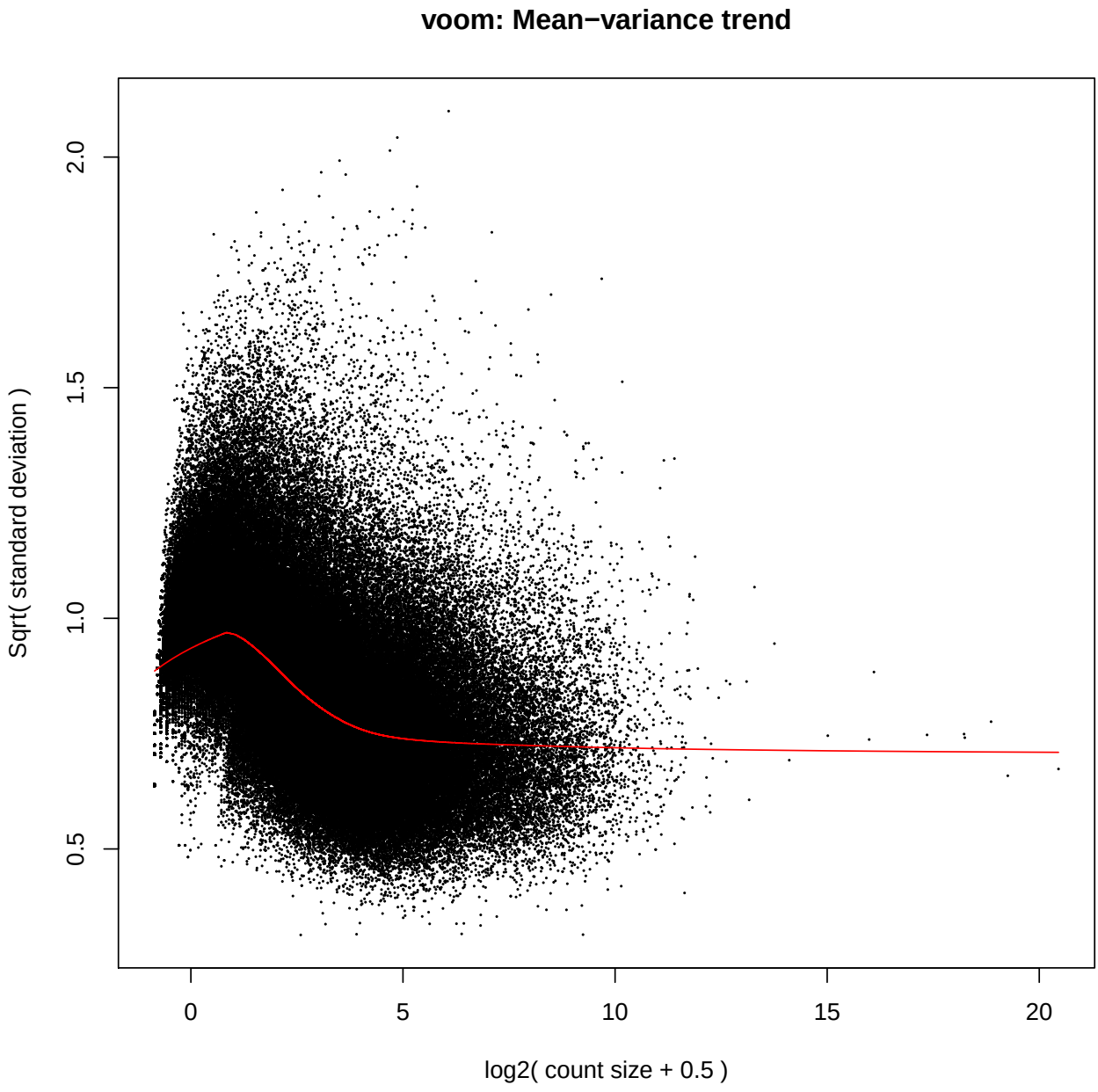
Again, differential expression analysis is carried out using `limma` and `voom`; however, this time at the exon, rather than gene, level. Data are again visualized using a volcano plot to assess the strength [$log_2(fold - change)$ in expression] and its significance [$-log_10(p - value)$].for each exon.

```
design <- model.matrix(~ rse$group)
design
```

```
##    (Intercept) rse$groupTNBC Breast Tumor
## 1            1                          1
## 2            1                          1
## 3            1                          1
## 4            1                          1
## 5            1                          1
```

```
## 6            1                       0
## 7            1                       0
## 8            1                       0
## 9            1                       0
## 10           1                       0
## 11           1                       1
## attr(,"assign")
## [1] 0 1
## attr(,"contrasts")
## attr(,"contrasts")$`rse$group`
## [1] "contr.treatment"
```

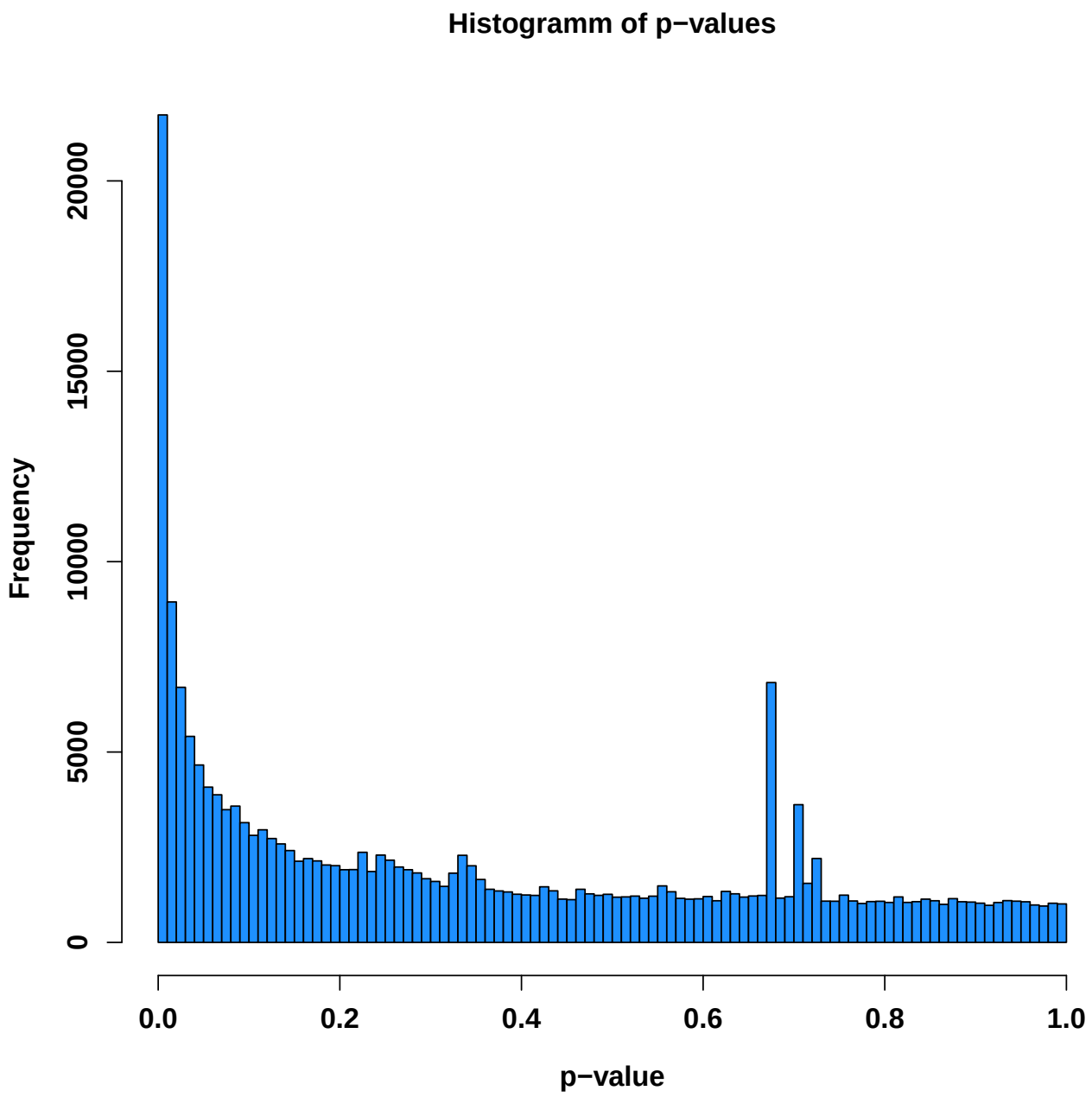
```
dge <- DGEList(counts = counts)
dge <- calcNormFactors(dge)
v <- voom(dge, design,plot = TRUE)
```



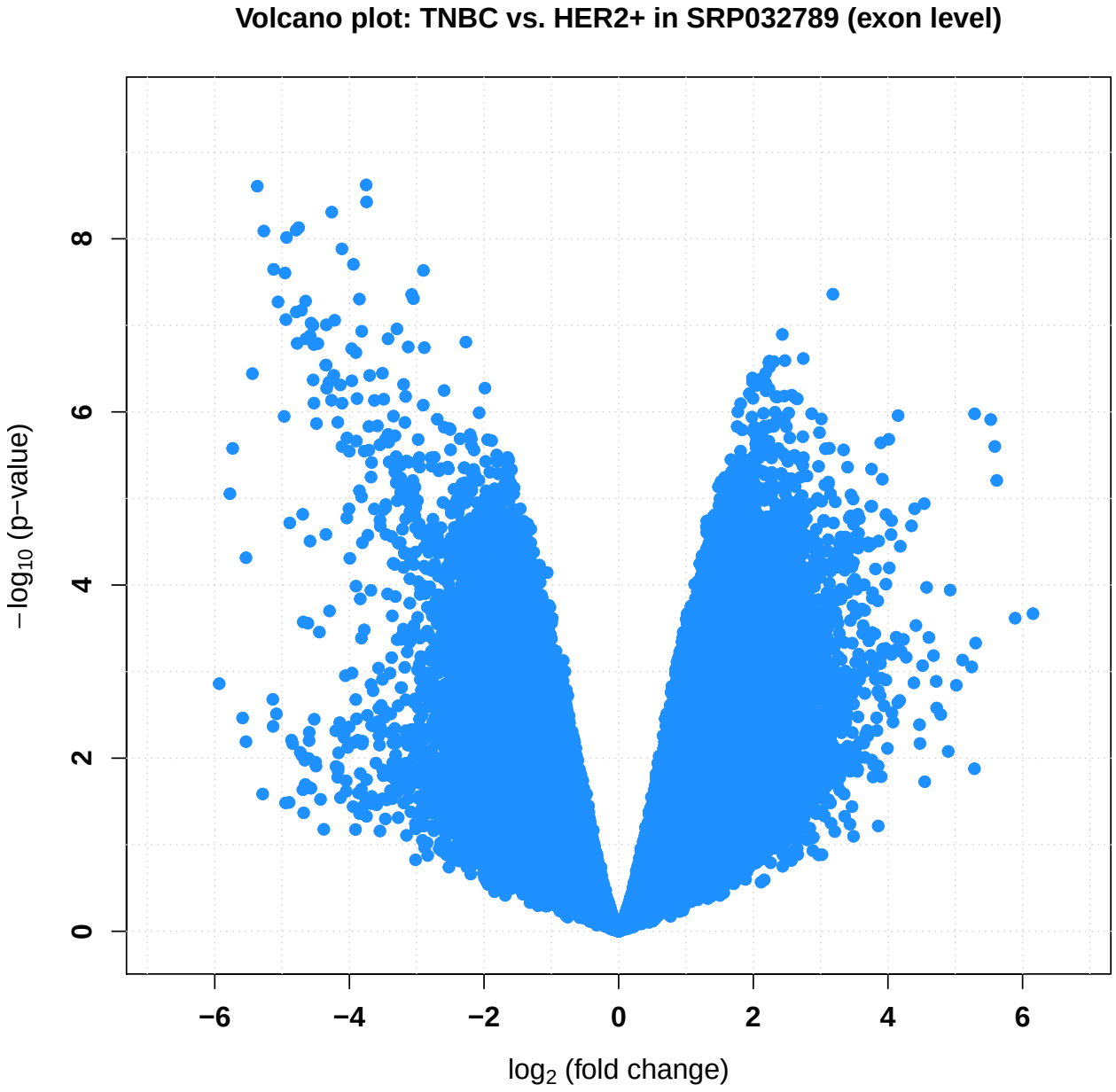
```
fit <- lmFit(v, design)
fit <- eBayes(fit)
log2FC <- fit$coefficients[, 2]
p.mod <- fit$p.value[, 2]
q.mod <- qvalue(p.mod)$q
res_exon <- data.frame(log2FC, p.mod, q.mod)


## Determine the number of exons differentially expressed at q<0.05
sum(res_exon$q.mod < 0.05)
```

```
## [1] 23647
```

```
## Histogram of p-values
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
hist(p.mod, col = trop[2], xlab = 'p-value',
     main = 'Histogramm of p-values', breaks = 100)
```




```
## Volcano plot
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
rx2 <- c(-1, 1) * 1.1 * max(abs(log2FC))
ry2 <- c(-0.1, max(-log10(p.mod))) * 1.1
plot(log2FC, -log10(p.mod),
     pch = 19, xlim = rx2, ylim = ry2, col = trop[2],
     xlab = bquote(paste(log[2], ' (fold change)')),
     ylab = bquote(paste(-log[10], ' (p-value)')))
abline(v = seq(-10, 10, 1), col = 'lightgray', lty = 'dotted')
abline(h = seq(0, 23, 1), col = 'lightgray', lty = 'dotted')
points(log2FC, -log10(p.mod), pch = 19, col = trop[2])
title('Volcano plot: TNBC vs. HER2+ in SRP032789 (exon level)')
```



Junction level analysis

As above, we are interested here in differential expression. However, rather than summarizing across genes, this analysis will look for differential expression at the junction level. In this analysis, we include all junctions that map to the previous filtered genes and again carry out differential expression analysis using `limma` and `voom`.

Here, we download data from the same project as above (SRP032798); however, this time, we are interested in obtaining the junction level data.

```
## Find a project of interest (SRP032789)
project_info <- abstract_search('To define the digital transcriptome of three breast cancer')
project_info
```

```
##     number_samples species
## 865             20   human
##
## 865 Goal: To define the digital transcriptome of three breast cancer subtypes (TNBC, Non-TNBC, and H
##       project
## 865 SRP032789
```

```
## Browse the project at SRA
browse_study(project_info$project)

## Download the exon level RangedSummarizedExperiment data
if(!file.exists(file.path('SRP032789', 'rse_jx.Rdata'))) {
    download_study(project_info$project, type = 'rse-jx')
}

## Load the data
load(file.path(project_info$project, 'rse_jx.Rdata'))
rse_jx
```

```
## class: RangedSummarizedExperiment
## dim: 672203 20
## metadata(0):
## assays(1): counts
## rownames: NULL
## rowData names(8): junction_id found_junction_gencode_v24 ...
##   symbol class
## colnames(20): SRR1027171 SRR1027173 ... SRR1027190 SRR1027172
## colData names(21): project sample ... title characteristics
```

```
## This is the sample phenotype data provided by the recount project
colData(rse_jx)
```

```
## DataFrame with 20 rows and 21 columns
##                project      sample  experiment         run
##            <character> <character> <character> <character>
## SRR1027171   SRP032789   SRS500214   SRX374850  SRR1027171
## SRR1027173   SRP032789   SRS500216   SRX374852  SRR1027173
## SRR1027174   SRP032789   SRS500217   SRX374853  SRR1027174
## SRR1027175   SRP032789   SRS500218   SRX374854  SRR1027175
## SRR1027176   SRP032789   SRS500219   SRX374855  SRR1027176
## ...               ...         ...         ...         ...
## SRR1027187   SRP032789   SRS500230   SRX374866  SRR1027187
## SRR1027188   SRP032789   SRS500231   SRX374867  SRR1027188
## SRR1027189   SRP032789   SRS500232   SRX374868  SRR1027189
## SRR1027190   SRP032789   SRS500233   SRX374869  SRR1027190
## SRR1027172   SRP032789   SRS500215   SRX374851  SRR1027172
##            read_count_as_reported_by_sra reads_downloaded
##                                <integer>       <integer>
## SRR1027171                      88869444        88869444
## SRR1027173                     107812596       107812596
## SRR1027174                      98563260        98563260
## SRR1027175                      91327892        91327892
## SRR1027176                      96513572        96513572
## ...                                  ...             ...
## SRR1027187                      75260678        75260678
## SRR1027188                      65709192        65709192
```

```
## SRR1027189                     65801392         65801392
## SRR1027190                     74356276         74356276
## SRR1027172                     80986440         58902122
##            proportion_of_reads_reported_by_sra_downloaded paired_end
##                                                 <numeric>  <logical>
## SRR1027171                                              1       TRUE
## SRR1027173                                              1       TRUE
## SRR1027174                                              1       TRUE
## SRR1027175                                              1       TRUE
## SRR1027176                                              1       TRUE
## ...                                                   ...        ...
## SRR1027187                                      1.0000000       TRUE
## SRR1027188                                      1.0000000       TRUE
## SRR1027189                                      1.0000000       TRUE
## SRR1027190                                      1.0000000       TRUE
## SRR1027172                                      0.7273084       TRUE
##            sra_misreported_paired_end mapped_read_count        auc
##                             <logical>         <integer>  <numeric>
## SRR1027171                      FALSE          86949307 5082692127
## SRR1027173                      FALSE         104337779 6077034329
## SRR1027174                      FALSE          95271238 5504462845
## SRR1027175                      FALSE          88820239 5150234117
## SRR1027176                      FALSE          93464650 5416681912
## ...                               ...               ...        ...
## SRR1027187                      FALSE          64697612 3567078255
## SRR1027188                      FALSE          65278500 4856453823
## SRR1027189                      FALSE          65328289 4858587600
## SRR1027190                      FALSE          73911898 5501089036
## SRR1027172                      FALSE          57523391 3351013968
##            sharq_beta_tissue sharq_beta_cell_type
##                  <character>          <character>
## SRR1027171            breast                  esc
## SRR1027173            breast                  esc
## SRR1027174            breast                  esc
## SRR1027175            breast                  esc
## SRR1027176            breast                  esc
## ...                      ...                  ...
## SRR1027187            breast                  esc
## SRR1027188            breast                  esc
## SRR1027189            breast                  esc
## SRR1027190            breast                  esc
## SRR1027172            breast                  esc
##            biosample_submission_date biosample_publication_date
##                          <character>                <character>
## SRR1027171   2013-11-07T12:40:22.203    2013-11-08T01:11:17.160
## SRR1027173   2013-11-07T12:40:32.283    2013-11-08T01:11:14.827
## SRR1027174   2013-11-07T12:40:28.283    2013-11-08T01:11:52.283
## SRR1027175   2013-11-07T12:40:34.343    2013-11-08T01:11:15.963
## SRR1027176   2013-11-07T12:40:36.303    2013-11-08T01:11:46.430
## ...                             ...                        ...
## SRR1027187   2013-11-07T12:40:56.180    2013-11-08T01:11:29.587
## SRR1027188   2013-11-07T12:40:58.170    2013-11-08T01:12:06.660
## SRR1027189   2013-11-07T12:40:20.227    2013-11-08T01:11:33.080
## SRR1027190   2013-11-07T12:40:18.090    2013-11-08T01:12:11.320
```

```
## SRR1027172   2013-11-07T12:40:26.217    2013-11-08T01:11:45.250
##              biosample_update_date avg_read_length geo_accession
##                        <character>       <integer>   <character>
## SRR1027171 2014-03-07T16:09:38.542             120    GSM1261016
## SRR1027173 2014-03-07T16:09:38.698             120    GSM1261018
## SRR1027174 2014-03-07T16:09:38.637             120    GSM1261019
## SRR1027175 2014-03-07T16:09:38.731             120    GSM1261020
## SRR1027176 2014-03-07T16:09:38.768             120    GSM1261021
## ...                            ...             ...           ...
## SRR1027187 2014-03-07T16:09:39.093             120    GSM1261032
## SRR1027188 2014-03-07T16:09:39.130             150    GSM1261033
## SRR1027189 2014-03-07T16:09:38.498             150    GSM1261034
## SRR1027190 2014-03-07T16:09:38.469             150    GSM1261035
## SRR1027172 2014-03-07T16:09:38.604              87    GSM1261017
##              bigwig_file       title
##              <character> <character>
## SRR1027171 SRR1027171.bw       TNBC1
## SRR1027173 SRR1027173.bw       TNBC3
## SRR1027174 SRR1027174.bw       TNBC4
## SRR1027175 SRR1027175.bw       TNBC5
## SRR1027176 SRR1027176.bw       TNBC6
## ...                   ...         ...
## SRR1027187 SRR1027187.bw      HER2-5
## SRR1027188 SRR1027188.bw        NBS1
## SRR1027189 SRR1027189.bw        NBS2
## SRR1027190 SRR1027190.bw        NBS3
## SRR1027172 SRR1027172.bw       TNBC2
##                                   characteristics
##                                   <CharacterList>
## SRR1027171          tumor type: TNBC Breast Tumor
## SRR1027173          tumor type: TNBC Breast Tumor
## SRR1027174          tumor type: TNBC Breast Tumor
## SRR1027175          tumor type: TNBC Breast Tumor
## SRR1027176          tumor type: TNBC Breast Tumor
## ...                                           ...
## SRR1027187 tumor type: HER2 Positive Breast Tumor
## SRR1027188    tumor type: Normal Breast Organoids
## SRR1027189    tumor type: Normal Breast Organoids
## SRR1027190    tumor type: Normal Breast Organoids
## SRR1027172          tumor type: TNBC Breast Tumor
```

As above, downloaded count data are first scaled to take into account differing coverage between samples. The same phenotype data (`pheno`) are used and again ordered to match the sample order of the expression data (`rse_jx`). Only those samples that are HER2-positive or TNBC are included for analysis. Prior to differential exon expression analysis, count data are obtained in matrix format and then filtered to only include junction within genes that had been analyzed previously.

```
## Scale counts by taking into account the total coverage per sample
rse <- scale_counts(rse_jx, by = 'mapped_reads', round = FALSE)

## Download pheno data from
## http://trace.ncbi.nlm.nih.gov/Traces/study/?acc=SRP032789
pheno <- read.table('SraRunTable_SRP032789.txt', sep = '\t',
                    header = TRUE,
                    stringsAsFactors = FALSE)
```

```
## Obtain correct order for pheno data
pheno <- pheno[match(rse$run, pheno$Run_s), ]
identical(pheno$Run_s, rse$run)
```

```
## [1] TRUE
```

```
head(cbind(pheno$Run_s, rse$run))
```

```
##      [,1]         [,2]
## [1,] "SRR1027171" "SRR1027171"
## [2,] "SRR1027173" "SRR1027173"
## [3,] "SRR1027174" "SRR1027174"
## [4,] "SRR1027175" "SRR1027175"
## [5,] "SRR1027176" "SRR1027176"
## [6,] "SRR1027177" "SRR1027177"
```

```
## Obtain grouping information
colData(rse)$group <- pheno$tumor_type_s
table(colData(rse)$group)
```

```
##
## HER2 Positive Breast Tumor      Non-TNBC Breast Tumor
##                          5                          6
##    Normal Breast Organoids          TNBC Breast Tumor
##                          3                          6
```

```
## Subset data to HER2 and TNBC types
rse <- rse[, rse$group %in% c('HER2 Positive Breast Tumor',
                              'TNBC Breast Tumor')]

## Save filtered rse object
rse_jx_filt <- rse
rse_jx_filt
```

```
## class: RangedSummarizedExperiment
## dim: 672203 11
## metadata(0):
## assays(1): counts
## rownames: NULL
## rowData names(8): junction_id found_junction_gencode_v24 ...
##   symbol class
## colnames(11): SRR1027171 SRR1027173 ... SRR1027187 SRR1027172
## colData names(22): project sample ... characteristics group
```

```
## Obtain count matrix
counts <- assays(rse_jx_filt)$counts
dim(counts)
```

```
## [1] 672203     11
```

```
##### Start: Obtain geneIDs for juctions
## Obtain geneIDs
gene_id <- rownames(counts_gene)

## Save number of genes that a junctions maps to
## We will exclude non-unique junctions later
num_genes <- lapply(rowData(rse_jx_filt)$gene_id_proposed, function(x) length(x))
```

```
num_genes <- unlist(num_genes)

## Save only the first gene_id
jx_gene_id <- lapply(rowData(rse_jx_filt)$gene_id, function(x) x[1])
jx_gene_id <- unlist(jx_gene_id)

## There are NAs: not every junctions is annotated
jx_gene_id[1:100]
```

```
##   [1] NA       NA       "653635" NA       NA       "653635" NA
##   [8] NA       NA       NA       NA       NA       NA       NA
##  [15] NA       NA       NA       NA       NA       NA       NA
##  [22] NA       NA       NA       NA       NA       "653635" NA
##  [29] NA       NA       NA       NA       NA       NA       NA
##  [36] NA       NA       NA       NA       NA       NA       NA
##  [43] NA       NA       NA       NA       NA       NA       NA
##  [50] NA       NA       "653635" NA       NA       NA       NA
##  [57] NA       NA       NA       NA       NA       NA       NA
##  [64] NA       NA       NA       NA       NA       NA       NA
##  [71] NA       NA       NA       NA       NA       NA       NA
##  [78] NA       NA       NA       NA       NA       NA       NA
##  [85] NA       NA       NA       NA       NA       NA       NA
##  [92] NA       NA       NA       NA       NA       NA       NA
##  [99] NA       NA
```

```
## Compare lengths
length(jx_gene_id) == dim(counts)[1]
```

```
## [1] TRUE
```

```
## Find non-unique mapping junctions
double_jx <- which(num_genes >1)

## Check non-unique mapping junctions
rowData(rse_jx_filt)[double_jx, 'gene_id_proposed']
```

```
## CharacterList of length 3411
## [[1]] 729737 100132062
## [[2]] 7293 8784
## [[3]] 219293 83858
## [[4]] 219293 83858
## [[5]] 219293 83858
## [[6]] 219293 83858
## [[7]] 55210 83858
## [[8]] 728642 984
## [[9]] 728642 984
## [[10]] 728642 984
## ...
## <3401 more elements>
```

```
## Set non-unique mapping junctions to "NA" in
jx_gene_id[double_jx] <- NA

rownames(counts) <- jx_gene_id
##### End: Obtain geneIDs for juctions
```

```
## Filter count matrix (keep exons that are in filtered gene counts matrix)
filter <- rownames(counts) %in% rownames(counts_gene)
counts <- counts[filter, ]
dim(counts)
```

```
## [1] 187013     11
```

```
## Since we only look at a subset of samples, there are many junctions with zero counts
## We remove them
counts <- counts[apply(counts, 1, sum) > 0, ]
dim(counts)
```

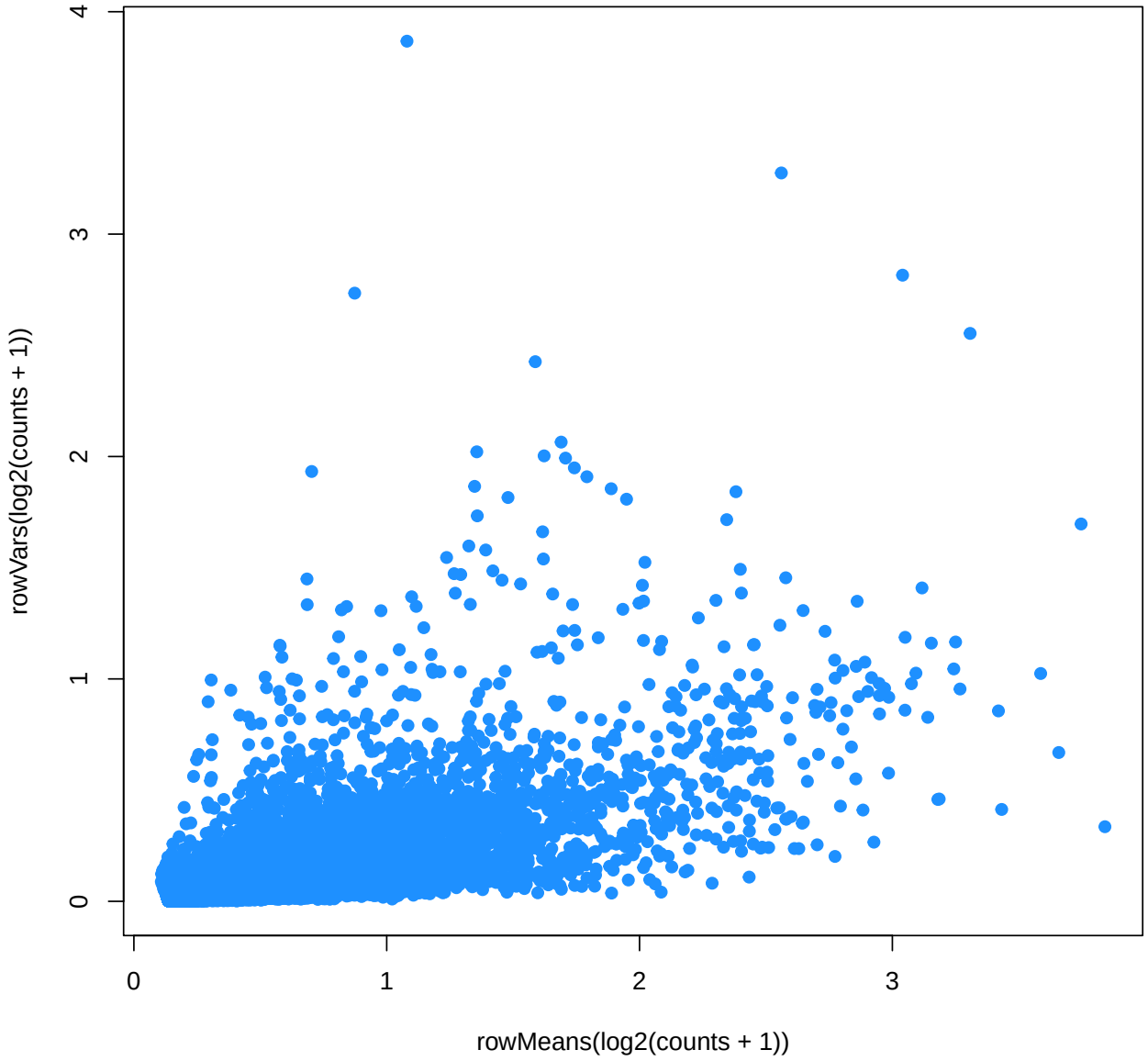
```
## [1] 171193     11
```

```
## Remove junctions with low counts across samples
counts <- counts[rowMeans(counts) > 0.1, ]

## Save for gene, exon and junction comparisons
counts_jx <- counts
counts_jx[1:10, ]
```

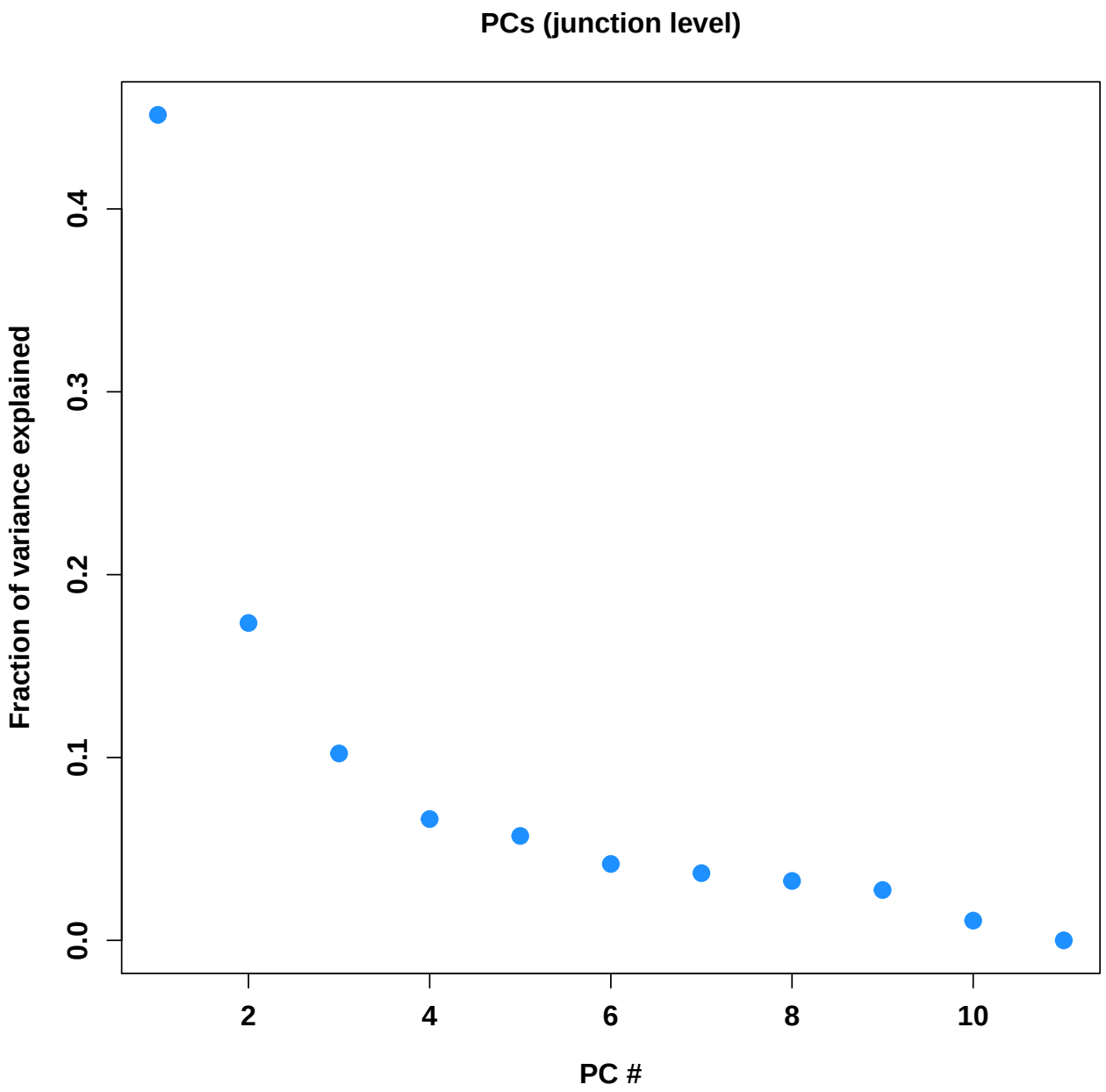
```
##        SRR1027171 SRR1027173 SRR1027174 SRR1027175 SRR1027176 SRR1027183
## 26155  0.20446141 0.10649173  0.1399513  0.1125870 0.43985733 0.00000000
## 26155  0.10223070 0.04259669  0.1516139  0.1125870 0.13076839 0.05579103
## 26155  0.14056722 0.07454421  0.1166261  0.1501159 0.16643250 0.08368654
## 26155  0.25557676 0.10649173  0.2099269  0.2501932 0.24964875 0.07438803
## 26155  0.10223070 0.02129835  0.2682400  0.2501932 0.26153679 0.09298504
## 57801  0.11500954 0.03194752  0.1516139  0.1876449 0.10699232 0.05579103
## 9636   0.60060539 0.03194752  0.1632765  0.3627802 0.07132822 0.18597009
## 375790 0.03833651 0.11714091  0.1516139  0.2251739 0.14265643 0.05579103
## 375790 0.24279792 0.21298347  0.1516139  0.3002319 0.11888036 0.12088056
## 375790 0.19168257 0.12779008  0.1166261  0.1876449 0.26153679 0.06508953
##        SRR1027184 SRR1027185  SRR1027186 SRR1027187 SRR1027172
## 26155  0.02473931 0.02160652 0.009604733 0.08586956  0.5144759
## 26155  0.08246437 0.07562282 0.028814200 0.17173912  0.2204897
## 26155  0.06597150 0.11883586 0.048023666 0.12021739  0.3674828
## 26155  0.18966806 0.19445869 0.057628399 0.20608695  0.5512242
## 26155  0.04123219 0.19445869 0.115256798 0.08586956  0.4777276
## 57801  0.02473931 0.08642608 0.009604733 0.06869565  0.3307345
## 9636   0.04123219 0.15124564 0.067233132 0.18891303  0.5144759
## 375790 0.09071081 0.20526195 0.163280464 0.17173912  0.3674828
## 375790 0.07421794 0.10803260 0.038418933 0.20608695  0.5144759
## 375790 0.08246437 0.10803260 0.019209466 0.29195651  0.5512242
```

As above, to get a better sense of the data, we assess the mean-variance relationship for each juction. Similarly, we run principal component analysis (PCA) to identify any sample outliers within the data. We assess the variance explained by each of the first 11 PCs as well as visualize the relationship of each sample in the first two PCs.

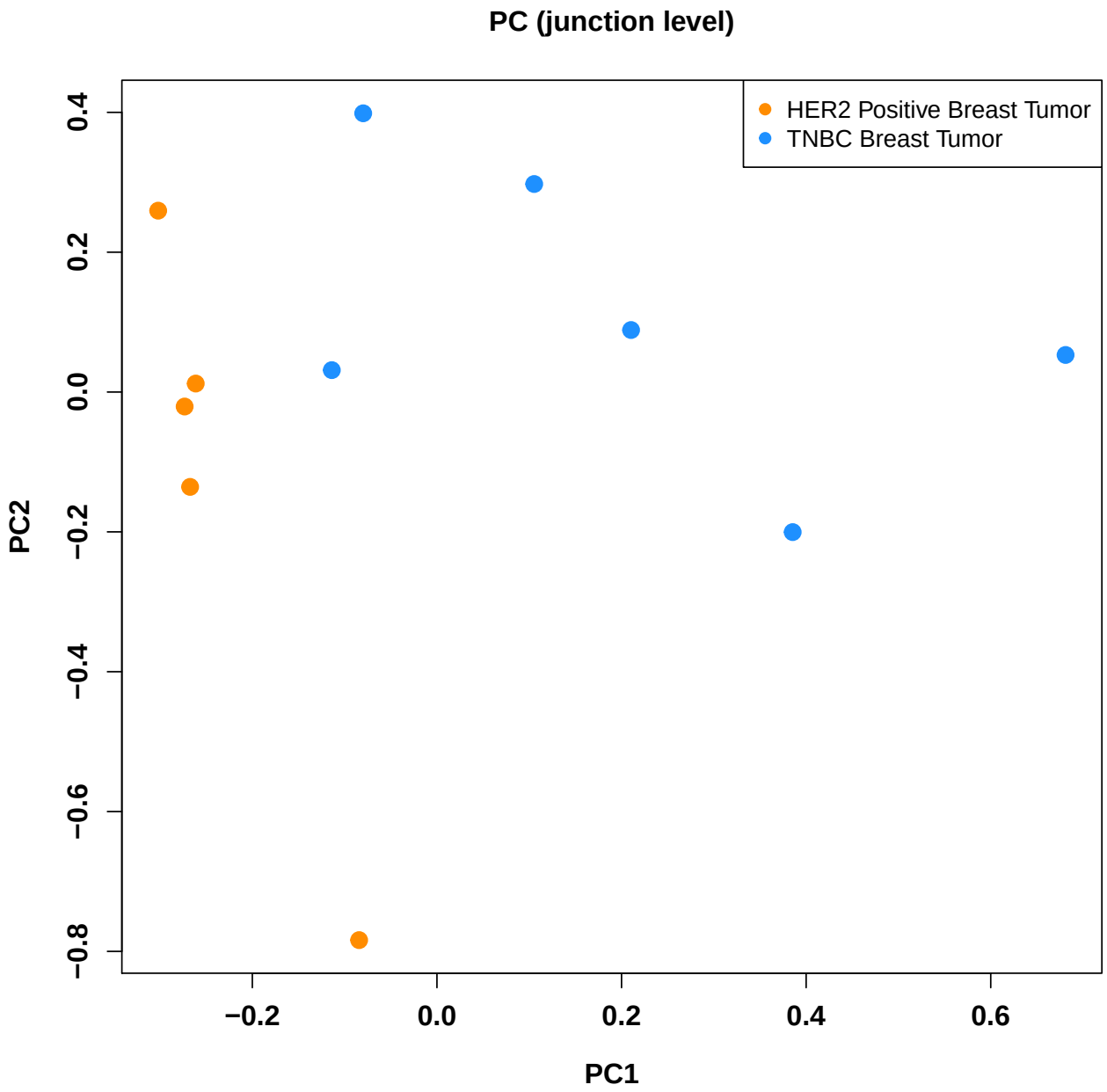
```
## Set colors
trop <- RSkittleBrewer('tropical')[c(1, 2)]
cols <- as.numeric(as.factor(rse$group))

## Look at mean variance relationship
plot(rowMeans(log2(counts + 1)), rowVars(log2(counts + 1)),
     pch = 19, col = trop[2])
```




```
## Calculate PCs with svd function
expr.pca <- svd(counts - rowMeans(counts))

## Plot PCs
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
plot(expr.pca$d^2 / sum(expr.pca$d^2), pch = 19, col = trop[2], cex = 1.5,
     ylab = 'Fraction of variance explained', xlab = 'PC #',
     main = 'PCs (junction level)')
```




```
## Plot PC1 vs. PC2
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
plot(expr.pca$v[, 1], expr.pca$v[, 2], pch = 19, col = trop[cols], cex = 1.5,
     xlab = 'PC1', ylab = 'PC2',
     main = 'PC (junction level)')
legend('topright', pch = 19, col = trop[c(1, 2)],
       names(summary(as.factor(rse$group))), bg="white")
```



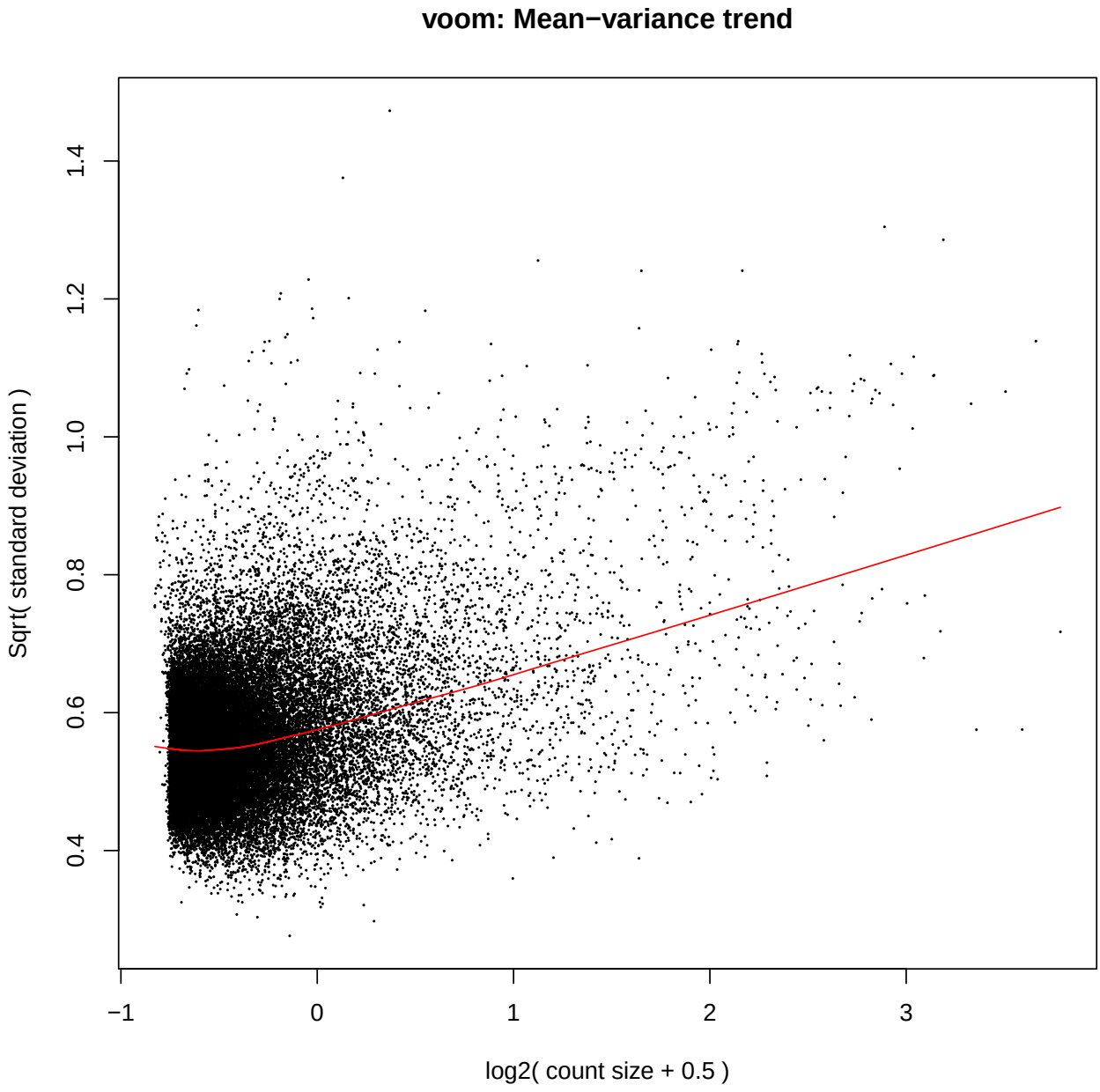
Again, differential expression analysis is carried out using `limma` and `voom`; however, this time at the junction, rather than gene, level. Data are again visualized using a volcano plot to assess the strength [$log_2(fold - change)$ in expression] and its significance [$-log_10(p - value)$] for each junction.

```
design <- model.matrix(~ rse$group)
design
```

```
##    (Intercept) rse$groupTNBC Breast Tumor
## 1            1                          1
## 2            1                          1
## 3            1                          1
## 4            1                          1
## 5            1                          1
## 6            1                          0
## 7            1                          0
## 8            1                          0
```

```
## 9            1                          0
## 10           1                          0
## 11           1                          1
## attr(,"assign")
## [1] 0 1
## attr(,"contrasts")
## attr(,"contrasts")$`rse$group`
## [1] "contr.treatment"
```

```
dge <- DGEList(counts = counts)
dge <- calcNormFactors(dge)
v <- voom(dge, design,plot = TRUE)
```

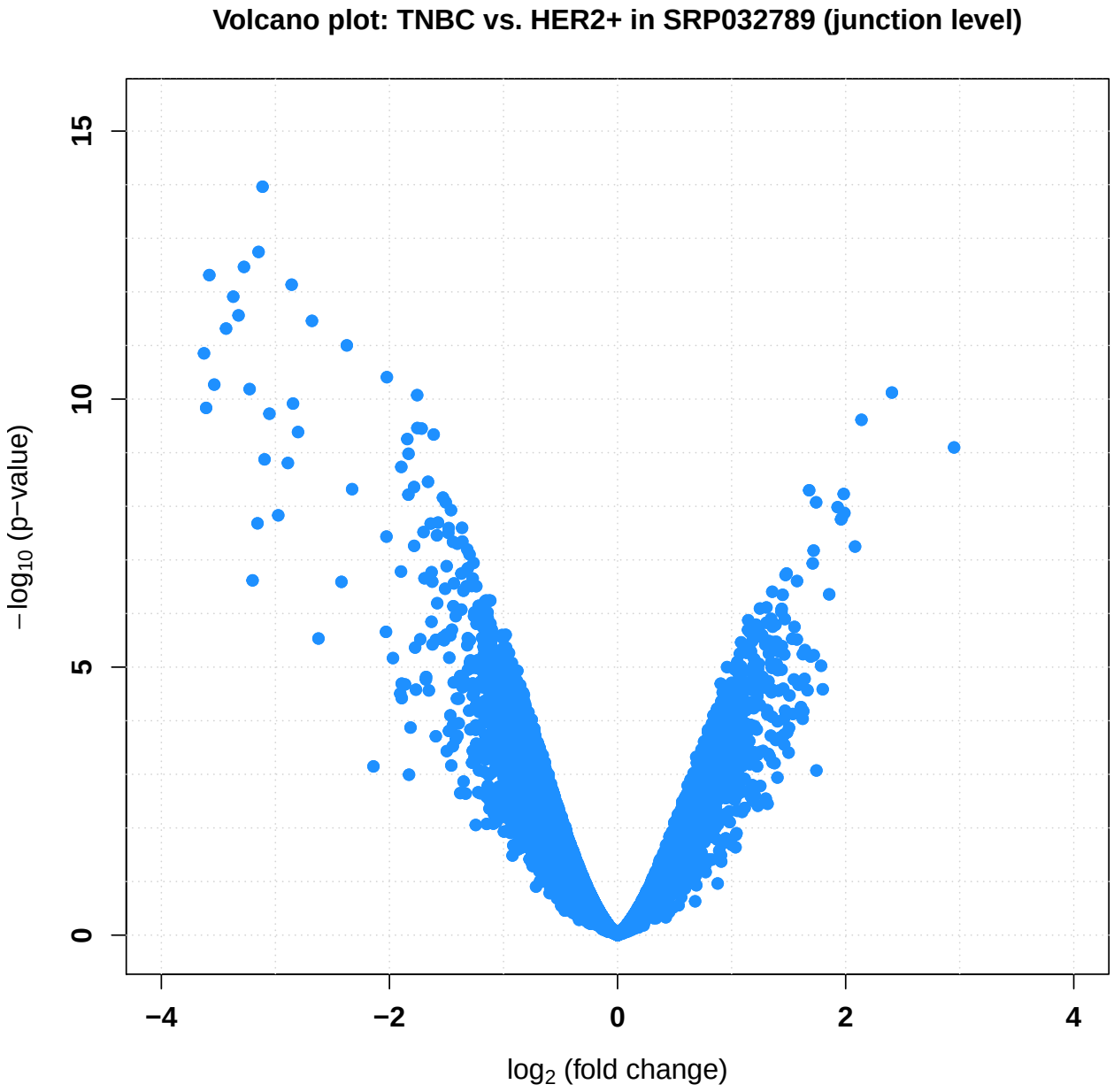



```
fit <- lmFit(v, design)
fit <- eBayes(fit)
log2FC <- fit$coefficients[, 2]
```

```
p.mod <- fit$p.value[, 2]
q.mod <- qvalue(p.mod)$q
res_jx <- data.frame(log2FC, p.mod, q.mod)

## Determine the number of exons differentially expressed at q<0.05
sum(res_jx$q.mod < 0.05)
```

```
## [1] 19805
```

```
## Volcano plot
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
rx2 <- c(-1, 1) * 1.1 * max(abs(log2FC))
ry2 <- c(-0.1, max(-log10(p.mod))) * 1.1
plot(log2FC, -log10(p.mod),
     pch = 19, xlim = rx2, ylim = ry2, col = trop[2],
     xlab = bquote(paste(log[2], ' (fold change)')),
     ylab = bquote(paste(-log[10], ' (p-value)')))
abline(v = seq(-10, 10, 1), col = 'lightgray', lty = 'dotted')
abline(h = seq(0, 2356, 1), col = 'lightgray', lty = 'dotted')
points(log2FC, -log10(p.mod), pch = 19, col = trop[2])
title('Volcano plot: TNBC vs. HER2+ in SRP032789 (junction level)')
```



Comparison of gene, exon, junction, and DER results

To compare findings at the gene, exon, junction, and DER level, we obtained a single exon level [or junction level or DER level] p-value for each gene included at the gene level analysis. To do this, we utilized Simes' rule, such that for each gene included in the gene level analysis, the p-values for exons [or junctions or DERs] within that gene were extracted and sorted. Each exon level [or junction level or DER level] p-value is then multiplied by the number of exons [or junctions or DERs] present within the gene. For each exon [or junction or DER] (1,2...n), this quantity is divided by that exon's rank [or junction's rank or DER's rank] (where 1=most significant exon [or junction or DER] and n=least significant). The minimum value from this calcultion is assigned as the exon level [or junction level or DER level] p-value at each gene. DER results are loaded from the DER analysis report that is described and rendered in `recount_DER_SRP032789.*`

```
## Obtain geneIDs
gene_id <- unique(rownames(counts_exon))

## Calculate p-values for genes with Simes' rule
p_exon_gene <- NULL
for(i in seq_len(length(gene_id))){
    p_exon <- res_exon$p.mod[rownames(counts_exon) %in% gene_id[i]]
    p_exon <- sort(p_exon)
    p_exon_simes <- NULL
    for(j in 1:length(p_exon)){
        p_exon_simes[j] <- length(p_exon) * p_exon[j] / j
    }
    p_exon_gene[i] <- min(p_exon_simes)
}
names(p_exon_gene) <- gene_id

## Determine the number of 'gene level exons' differentially expressed q < 0.05
q_exon_gene <- qvalue(p_exon_gene)$q
sum(q_exon_gene < 0.05)
```

```
## [1] 7935
```

```
## As above, 'topGO' can be utilized to assign biological function to
## differentially expressed exons.

## Gene set analysis (p-values of genes derived with Simes' rule from exon p-values)
interesting <- function(x) x < 0.05

topgoobjBP <- new('topGOdata',
    description = 'biological process',
    ontology = 'BP', allGenes = q_exon_gene, geneSelectionFun = interesting,
    annotationFun = annFUN.org, mapping = 'org.Hs.eg.db', ID = 'entrez')
```

```
##
## Building most specific GOs .....
##  ( 10729 GO terms found. )
##
## Build GO DAG topology ..........
##  ( 14588 GO terms and 34554 relations. )
##
## Annotating nodes ...............
##  ( 13784 genes annotated to the GO terms. )
```

```
bptest <- runTest(topgoobjBP, algorithm = 'weight01', statistic = 'ks')
```

```
##
##           -- Weight01 Algorithm --
##
##       the algorithm is scoring 14588 nontrivial nodes
##       parameters:
##           test statistic: ks
##           score order: increasing
```

```
##
##   Level 20:  1 nodes to be scored    (0 eliminated genes)
##
##   Level 19:  7 nodes to be scored    (0 eliminated genes)
##
##   Level 18:  18 nodes to be scored   (1 eliminated genes)
##
##   Level 17:  41 nodes to be scored   (29 eliminated genes)
##
##   Level 16:  119 nodes to be scored  (82 eliminated genes)
##
##   Level 15:  238 nodes to be scored  (170 eliminated genes)
##
##   Level 14:  477 nodes to be scored  (495 eliminated genes)
##
##   Level 13:  827 nodes to be scored  (1140 eliminated genes)
##
##   Level 12:  1202 nodes to be scored (2246 eliminated genes)
##
##   Level 11:  1541 nodes to be scored (4239 eliminated genes)
##
##   Level 10:  1930 nodes to be scored (5909 eliminated genes)
##
##   Level 9:   2043 nodes to be scored (8114 eliminated genes)
##
##   Level 8:   1940 nodes to be scored (9773 eliminated genes)
##
##   Level 7:   1776 nodes to be scored (10997 eliminated genes)
##
##   Level 6:   1304 nodes to be scored (11911 eliminated genes)
##
##   Level 5:   723 nodes to be scored  (12580 eliminated genes)
##
##   Level 4:   302 nodes to be scored  (13126 eliminated genes)
##
##   Level 3:   77 nodes to be scored   (13351 eliminated genes)
##
##   Level 2:   21 nodes to be scored   (13513 eliminated genes)
##
##   Level 1:   1 nodes to be scored    (13586 eliminated genes)
```

```
bptest
```

```
##
## Description: biological process
```

```
## Ontology: BP
## 'weight01' algorithm with the 'ks' test
## 14588 GO terms scored: 84 terms with p < 0.01
## Annotation data:
##     Annotated genes: 13784
##     Significant genes: 6200
##     Min. no. of genes annotated to a GO: 1
##     Nontrivial nodes: 14588
```

```
bpres_exon <- GenTable(topgoobjBP, pval = bptest,
                       topNodes = length(bptest@score), numChar = 100)
head(bpres_exon, n = 10)
```

```
##         GO.ID                                                          Term
## 1  GO:0051493                 regulation of cytoskeleton organization
## 2  GO:0016569                                    chromatin modification
## 3  GO:0000398                             mRNA splicing, via spliceosome
## 4  GO:0042795      snRNA transcription from RNA polymerase II promoter
## 5  GO:0098609                                        cell-cell adhesion
## 6  GO:0007049                                              cell cycle
## 7  GO:0000381 regulation of alternative mRNA splicing, via spliceosome
## 8  GO:0071363              cellular response to growth factor stimulus
## 9  GO:0016579                                   protein deubiquitination
## 10 GO:0006886                            intracellular protein transport
##    Annotated Significant Expected    pval
## 1        370         177   166.42 1.3e-05
## 2        491         288   220.85 2.5e-05
## 3        277         185   124.59 2.7e-05
## 4         67          44    30.14 3.9e-05
## 5       1024         426   460.59 0.00013
## 6       1546         756   695.39 0.00022
## 7         34          27    15.29 0.00027
## 8        528         252   237.49 0.00045
## 9        107          63    48.13 0.00048
## 10       938         477   421.91 0.00052
```

```
## Obtain geneIDs
gene_id <- unique(rownames(counts_jx))

## Calculate p-values for genes with Simes' rule
p_jx_gene <- NULL
for(i in seq_len(length(gene_id))){
    p_jx <- res_jx$p.mod[rownames(counts_jx) %in% gene_id[i]]
    p_jx <- sort(p_jx)
    p_jx_simes <- NULL
    for(j in 1:length(p_jx)){
        p_jx_simes[j] <- length(p_jx) * p_jx[j] / j
    }
    p_jx_gene[i] <- min(p_jx_simes)
}
names(p_jx_gene) <- gene_id


## Determine the number of 'gene leveljunction' differentially expressed q < 0.05
q_jx_gene <- qvalue(p_jx_gene)$q
```

```
sum(q_jx_gene < 0.05)
```

```
## [1] 4379
```

```
## As above, 'topGO' can be utilized to assign biological function to
## differentially expressed exons.

## Gene set analysis (p-values of genes derived with Simes' rule from junction p-values)
interesting <- function(x) x < 0.05

topgoobjBP <- new('topGOdata',
    description = 'biological process',
    ontology = 'BP', allGenes = q_jx_gene, geneSelectionFun = interesting,
    annotationFun = annFUN.org, mapping = 'org.Hs.eg.db', ID = 'entrez')
```

```
##
## Building most specific GOs .....
##  ( 8047 GO terms found. )
##
## Build GO DAG topology ..........
##  ( 12013 GO terms and 28295 relations. )
##
## Annotating nodes ...............
##  ( 6401 genes annotated to the GO terms. )
```

```
bptest <- runTest(topgoobjBP, algorithm = 'weight01', statistic = 'ks')
```

```
##
##           -- Weight01 Algorithm --
##
##       the algorithm is scoring 12013 nontrivial nodes
##       parameters:
##           test statistic: ks
##           score order: increasing
##
##   Level 20:  1 nodes to be scored    (0 eliminated genes)
##
##   Level 19:  5 nodes to be scored    (0 eliminated genes)
##
##   Level 18:  10 nodes to be scored   (1 eliminated genes)
##
##   Level 17:  22 nodes to be scored   (10 eliminated genes)
##
##   Level 16:  82 nodes to be scored   (31 eliminated genes)
##
##   Level 15:  174 nodes to be scored  (67 eliminated genes)
##
##   Level 14:  359 nodes to be scored  (212 eliminated genes)
```

```
##
##   Level 13:  622 nodes to be scored  (577 eliminated genes)
##
##   Level 12:  926 nodes to be scored  (1173 eliminated genes)
##
##   Level 11:  1208 nodes to be scored (2143 eliminated genes)
##
##   Level 10:  1556 nodes to be scored (3002 eliminated genes)
##
##   Level 9:   1688 nodes to be scored (3944 eliminated genes)
##
##   Level 8:   1649 nodes to be scored (4713 eliminated genes)
##
##   Level 7:   1546 nodes to be scored (5284 eliminated genes)
##
##   Level 6:   1138 nodes to be scored (5688 eliminated genes)
##
##   Level 5:   645 nodes to be scored  (5965 eliminated genes)
##
##   Level 4:   283 nodes to be scored  (6145 eliminated genes)
##
##   Level 3:   77 nodes to be scored   (6237 eliminated genes)
##
##   Level 2:   21 nodes to be scored   (6301 eliminated genes)
##
##   Level 1:   1 nodes to be scored    (6323 eliminated genes)
```

```
bptest
```

```
##
## Description: biological process
## Ontology: BP
## 'weight01' algorithm with the 'ks' test
## 12013 GO terms scored: 47 terms with p < 0.01
## Annotation data:
##     Annotated genes: 6401
##     Significant genes: 4061
##     Min. no. of genes annotated to a GO: 1
##     Nontrivial nodes: 12013
```

```
bpres_jx <- GenTable(topgoobjBP, pval = bptest,
                     topNodes = length(bptest@score), numChar = 100)
head(bpres_jx, n = 10)
```

```
##         GO.ID
## 1  GO:0006614
## 2  GO:0019083
## 3  GO:0000184
## 4  GO:0006413
```

```
## 5  GO:0098609
## 6  GO:0006364
## 7  GO:0042060
## 8  GO:0006446
## 9  GO:0000209
## 10 GO:0038095
##                                                                    Term
## 1          SRP-dependent cotranslational protein targeting to membrane
## 2                                                    viral transcription
## 3  nuclear-transcribed mRNA catabolic process, nonsense-mediated decay
## 4                                              translational initiation
## 5                                                    cell-cell adhesion
## 6                                                       rRNA processing
## 7                                                         wound healing
## 8                                regulation of translational initiation
## 9                                             protein polyubiquitination
## 10                                Fc-epsilon receptor signaling pathway
##    Annotated Significant Expected    pval
## 1         76          64    48.22 1.4e-14
## 2        131         102    83.11 4.5e-13
## 3         93          79    59.00 2.8e-12
## 4        150         125    95.16 5.4e-12
## 5        524         376   332.44 1.8e-07
## 6        191         129   121.18 5.2e-06
## 7        234         166   148.46 8.8e-05
## 8         65          53    41.24 0.00013
## 9        168         118   106.58 0.00017
## 10        92          73    58.37 0.00021
```

```
## Load p-values from DER anaysis
load('AnnotatedDERs.Rdata')
p.mod <- annotatedDERs

## Obtain geneIDs
gene_id <- unique(names(p.mod))

## Calculate p-values for genes with Simes' rule
p_DER_gene <- NULL
for(i in seq_len(length(gene_id))){
    p_DER <- p.mod[names(p.mod) %in% gene_id[i]]
    p_DER <- sort(p_DER)
    p_DER_simes <- NULL
    for(j in 1:length(p_DER)){
        p_DER_simes[j] <- length(p_DER) * p_DER[j] / j
    }
    p_DER_gene[i] <- min(p_DER_simes)
}
names(p_DER_gene) <- gene_id

## Determine the number of 'gene level DERs' differentially expressed q < 0.05
q_DER_gene <- qvalue(p_DER_gene)$q
sum(q_DER_gene < 0.05)
```

```
## [1] 6938
```

```
## As above, 'topGO' can be utilized to assign biological function to
## differentially expressed DERs.

## Gene set analysis (p-values of genes derived with Simes' rule from DER p-values)
interesting <- function(x) x < 0.05

topgoobjBP <- new('topGOdata',
    description = 'biological process',
    ontology = 'BP', allGenes = q_DER_gene, geneSelectionFun = interesting,
    annotationFun = annFUN.org, mapping = 'org.Hs.eg.db', ID = 'entrez')
```

```
##
## Building most specific GOs .....
##  ( 10056 GO terms found. )
##
## Build GO DAG topology ..........
##  ( 13984 GO terms and 33121 relations. )
##
## Annotating nodes ...............
##  ( 11146 genes annotated to the GO terms. )
```

```
bptest <- runTest(topgoobjBP, algorithm = 'weight01', statistic = 'ks')
```

```
##
##           -- Weight01 Algorithm --
##
##       the algorithm is scoring 13984 nontrivial nodes
##       parameters:
##           test statistic: ks
##           score order: increasing
##
##   Level 20:  1 nodes to be scored    (0 eliminated genes)
##
##   Level 19:  6 nodes to be scored    (0 eliminated genes)
##
##   Level 18:  16 nodes to be scored   (1 eliminated genes)
##
##   Level 17:  41 nodes to be scored   (19 eliminated genes)
##
##   Level 16:  110 nodes to be scored  (55 eliminated genes)
##
##   Level 15:  227 nodes to be scored  (130 eliminated genes)
##
##   Level 14:  454 nodes to be scored  (396 eliminated genes)
##
##   Level 13:  785 nodes to be scored  (949 eliminated genes)
##
```

```
##   Level 12:  1157 nodes to be scored (1902 eliminated genes)
##
##   Level 11:  1474 nodes to be scored (3574 eliminated genes)
##
##   Level 10:  1831 nodes to be scored (4986 eliminated genes)
##
##   Level 9:   1958 nodes to be scored (6767 eliminated genes)
##
##   Level 8:   1856 nodes to be scored (8099 eliminated genes)
##
##   Level 7:   1710 nodes to be scored (9064 eliminated genes)
##
##   Level 6:   1262 nodes to be scored (9777 eliminated genes)
##
##   Level 5:   700 nodes to be scored  (10299 eliminated genes)
##
##   Level 4:   298 nodes to be scored  (10647 eliminated genes)
##
##   Level 3:   76 nodes to be scored   (10803 eliminated genes)
##
##   Level 2:   21 nodes to be scored   (10937 eliminated genes)
##
##   Level 1:   1 nodes to be scored    (10992 eliminated genes)
```

```
bptest
```

```
##
## Description: biological process
## Ontology: BP
## 'weight01' algorithm with the 'ks' test
## 13984 GO terms scored: 55 terms with p < 0.01
## Annotation data:
##     Annotated genes: 11146
##     Significant genes: 5889
##     Min. no. of genes annotated to a GO: 1
##     Nontrivial nodes: 13984
```

```
bpres_DER <- GenTable(topgoobjBP, pval = bptest,
                      topNodes = length(bptest@score), numChar = 100)
head(bpres_DER, n = 10)
```

```
##         GO.ID
## 1  GO:0000398
## 2  GO:0051493
## 3  GO:0000381
## 4  GO:0031124
## 5  GO:0098609
## 6  GO:0090503
## 7  GO:0010606
## 8  GO:0042795
```

```
## 9  GO:0006351
## 10 GO:0006338
##                                                                    Term
## 1                                    mRNA splicing, via spliceosome
## 2                             regulation of cytoskeleton organization
## 3          regulation of alternative mRNA splicing, via spliceosome
## 4                                           mRNA 3'-end processing
## 5                                               cell-cell adhesion
## 6                RNA phosphodiester bond hydrolysis, exonucleolytic
## 7  positive regulation of cytoplasmic mRNA processing body assembly
## 8               snRNA transcription from RNA polymerase II promoter
## 9                                  transcription, DNA-templated
## 10                                       chromatin remodeling
##    Annotated Significant Expected    pval
## 1        261         191   137.90 4.4e-05
## 2        328         192   173.30 0.00032
## 3         29          25    15.32 0.00039
## 4         76          56    40.15 0.00042
## 5        837         434   442.23 0.00053
## 6         33          23    17.44 0.00098
## 7          6           6     3.17 0.00098
## 8         66          44    34.87 0.00099
## 9       2731        1489  1442.93 0.00117
## 10       136          75    71.86 0.00126
```

To determine the concordance between the gene level and (exon, junction, DER) level analyses, the top hits (as determined by p-value) are compared. Results are plotted such that the points falling along the identity line would indicate complete agreement between the top hits of each analysis.

```
## Set colors
trop <- RSkittleBrewer('tropical')[c(1, 2, 3)]

## Obtain and sort p-values for genes
p.mod1 <- res_gene$p.mod
names(p.mod1) <- rownames(res_gene)
p.mod1.sort <- p.mod1[order(p.mod1)]

## Obtain and sort p-values for genes derived from exons
p.mod2 <- p_exon_gene
p.mod2.sort <- p.mod2[order(p.mod2)]

## Obtain and sort p-values for genes derived from junctions
p.mod3 <- p_jx_gene
p.mod3.sort <- p.mod3[order(p.mod3)]

## Obtain and sort p-values for genes derived from DER
p.mod4 <- p_DER_gene
p.mod4.sort <- p.mod4[order(p.mod4)]


## Overlap of features:
    ## gene level and exon level
table(names(p.mod1.sort) %in% names(p.mod2.sort))
```

```
##
```

```
##  TRUE
## 17874
```

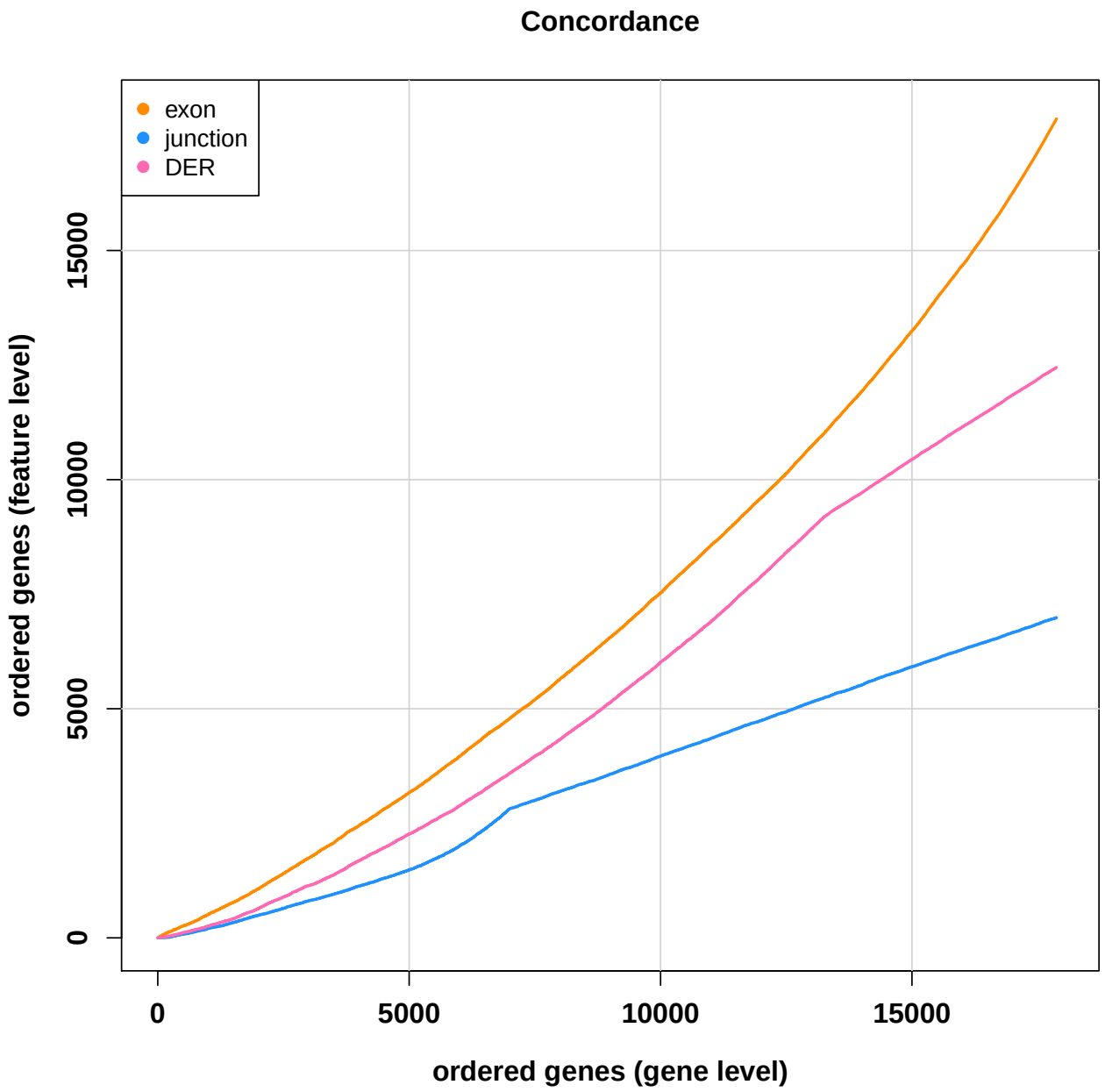
```
    ## gene level and junction level
table(names(p.mod1.sort) %in% names(p.mod3.sort))
```

```
##
## FALSE  TRUE
## 10885  6989
```

```
    ## gene level and DER level
table(names(p.mod1.sort) %in% names(p.mod4.sort))
```

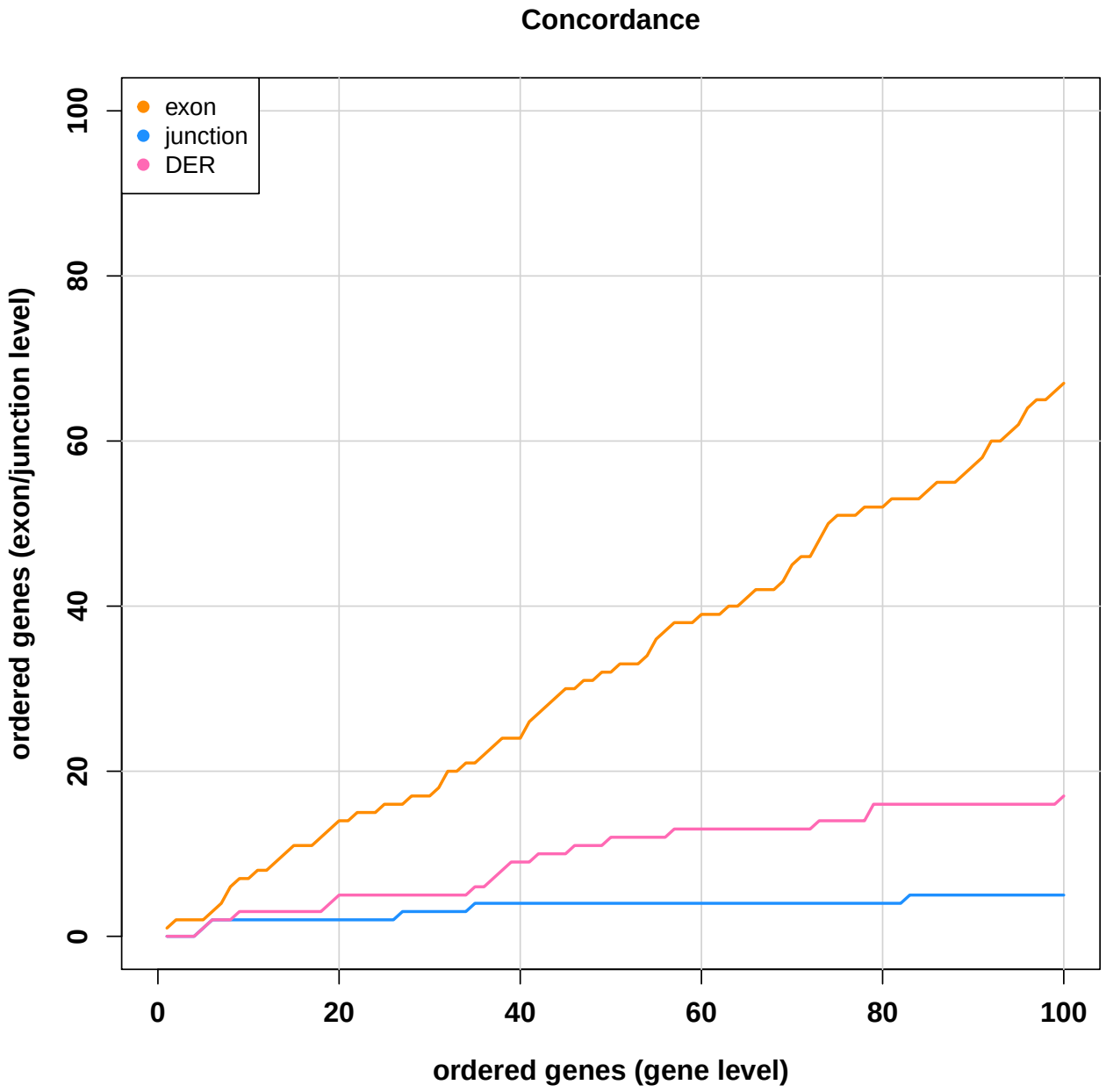
```
##
## FALSE  TRUE
##  5425 12449
```

```
conc_exon <- NULL
conc_jx <- NULL
conc_DER <- NULL
for(i in seq_len(length(p.mod1.sort))) {
    conc_exon[i] <- sum(names(p.mod1.sort)[1:i] %in% names(p.mod2.sort)[1:i])
    conc_jx[i] <- sum(names(p.mod1.sort)[1:i] %in% names(p.mod3.sort)[1:i])
    conc_DER[i] <- sum(names(p.mod1.sort)[1:i] %in% names(p.mod4.sort)[1:i])
}


## All genes
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
plot(seq(1:length(p.mod1.sort)), conc_exon,
     type = 'l', las = 0,
     xlim = c(0, 18000),
     ylim = c(0, 18000),
     xlab = 'ordered genes (gene level)',
     ylab = 'ordered genes (feature level)',
     main = 'Concordance')
for(k in 1:3){
    abline(v = k * 5000, cex = 0.5, col = 'lightgrey')
    abline(h = k * 5000, cex = 0.5, col = 'lightgrey')
}
points(seq(1:length(p.mod1.sort)), conc_jx, type = 'l', lwd = 2, col = trop[2])
lines(seq(1:length(p.mod1.sort)), conc_exon, lwd = 2,  col = trop[1])
lines(seq(1:length(p.mod1.sort)), conc_DER, lwd = 2,  col = trop[3])
legend('topleft', pch = 19, col = trop[c(1, 2, 3)], c("exon", "junction", "DER"), bg="white")
```



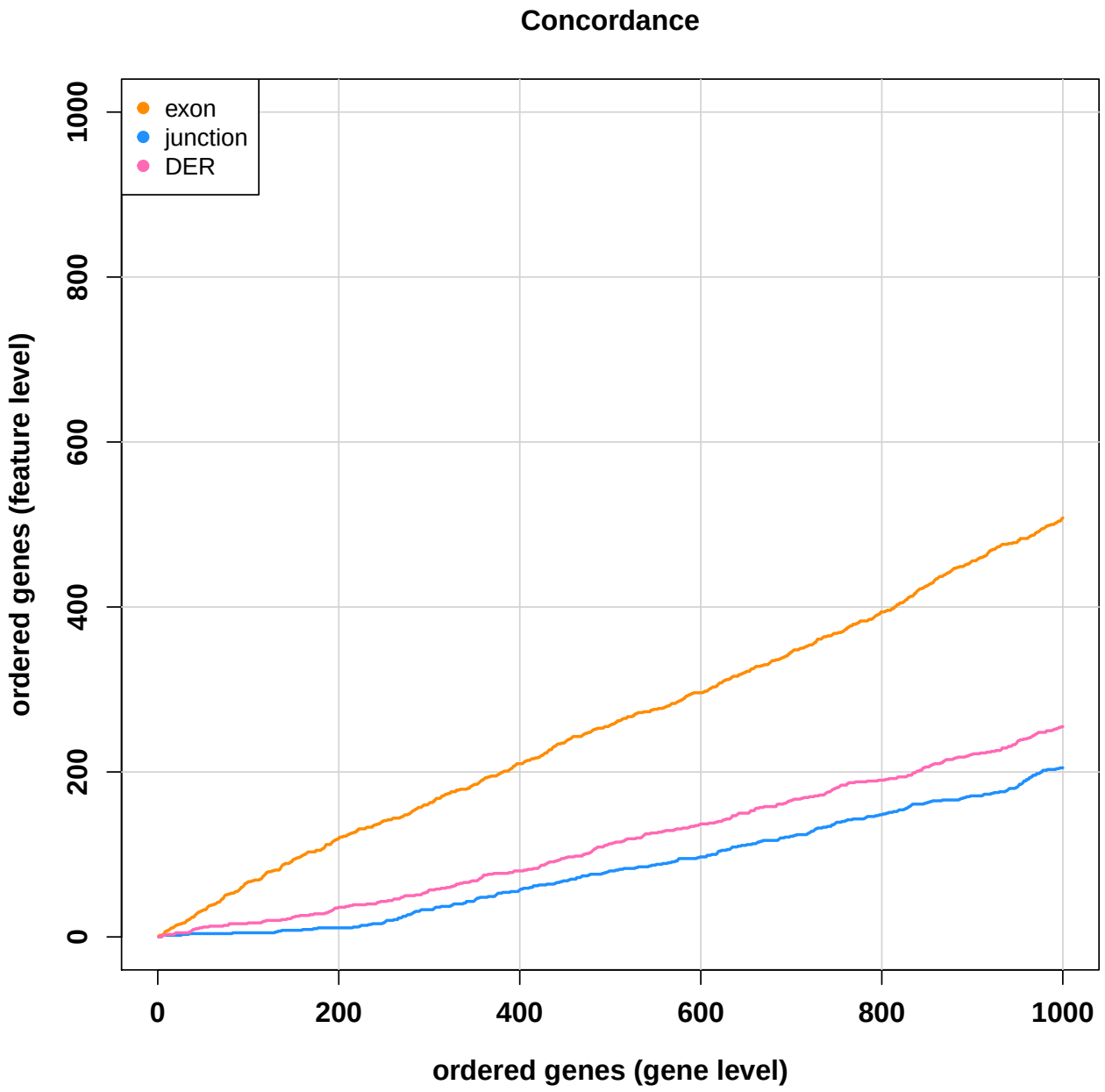

```
## Top 100 genes
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
plot(seq(1:length(p.mod1.sort[1:100])), conc_exon[1:100],
     type = 'l',
     xlim = c(0, 100),
     ylim = c(0, 100),
     xlab = 'ordered genes (gene level)',
     ylab = 'ordered genes (exon/junction level)',
     main = 'Concordance')
for(k in 1:5){
    abline(v = k * 20, cex = 0.5, col = 'lightgrey')
    abline(h = k * 20, cex = 0.5, col = 'lightgrey')
}
points(seq(1:length(p.mod1.sort[1:100])), conc_jx[1:100], type = 'l', lwd = 2, col = trop[2])
lines(seq(1:length(p.mod1.sort[1:100])), conc_exon[1:100], lwd = 2,  col = trop[1])
```

```
lines(seq(1:length(p.mod1.sort[1:100])), conc_DER[1:100], lwd = 2,  col = trop[3])
legend('topleft', pch = 19, col = trop[c(1, 2, 3)], c("exon", "junction", "DER"), bg="white")
```




```
## Top 1,000 genes
par(font.lab = 2, cex.lab = 1.2, font.axis = 2, cex.axis = 1.2)
plot(seq(1:length(p.mod1.sort[1:1000])), conc_exon[1:1000],
     type = 'l',
     xlim = c(0, 1000),
     ylim = c(0, 1000),
     xlab = 'ordered genes (gene level)',
     ylab = 'ordered genes (feature level)',
     main = 'Concordance')
for(k in 1:5){
    abline(v = k * 200, cex = 0.5, col = 'lightgrey')
```

```
    abline(h = k * 200, cex = 0.5, col = 'lightgrey')
}
points(seq(1:length(p.mod1.sort[1:1000])), conc_jx[1:1000], type = 'l', lwd = 2, col = trop[2])
lines(seq(1:length(p.mod1.sort[1:1000])), conc_exon[1:1000], lwd = 2,  col = trop[1] )
lines(seq(1:length(p.mod1.sort[1:1000])), conc_DER[1:1000], lwd = 2,  col = trop[3])
legend('topleft', pch = 19, col = trop[c(1, 2, 3)], c("exon", "junction", "DER"), bg="white")
```



Concordance can also be calculated looking at the gene ontology (GO) groups identified from the gene and exon level analyses. Again, we plot the agreement between the two analyses such that complete agreement between the two analyses would fall along the identity line.

Reproducibility

This analysis report was made possible thanks to:

- R (R Core Team, 2016)
- *BiocStyle* (Oleś, Morgan, and Huber, 2016)
- *derfinder* (Collado-Torres, Nellore, Frazee, Wilks, et al., 2016)
- *devtools* (Wickham and Chang, 2016)
- *edgeR* (Robinson, McCarthy, and Smyth, 2010)
- *knitcitations* (Boettiger, 2015)
- *matrixStats* (Bengtsson, 2016)
- *qvalue* (with contributions from Andrew J. Bass, Dabney, and Robinson, 2015)
- *recount* (Collado-Torres, Nellore, Kammers, Ellis, et al., 2016)
- *rmarkdown* (Allaire, Cheng, Xie, McPherson, et al., 2016)
- *RSkittleBrewer* (Frazee, 2016)
- *SummarizedExperiment* (Morgan, Obenchain, Hester, and Pagès, 2016)
- *topGO* (Alexa and Rahnenfuhrer, 2016)
- *limma* (Law, Chen, Shi, and Smyth, 2014)

Bibliography file

[1] A. Alexa and J. Rahnenfuhrer. topGO: Enrichment Analysis for Gene Ontology. R package version 2.26.0. 2016.

[2] J. Allaire, J. Cheng, Y. Xie, J. McPherson, et al. rmarkdown: Dynamic Documents for R. R package version 1.2. 2016. URL: https://CRAN.R-project.org/package=rmarkdown.

[3] J. D. S. with contributions from Andrew J. Bass, A. Dabney and D. Robinson. qvalue: Q-value estimation for false discovery rate control. R package version 2.6.0. 2015. URL: http://github.com/jdstorey/qvalue.

[4] H. Bengtsson. matrixStats: Functions that Apply to Rows and Columns of Matrices (and to Vectors). R package version 0.51.0. 2016. URL: https://CRAN.R-project.org/package=matrixStats.

[5] C. Boettiger. knitcitations: Citations for 'Knitr' Markdown Files. R package version 1.0.7. 2015. URL: https://CRAN.R-project.org/package=knitcitations.

[6] L. Collado-Torres, A. Nellore, A. C. Frazee, C. Wilks, et al. "Flexible expressed region analysis for RNA-seq with derfinder". In: Nucl. Acids Res. (2016). DOI: 10.1093/nar/gkw852. URL: http://nar.oxfordjournals.org/content/early/2016/09/29/nar.gkw852.

[7] L. Collado-Torres, A. Nellore, K. Kammers, S. E. Ellis, et al. "recount: A large-scale resource of analysis-ready RNA-seq expression data". In: bioRxiv (2016). DOI: 10.1101/068478. URL: http://biorxiv.org/content/early/2016/08/08/068478.

[8] A. Frazee. RSkittleBrewer: Fun with R Colors. R package version 1.1. 2016. URL: https://github.com/alyssafrazee/RSkittlel

[9] C. Law, Y. Chen, W. Shi and G. Smyth. "Voom: precision weights unlock linear model analysis tools for RNA-seq read counts". In: Genome Biology 15 (2014), p. R29.

[10] M. Morgan, V. Obenchain, J. Hester and H. Pagès. SummarizedExperiment: SummarizedExperiment container. R package version 1.4.0. 2016.

[11] A. Oleś, M. Morgan and W. Huber. BiocStyle: Standard styles for vignettes and other Bioconductor documents. R package version 2.2.1. 2016. URL: https://github.com/Bioconductor/BiocStyle.

[12] R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing. Vienna, Austria, 2016. URL: https://www.R-project.org/.

[13] M. D. Robinson, D. J. McCarthy and G. K. Smyth. "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data". In: Bioinformatics 26 (2010), pp. -1.

[14] H. Wickham and W. Chang. devtools: Tools to Make Developing R Packages Easier. R package version 1.12.0. 2016. URL: https://CRAN.R-project.org/package=devtools.

```
## Time spent creating this report:
diff(c(timestart, Sys.time()))
```

```
## Time difference of 24.35083 mins
```

```
## Date this report was generated
message(Sys.time())
```

```
## 2016-11-30 17:12:01
```

```
## Reproducibility info
options(width = 120)
devtools::session_info()
```

```
## Session info -------------------------------------------------------------------------------------
```

```
##  setting  value
##  version  R version 3.3.2 RC (2016-10-26 r71594)
##  system   x86_64, darwin13.4.0
##  ui       X11
##  language (EN)
##  collate  en_US.UTF-8
##  tz       America/New_York
##  date     2016-11-30
```

```
## Packages -----------------------------------------------------------------------------------------
```

```
##  package              * version  date       source
##  acepack                1.4.1    2016-10-29 CRAN (R 3.3.0)
##  AnnotationDbi        * 1.36.0   2016-10-18 Bioconductor
##  assertthat             0.1      2013-12-06 cran (@0.1)
##  backports              1.0.4    2016-10-24 CRAN (R 3.3.0)
##  bibtex                 0.4.0    2014-12-31 CRAN (R 3.3.0)
##  Biobase              * 2.34.0   2016-10-18 Bioconductor
##  BiocGenerics         * 0.20.0   2016-10-18 Bioconductor
##  BiocParallel           1.8.1    2016-10-30 Bioconductor
##  BiocStyle            * 2.2.1    2016-11-24 Bioconductor
##  biomaRt                2.30.0   2016-10-18 Bioconductor
##  Biostrings             2.42.0   2016-10-18 Bioconductor
##  bitops                 1.0-6    2013-08-17 cran (@1.0-6)
##  BSgenome               1.42.0   2016-10-18 Bioconductor
##  bumphunter             1.14.0   2016-10-18 Bioconductor
##  cluster                2.0.5    2016-10-08 CRAN (R 3.3.2)
##  codetools              0.2-15   2016-10-05 CRAN (R 3.3.2)
##  colorout             * 1.1-2    2016-10-19 Github (jalvesaq/colorout@6d84420)
##  colorspace             1.3-1    2016-11-18 CRAN (R 3.3.2)
##  data.table             1.9.8    2016-11-25 CRAN (R 3.3.2)
##  DBI                    0.5-1    2016-09-10 cran (@0.5-1)
##  derfinder            * 1.8.0    2016-10-18 Bioconductor
##  derfinderHelper        1.8.0    2016-10-18 Bioconductor
##  devtools               1.12.0   2016-06-24 CRAN (R 3.3.0)
##  digest                 0.6.10   2016-08-02 CRAN (R 3.3.0)
##  doRNG                  1.6      2014-03-07 CRAN (R 3.3.0)
##  downloader             0.4      2015-07-09 CRAN (R 3.3.0)
##  edgeR                * 3.16.4   2016-11-27 Bioconductor
```

```
##  evaluate              0.10     2016-10-11 cran (@0.10)
##  foreach               1.4.3    2015-10-13 CRAN (R 3.3.0)
##  foreign               0.8-67   2016-09-13 CRAN (R 3.3.2)
##  Formula               1.2-1    2015-04-07 CRAN (R 3.3.0)
##  GenomeInfoDb        * 1.10.1   2016-11-04 Bioconductor
##  GenomicAlignments     1.10.0   2016-10-18 Bioconductor
##  GenomicFeatures       1.26.0   2016-10-18 Bioconductor
##  GenomicFiles          1.10.3   2016-10-21 Bioconductor
##  GenomicRanges       * 1.26.1   2016-10-20 Bioconductor
##  GEOquery              2.40.0   2016-10-18 Bioconductor
##  ggplot2               2.2.0    2016-11-11 CRAN (R 3.3.2)
##  GO.db               * 3.4.0    2016-10-19 Bioconductor
##  graph               * 1.52.0   2016-10-18 Bioconductor
##  gridExtra             2.2.1    2016-02-29 CRAN (R 3.3.0)
##  gtable                0.2.0    2016-02-26 CRAN (R 3.3.0)
##  Hmisc                 4.0-0    2016-11-01 CRAN (R 3.3.0)
##  htmlTable             1.7      2016-10-19 CRAN (R 3.3.0)
##  htmltools             0.3.5    2016-03-21 cran (@0.3.5)
##  httr                  1.2.1    2016-07-03 CRAN (R 3.3.0)
##  IRanges             * 2.8.1    2016-11-08 Bioconductor
##  iterators             1.0.8    2015-10-13 CRAN (R 3.3.0)
##  jsonlite              1.1      2016-09-14 CRAN (R 3.3.0)
##  knitcitations       * 1.0.7    2015-10-28 CRAN (R 3.3.0)
##  knitr                 1.15.1   2016-11-22 CRAN (R 3.3.2)
##  lattice               0.20-34  2016-09-06 CRAN (R 3.3.2)
##  latticeExtra          0.6-28   2016-02-09 CRAN (R 3.3.0)
##  lazyeval              0.2.0    2016-06-12 cran (@0.2.0)
##  limma               * 3.30.6   2016-11-29 Bioconductor
##  locfit                1.5-9.1  2013-04-20 CRAN (R 3.3.0)
##  lubridate             1.6.0    2016-09-13 CRAN (R 3.3.0)
##  magrittr              1.5      2014-11-22 cran (@1.5)
##  Matrix                1.2-7.1  2016-09-01 CRAN (R 3.3.2)
##  matrixStats         * 0.51.0   2016-10-09 CRAN (R 3.3.0)
##  memoise               1.0.0    2016-01-29 CRAN (R 3.3.0)
##  munsell               0.4.3    2016-02-13 cran (@0.4.3)
##  nnet                  7.3-12   2016-02-02 CRAN (R 3.3.2)
##  org.Hs.eg.db        * 3.4.0    2016-10-19 Bioconductor
##  pkgmaker              0.22     2014-05-14 CRAN (R 3.3.0)
##  plyr                  1.8.4    2016-06-08 cran (@1.8.4)
##  qvalue              * 2.6.0    2016-10-18 Bioconductor
##  R6                    2.2.0    2016-10-05 CRAN (R 3.3.0)
##  RColorBrewer          1.1-2    2014-12-07 cran (@1.1-2)
##  Rcpp                  0.12.8   2016-11-17 CRAN (R 3.3.2)
##  RCurl                 1.95-4.8 2016-03-01 cran (@1.95-4.)
##  recount             * 1.0.3    2016-11-27 Bioconductor
##  RefManageR            0.13.1   2016-11-13 CRAN (R 3.3.2)
##  registry              0.3      2015-07-08 CRAN (R 3.3.0)
##  rentrez               1.0.4    2016-10-26 CRAN (R 3.3.0)
##  reshape2              1.4.2    2016-10-22 CRAN (R 3.3.0)
##  RJSONIO               1.3-0    2014-07-28 cran (@1.3-0)
##  rmarkdown           * 1.2      2016-11-21 CRAN (R 3.3.2)
##  rngtools              1.2.4    2014-03-06 CRAN (R 3.3.0)
##  rpart                 4.1-10   2015-06-29 CRAN (R 3.3.2)
##  rprojroot             1.1      2016-10-29 CRAN (R 3.3.0)
```

```
##  Rsamtools              1.26.1   2016-10-22 Bioconductor
##  RSkittleBrewer       * 1.1      2016-10-19 Github (alyssafrazee/RSkittleBrewer@0088112)
##  RSQLite                1.1      2016-11-27 CRAN (R 3.3.1)
##  rtracklayer            1.34.1   2016-11-02 Bioconductor
##  S4Vectors            * 0.12.0   2016-10-18 Bioconductor
##  scales                 0.4.1    2016-11-09 CRAN (R 3.3.2)
##  SparseM              * 1.74     2016-11-10 CRAN (R 3.3.2)
##  stringi                1.1.2    2016-10-01 cran (@1.1.2)
##  stringr                1.1.0    2016-08-19 cran (@1.1.0)
##  SummarizedExperiment * 1.4.0    2016-10-18 Bioconductor
##  survival               2.40-1   2016-10-30 CRAN (R 3.3.0)
##  tibble                 1.2      2016-08-26 cran (@1.2)
##  topGO                * 2.26.0   2016-10-18 Bioconductor
##  VariantAnnotation      1.20.1   2016-11-14 Bioconductor
##  withr                  1.0.2    2016-06-20 CRAN (R 3.3.0)
##  XML                    3.98-1.5 2016-11-10 CRAN (R 3.3.2)
##  xtable                 1.8-2    2016-02-05 CRAN (R 3.3.0)
##  XVector                0.14.0   2016-10-18 Bioconductor
##  yaml                   2.1.14   2016-11-12 CRAN (R 3.3.2)
##  zlibbioc               1.20.0   2016-10-18 Bioconductor
```