A pipeline of functional analysis for CRISPR screen data

MAGeCKFlute package

MAGeCKFlute version 0.1.0

**Abstract**

The purpose to do CRISPR screen is to filter essential genes, pathways and explain the background mechanisms. We developed a pipeline to do the downstream analysis with the output of MAGeCK. Our pipeline includes several functions, including cell cycle normalization, KEGG enrichment, GO analysis and pathway visualization.

Contents

[1 Input data and preparations 2](#_Toc488422028)

[1.1 The beta score table 2](#_Toc488422029)

[1.2 Normalization of beta scores 4](#_Toc488422030)

[2 Distribution of all gene beta scores 4](#_Toc488422031)

[2.1 Violin plot and density plot 4](#_Toc488422032)

[2.2 MAplot 6](#_Toc488422033)

[2.3 Estimate the cell cycle time by linear fitting 6](#_Toc488422034)

[3 Positive selection and negative selection 8](#_Toc488422035)

[3.1 Group all genes into subgroups 8](#_Toc488422036)

[3.2 Rank the beta score differences between control and treatment samples 9](#_Toc488422037)

[3.3 Functional enrichment analysis 9](#_Toc488422038)

[3.4 Visualize the significantly enriched pathway map 10](#_Toc488422039)

[4 Select drug related genes 11](#_Toc488422040)

[4.1 Select four group meaningful genes using 9-square model 11](#_Toc488422041)

[4.2 BP and KEGG enrichment analysis 12](#_Toc488422042)

[4.3 Pathway visualization 13](#_Toc488422043)

[5 Run pipeline with MAGeCK-RRA results 16](#_Toc488422044)

[5.1 Gene summary file in MAGeCK-RRA results 16](#_Toc488422045)

[5.2 Negative selection and positive selection 17](#_Toc488422046)

[6 Session Info 19](#_Toc488422047)

Quick start

To run MAGeCKFlute pipeline, we need gene summary file generated by MAGeCK MLE or MAGeCK RRA. MAGeCKFlute package provides two example data from Feng Zhang’s paper, one is ‘BRAF\_mle.gene\_sunmmary’ and the other is ‘BRAF\_rra.gene\_summary’. We will work with them in this documents.

The pipeline for MAGeCK MLE results is as follows .

> library(MAGeCKFlute)

> ##Load MLE gene summary

> data("BRAF\_mle.gene\_summary")

> ##Run the MAGeCKFlute MLE pipeline

> FluteMLE(gene\_summary= BRAF\_mle.gene\_summary, ctrlname=c("D7\_R1", "D7\_R2"), treatname=c("PLX7\_R1","PLX7\_R2"), prefix="BRAF", workspace=".")

All pipeline results are written into local directory ‘./BRAF\_Flute\_Results/’, and all figures are integrated into file ‘BRAF\_Flute.mle\_summary.pdf’.

The pipeline for MAGeCK RRA results is as follows .

> ##Load RRA gene summary

> data("BRAF\_rra.gene\_summary")

> ##Run the MAGeCKFlute RRA pipeline

> FluteRRA(BRAF\_rra.gene\_summary, prefix="BRAF")

All pipeline results are written into local directory ‘./BRAF\_Flute\_Results/’ too, and all figures are integrated into file ‘BRAF\_Flute.rra\_summary.pdf’.

1 Input data and preparations

1.1 The beta score table

As the input of MAGeCKFlute package, the gene summary file from MAGeCK-MLE includes beta scores of all genes in control and treatment samples. The “beta score” is similar to the value of log2-transformed fold-change in differential expression analysis.

Use function ‘data’ to load the dataset, and have a look at the file with a text editor to see how it is formatted. Then, extract beta scores of control and treatment from the gene summary data matrix, and finally have a look at the beta score matrix.

> library(MAGeCKFlute)

> data("BRAF\_mle.gene\_summary")

> gene\_summary=BRAF\_mle.gene\_summary

> gene\_summary[1:6,1:8]

Gene sgRNA D7\_R1.beta D7\_R1.z D7\_R1.p.value D7\_R1.fdr D7\_R1.wald.p.value D7\_R1.wald.fdr

1 SMPD2 3 -0.458770 -3.50370 0.202310 0.83971 4.5880e-04 2.1478e-03

2 MIXL1 4 0.094571 1.01150 0.352670 0.90949 3.1179e-01 4.5218e-01

3 ENPEP 4 -0.082621 -0.87650 0.911990 0.99475 3.8076e-01 5.2290e-01

4 VAV2 3 -0.024270 -0.26885 0.710060 0.97590 7.8804e-01 8.6093e-01

5 C5orf24 2 0.452310 3.83170 0.014501 0.41342 1.2726e-04 7.0174e-04

6 TLCD1 5 -0.883320 -10.62600 0.018899 0.44840 2.2600e-26 1.8900e-24

> ctrlName=c("D7\_R1", "D7\_R2")

> treatName=c("PLX7\_R1", "PLX7\_R2")

> #Read beta scores from gene summary matrix

> dd=ReadBeta(gene\_summary, ctrlName=ctrlName, treatName=treatName)

> head(dd)

Gene Control Treatment ENTREZID

1 A1BG -0.1230650 -0.0007493 1

2 A1CF 0.0976245 0.0543575 29974

3 A2M 0.1292900 0.2057300 2

4 A2ML1 0.1704850 0.2465200 144568

5 A4GALT 0.2346750 0.4057950 53947

6 A4GNT -0.2485200 -0.3280750 51146

1.2 Normalization of beta scores

It is difficult to control all samples with a consistent cell cycle in an CRISPR screen experiment with multi conditions. Besides, beta score among different conditions with an inconsistent cell cycle are incomparable. So it is necessary to do the normalization when comparing the beta scores in different conditions. Essential genes are those genes that are indispensable for its survival. The effect generated by knocking out these genes in different cell types is consistent. Based on this, we developed cell cycle normalization method to shorten the gap of cell cycle in different conditions. In addition, a previous normalization method called loess normalization is available in this package. These two normalization methods are optional.

> dd\_essential = NormalizeBeta(dd, method="cell\_cycle")

2 Distribution of all gene beta scores

2.1 Violin plot and density plot

After normalization, the distribution of beta scores in different conditions should be similar. We can evaluate the distribution of beta scores using function ‘Violin.plot’, ‘Density.plot’, and ‘Density.diff’.

> Violin.plot(dd\_essential, main="Cell cycle normalized")

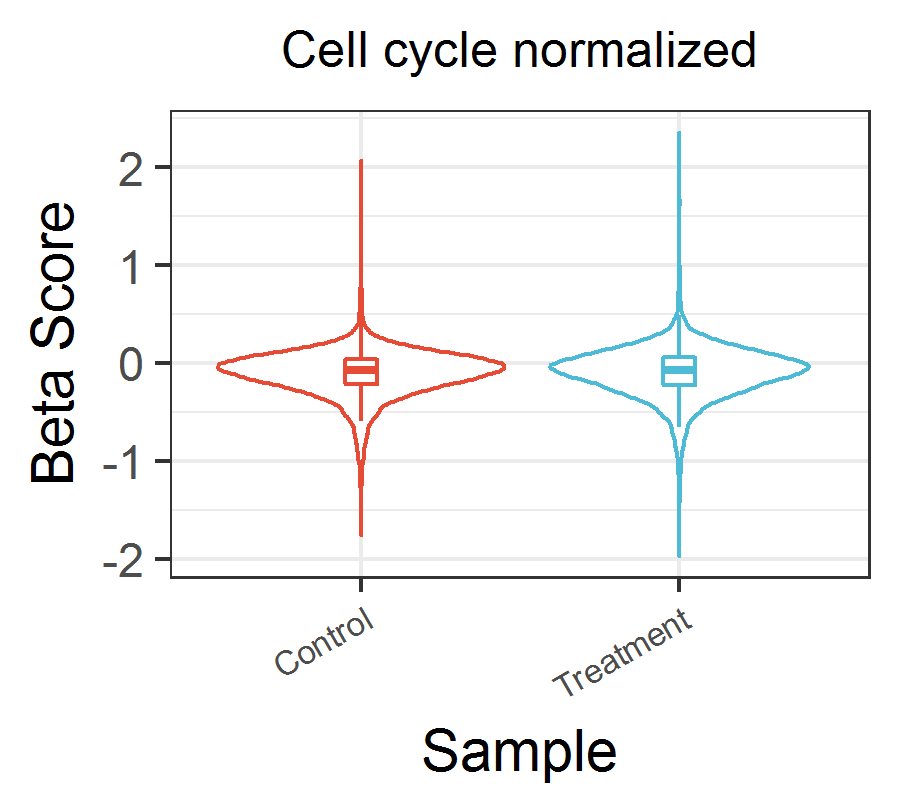


Figure 1 Violin plot of all gene beta scores

> Density.plot(dd\_essential, main="Cell cycle normalized")

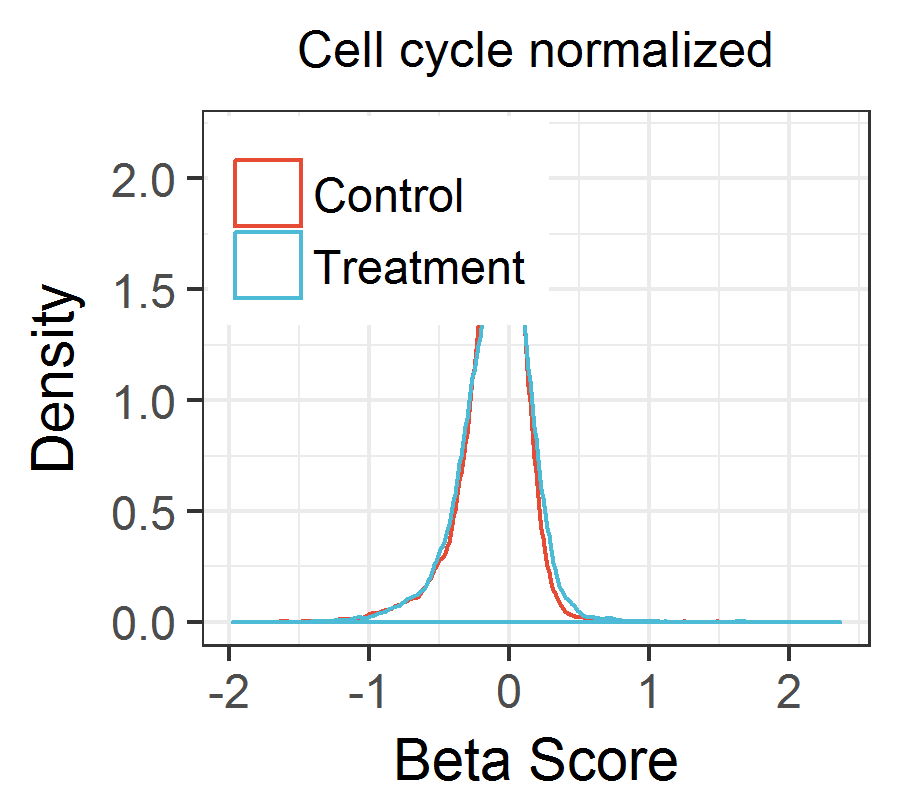


Figure 2 Density plot of all gene beta scores

> Density.diff(dd\_essential, main="Cell cycle normalized")

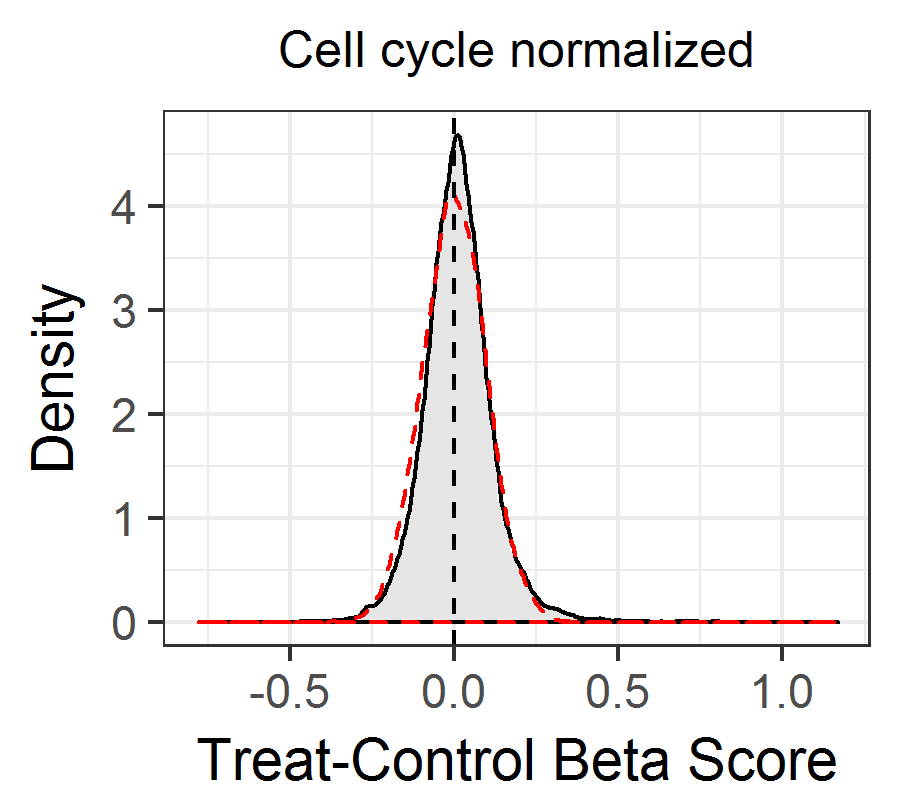


Figure 3 Density plot of Treat-Control beta scores

2.2 MAplot

MAplot was used to evaluate the data quality of gene expression profile in the past, here we use it to evaluate the data quality of normalized beta score profile.

> MA.plot(dd\_essential, cex=1, main="Cell cycle normalized")

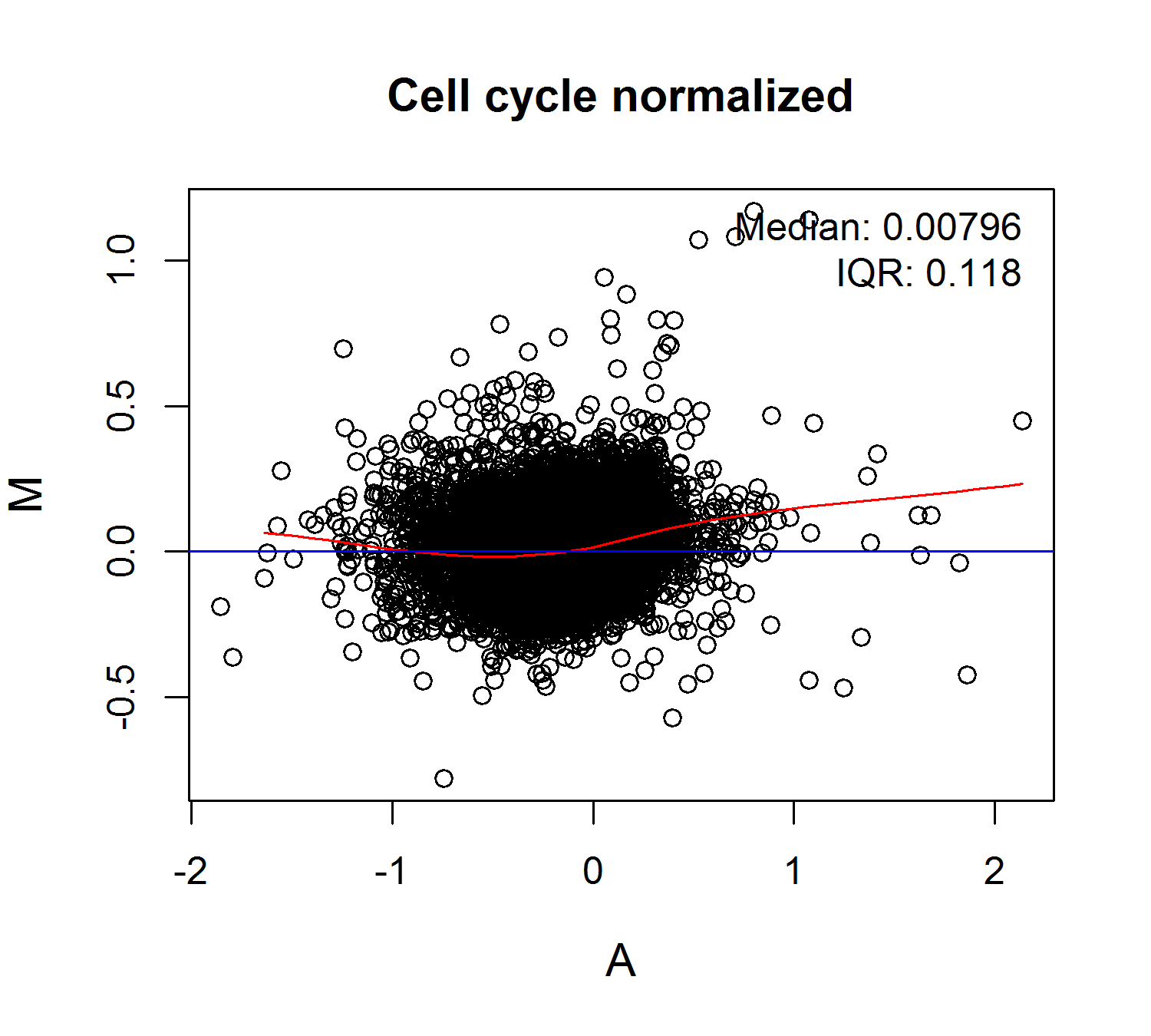


Figure 4 MAplot of beta score profile

2.3 Estimate the cell cycle time by linear fitting

After normalization, the cell cycle time in different condition should be almost consistent. Here we use linear fitting to estimate the cell cycle time in different samples with the sample ‘Control’ as x- axis control.

> ##Fitting beta score of all genes

> CellCycleFit(dd\_essential, ylab="Beta Score", main="Cell cycle normalized")

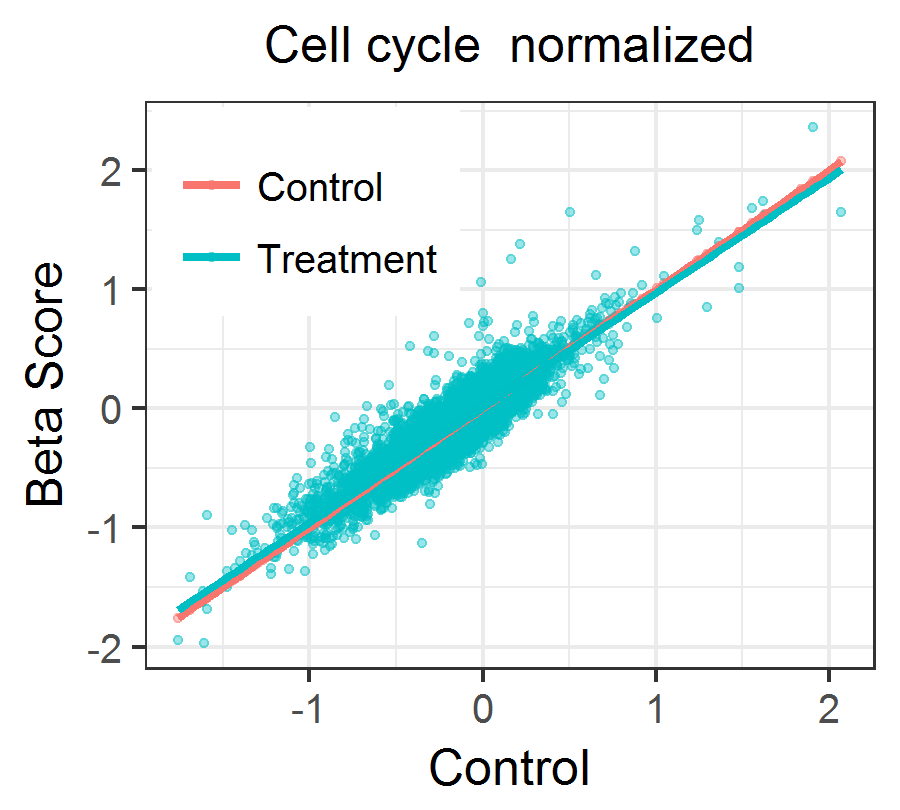


Figure 5 Linear fitting of all gene beta scores

3 Positive selection and negative selection

3.1 Group all genes into subgroups

First, use function ‘GroupAB’ to group all genes into three groups, positive selection genes (GroupA), negative selection genes (GroupB), and others. Then, visualize these three grouped genes in scatter plot using function ‘ScatterAB’.

> ddAB\_essential=GroupAB(dd\_essential)

> ScatterAB(ddAB\_essential$result, intercept=ddAB\_essential$cutoff,slope=1,main="Cell cycle normalized")

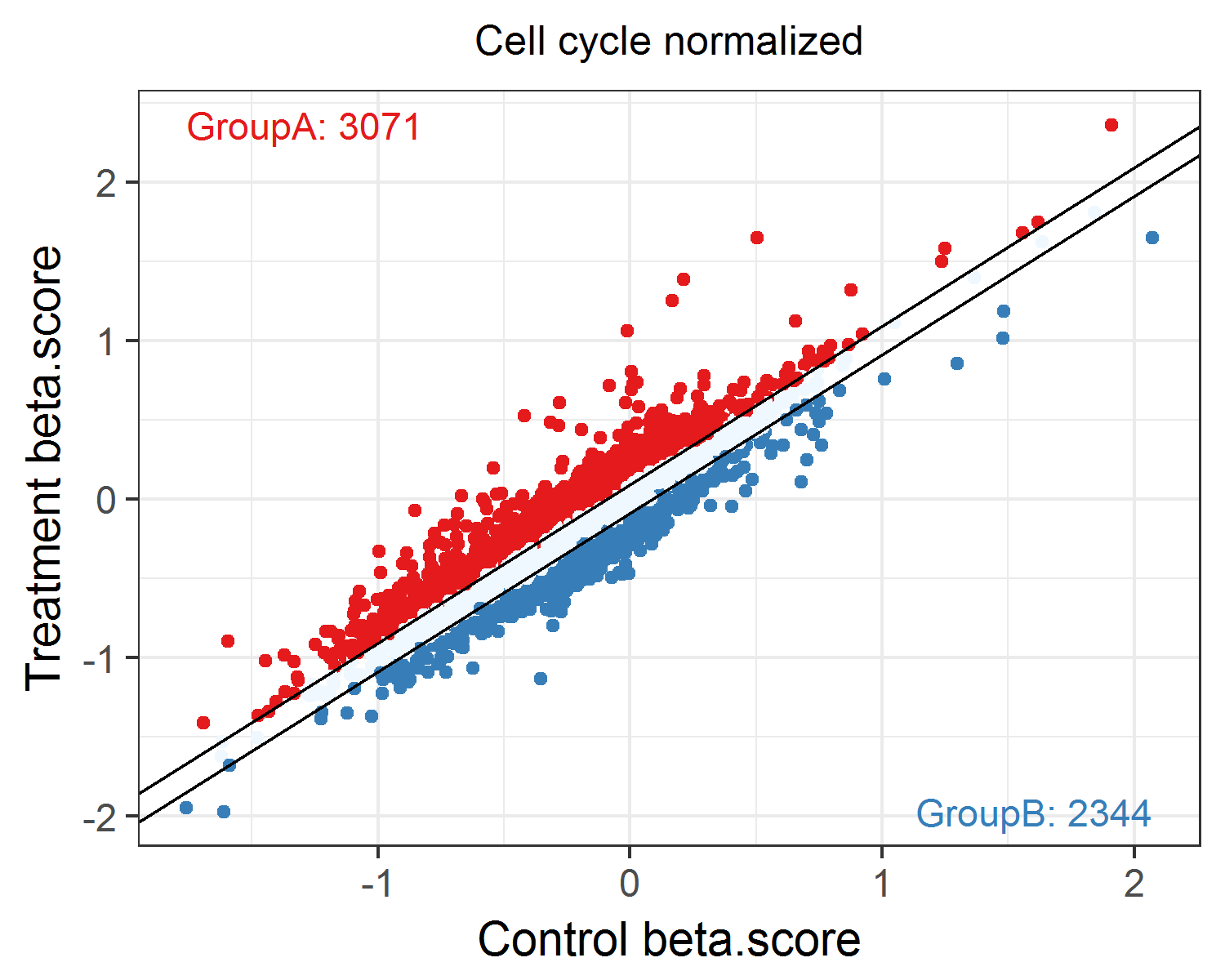


Figure 6 Scatter plot of three grouped genes

3.2 Rank the beta score differences between control and treatment samples

For groupA and GroupB genes, rank the beta score deviation between control and treatment, and find out the top and bottom genes. Use function ‘RankAB’ to mark these genes in figure.

> RankAB(ddAB\_essential$result,genelist=c(),ddAB\_essential$cutoff,main="Cell cycle normalized")

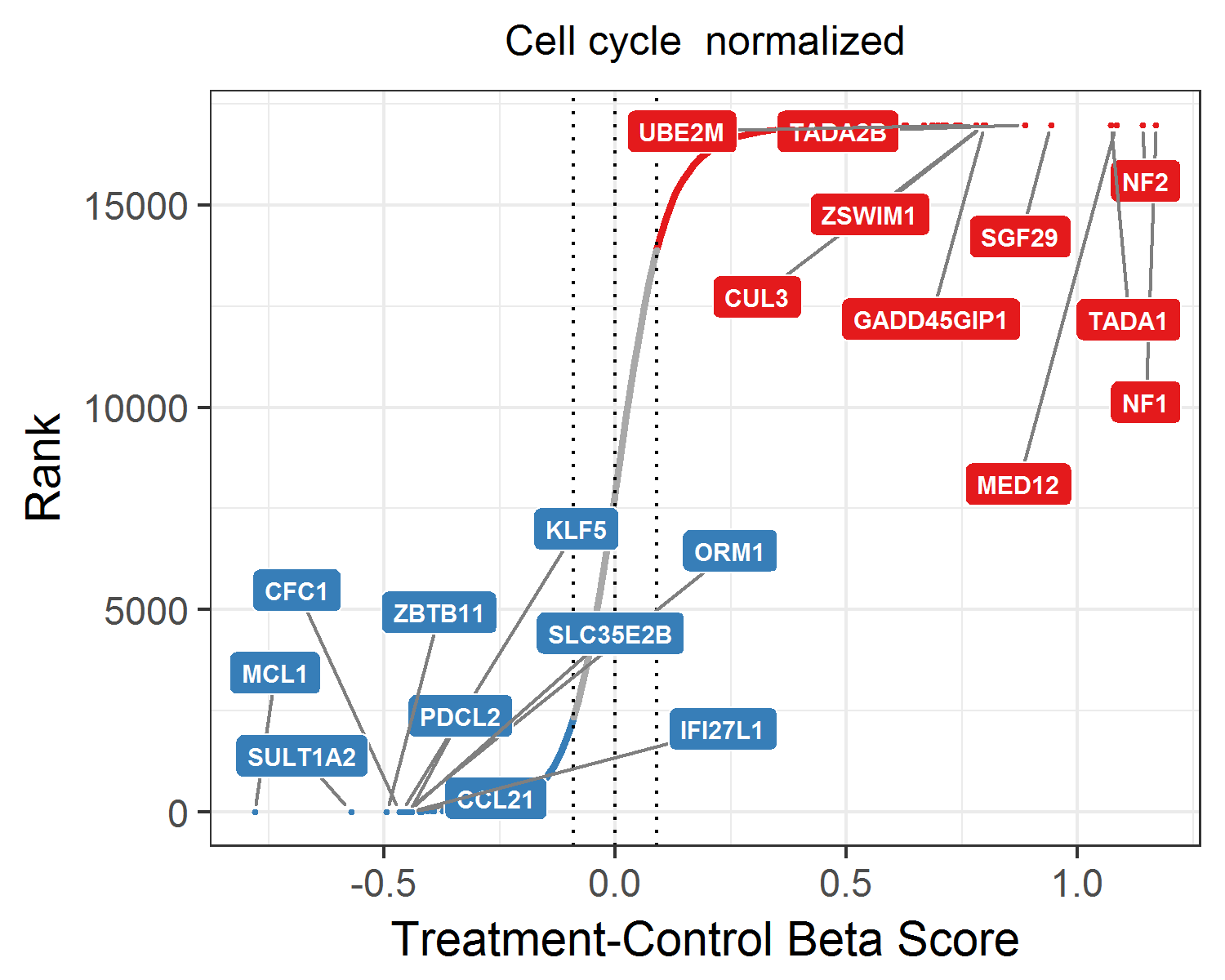


Figure 7 Rank beta score differences

3.3 Functional enrichment analysis

Gene ontology-biological process and Kyoto encyclopedia of genes and genomes (KEGG) pathway enrichment analysis are performed in this pipeline. For GroupA and GroupB genes, we use function ‘EnrichAB’ to do enrichment analysis.

> enrich\_result = EnrichAB(ddAB\_essential$result,pvalue=0.05)

> print(enrich\_result$keggA$gridPlot)

> print(enrich\_result$bpA$gridPlot)

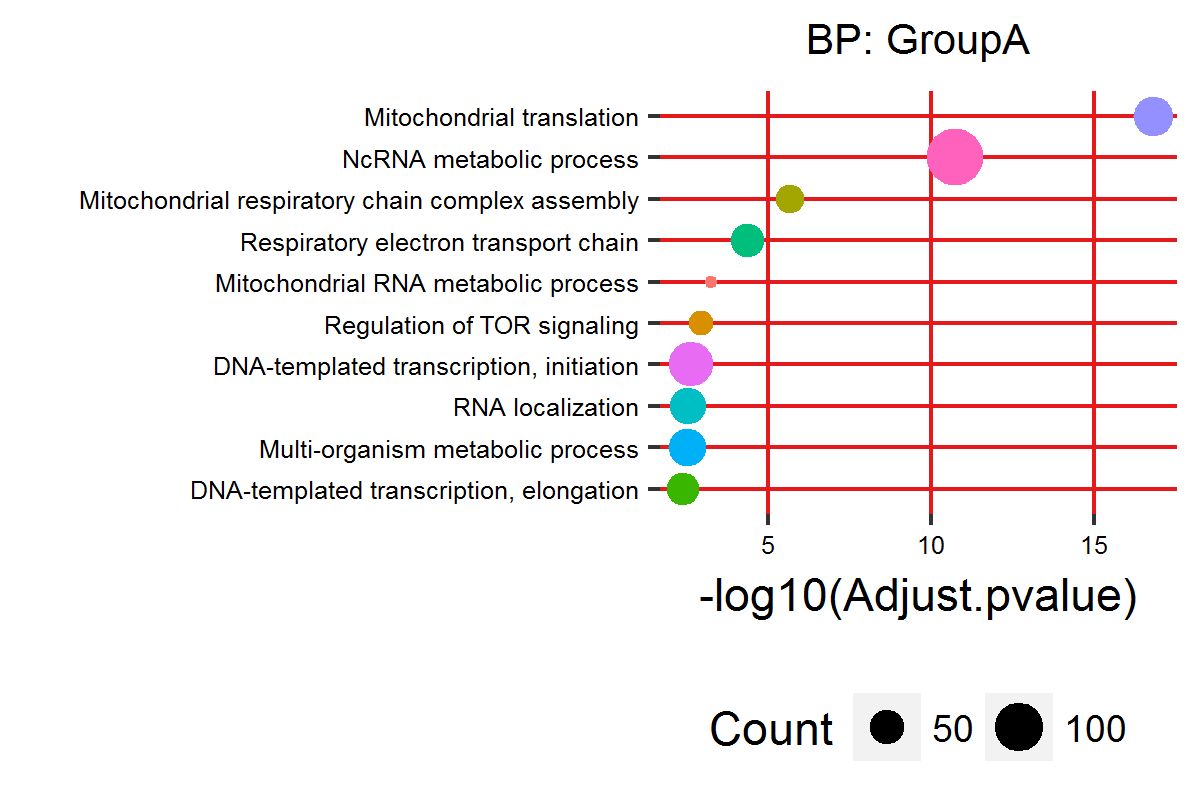
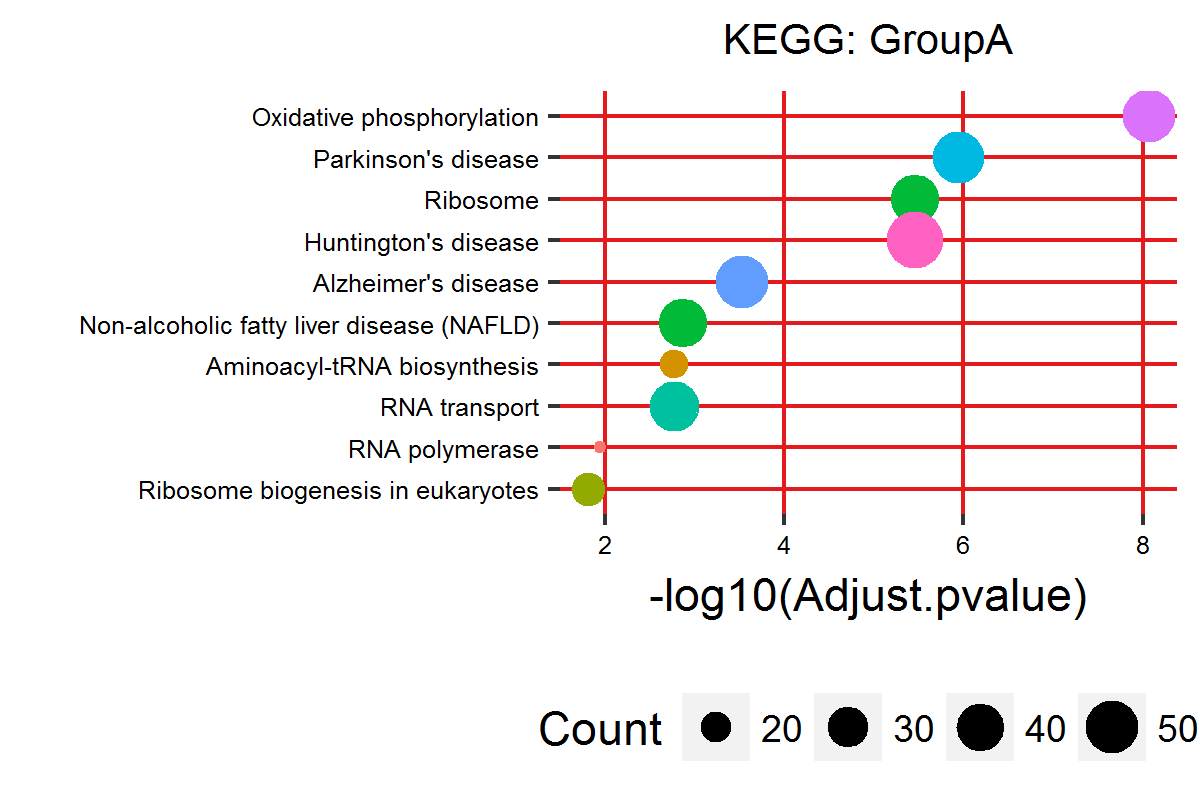


Figure 8 Grid plot of enriched terms for GroupA genes

3.4 Visualize the significantly enriched pathway map

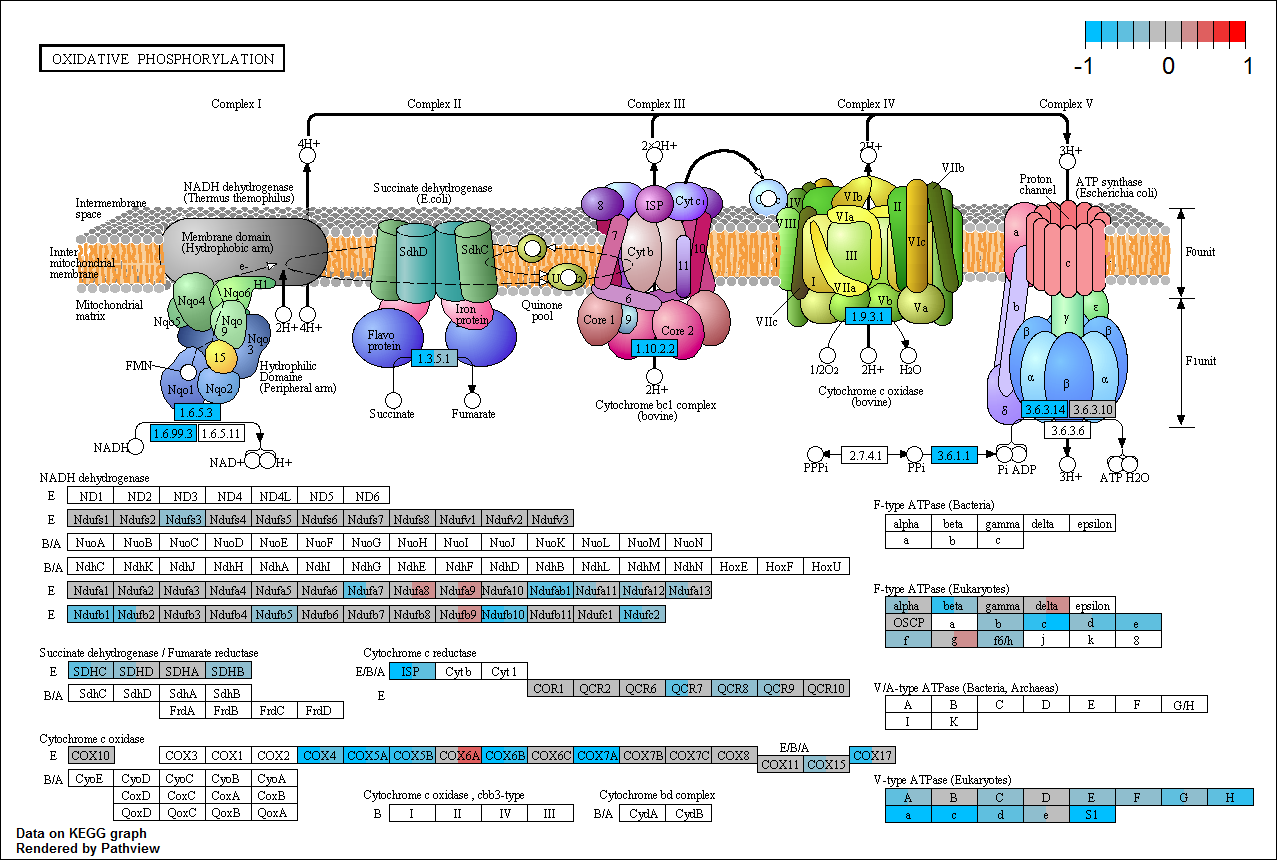
Pathway visualization was performed using pathview package, based on which we developed function ‘KeggPathwayView’ to view the beta score level in control and treatment on pathway map.

> genedata = dd\_essential

> rownames(genedata)=genedata$ENTREZID

> keggID = enrich\_result$keggA$enrichRes@result$ID[1:2]

> KeggPathwayView(gene.data = genedata[,c("Control","Treatment")], pathway.id = keggID)



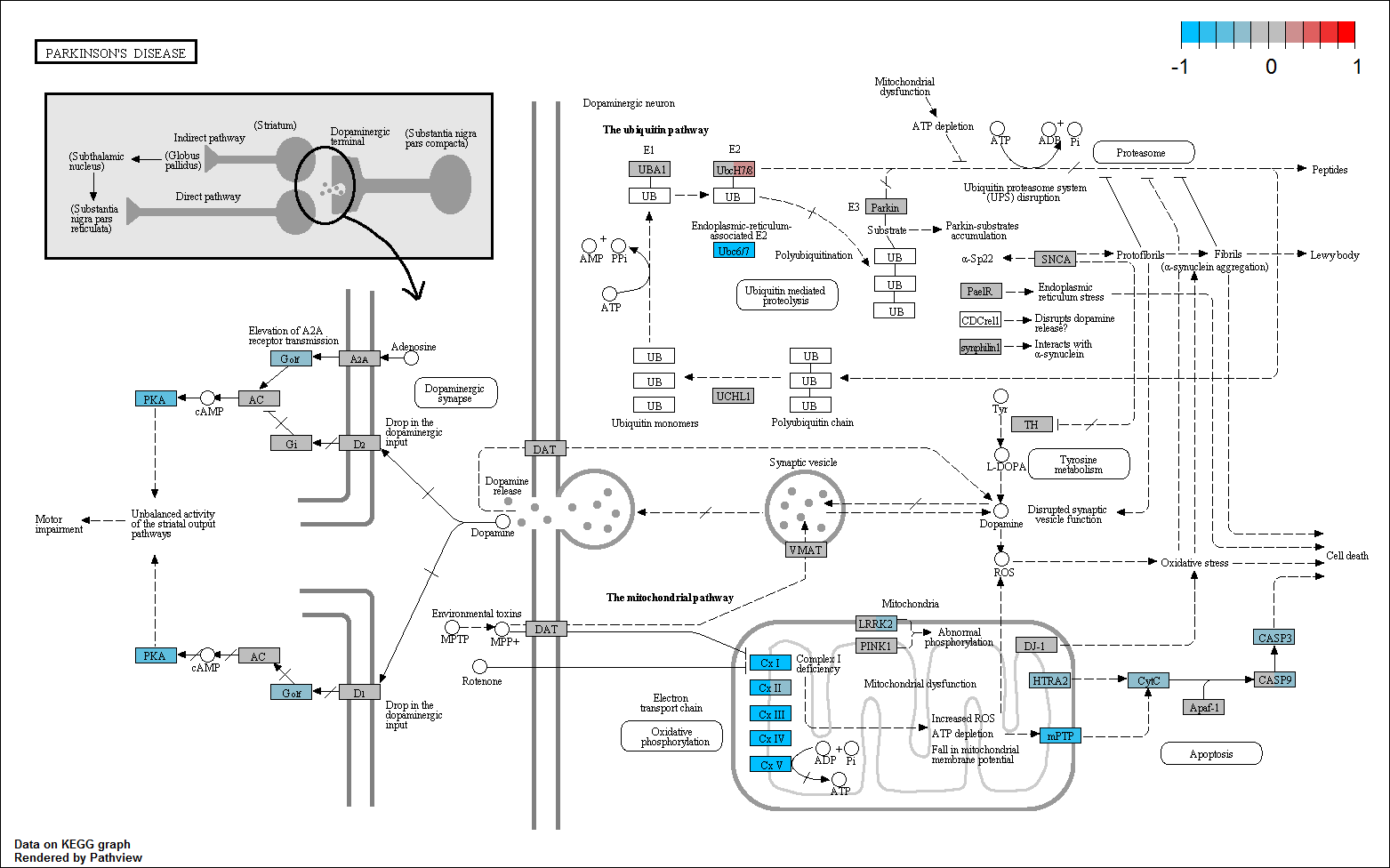


Figure 9 Visualization of top 2 enriched pathways for GroupA genes

The left color in gene block indicates beta score level in control sample, and the right indicates beta score in treatment sample.

4 Select drug related genes

4.1 Select four group meaningful genes using 9-square model

Considering beta score in control and treatment sample, we developed a 9-square model, which group all genes into several subgroups. Among these subgroups, four subgroup genes may be related to drug treatment. Group1 and Group3 genes are not selected in control sample, while they are significantly selected in treatment sample, so these genes may related to drug resistance pathways. Group2 and Group4 genes are selected in control, but they are not selected in treatment, so maybe these genes are located on drug target pathways.

In this package, use function ‘Square.plot’ can select these four meaningful subgroup genes and get the 9-Square scatter plot object.

> dd2=dd\_essential[,c("Gene","Treatment","Control","ENTREZID")]

> P2=Square.plot(dd2,main="Cell cycle normalized")

> print(P2$p)

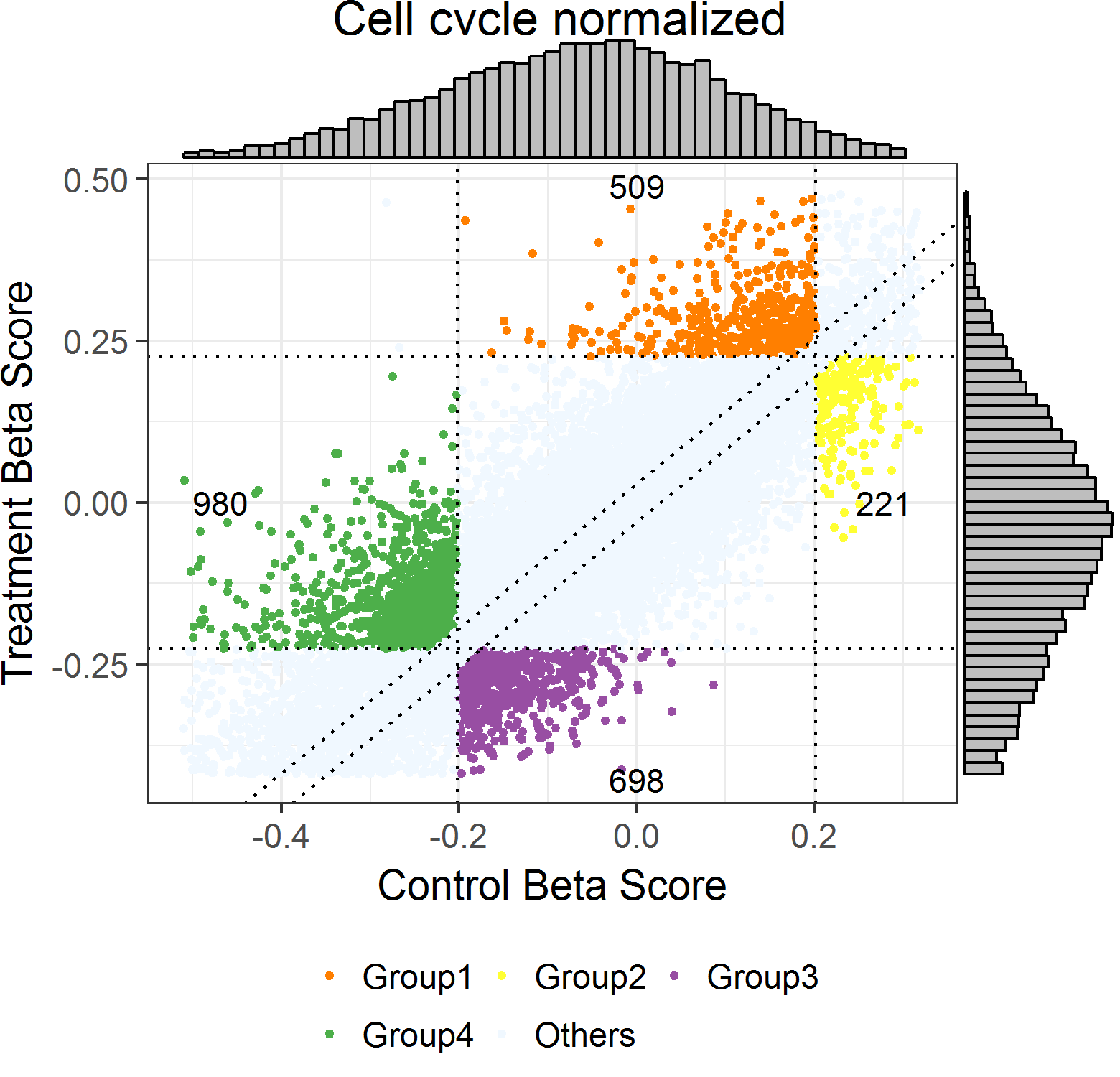


Figure 10 Nine-Square model

4.2 BP and KEGG enrichment analysis

For all the four subgroup genes and random combination of them, BP and KEGG enrichment analysis can be performed easily using ‘EnrichSquare’ function.

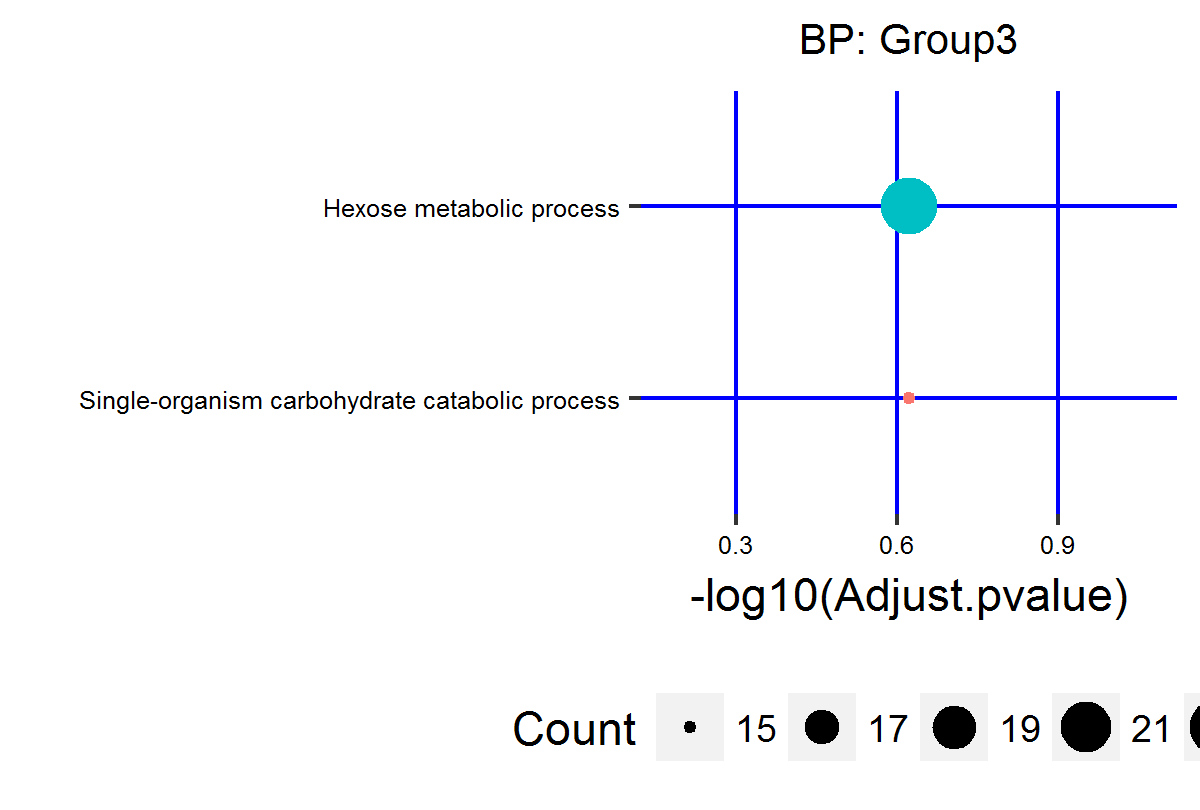
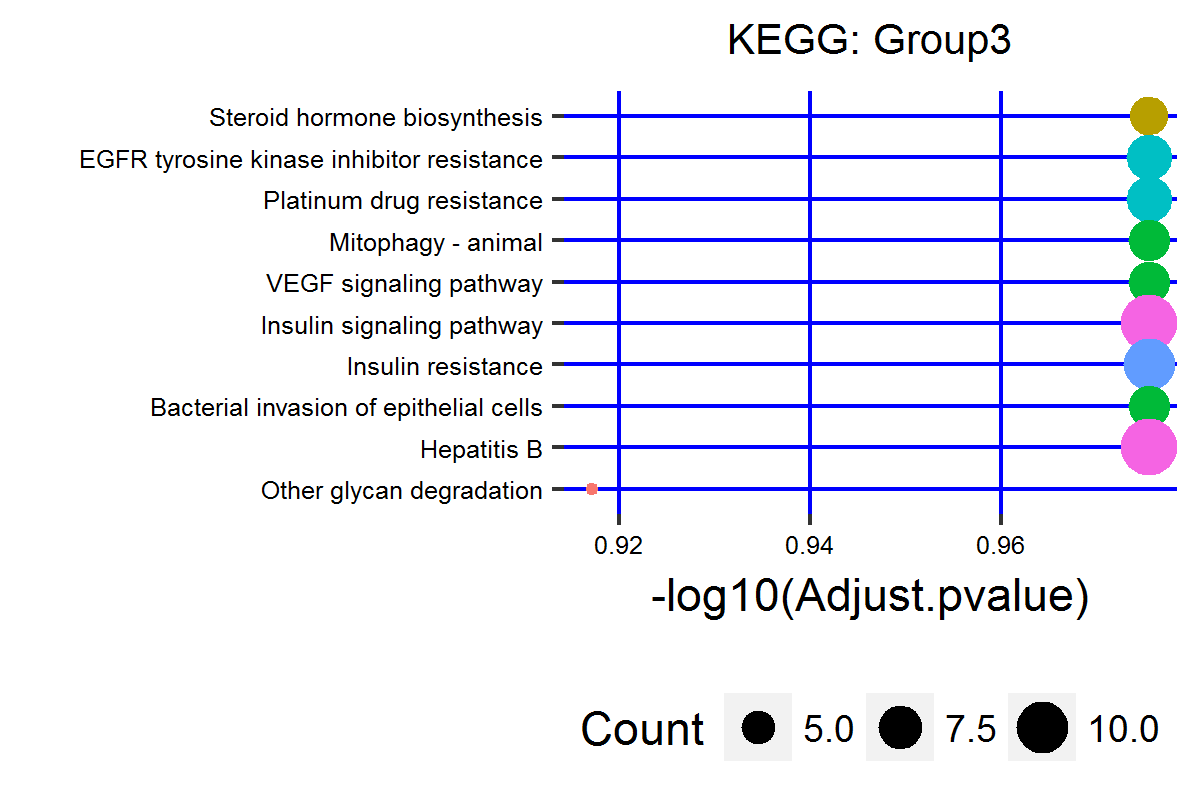
> enrich\_result2 = EnrichSquare(P2$dd1,pvalue=0.05)

> print(enrich\_result2$kegg3$gridPlot)

> print(enrich\_result2$bp3$gridPlot)

> print(enrich\_result2$kegg4$gridPlot)

> print(enrich\_result2$bp4$gridPlot)



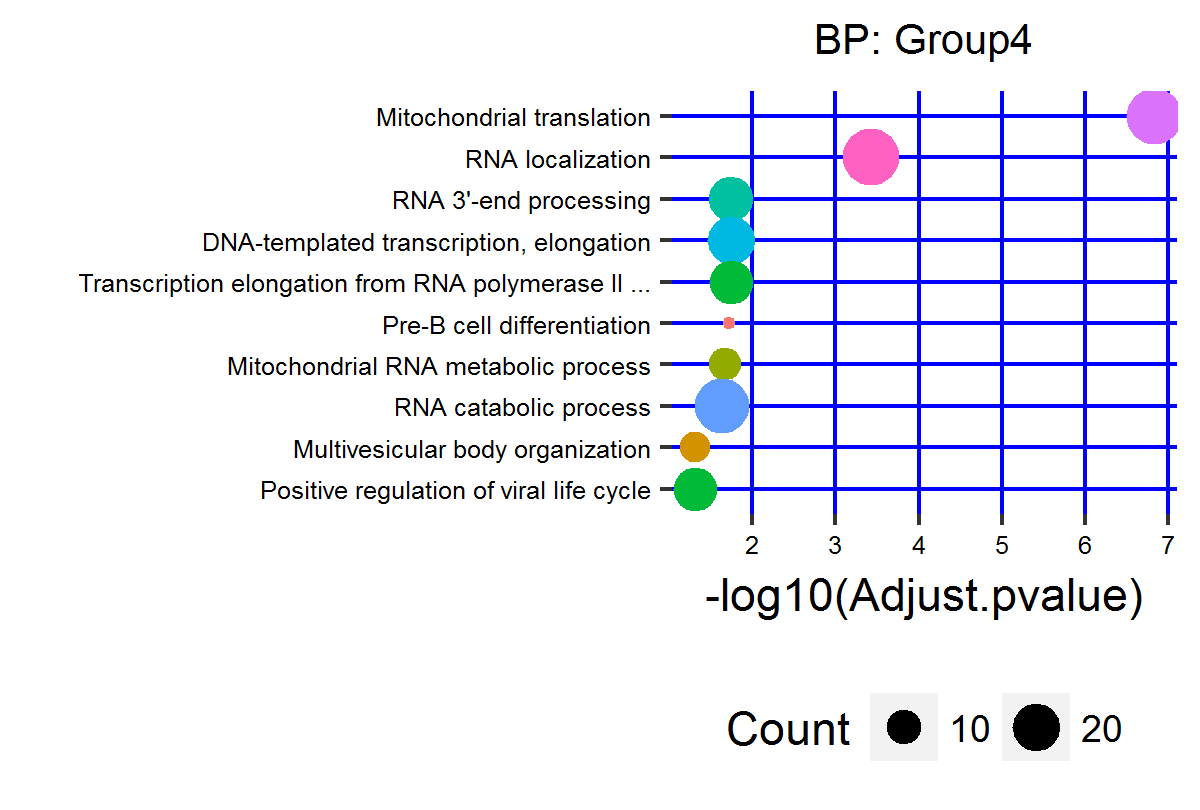
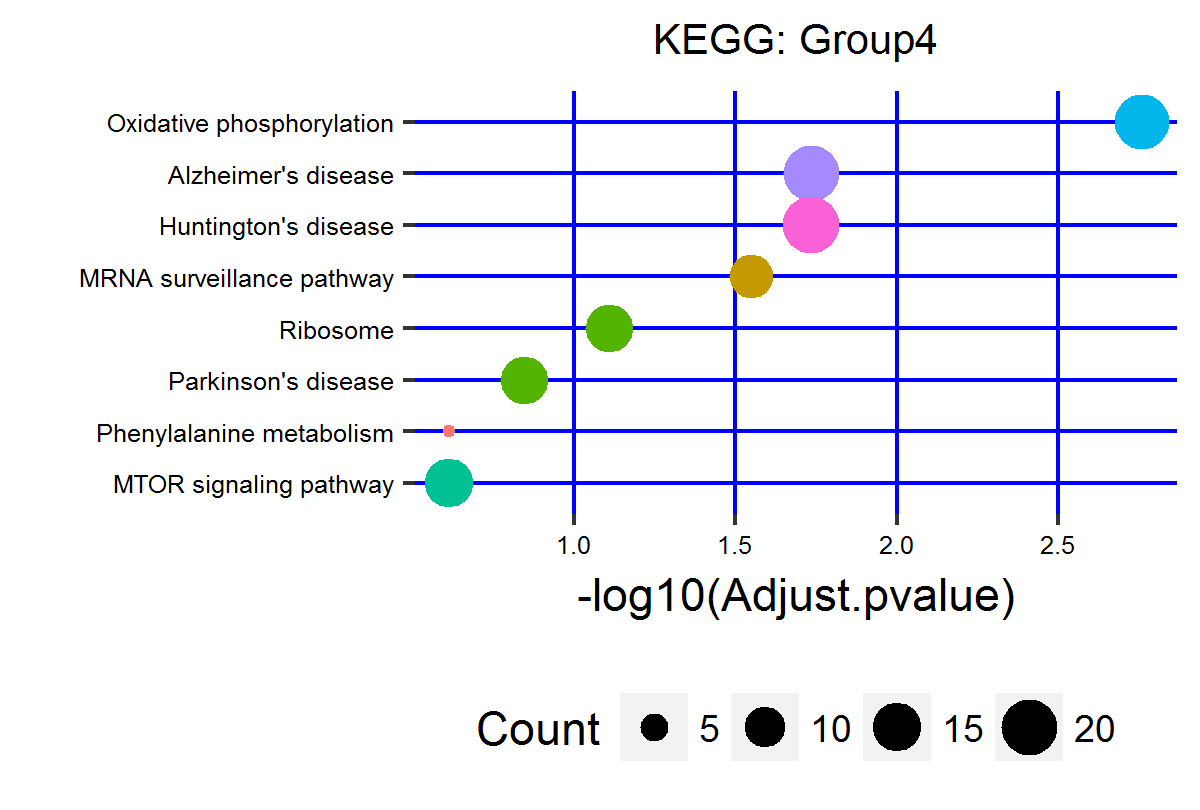


Figure 11 Grid plot of enriched terms for Group3 and Group4 genes

4.3 Pathway visualization

Pathway visualization can be done using function ‘KeggPathwayView’, the same as section 3.4. Below we visualize the top one significantly enriched pathway in four subgroups separately.

> genedata = dd\_essential

> rownames(genedata)=genedata$ENTREZID

> keggID = enrich\_result2$kegg1$enrichRes@result$ID[1]

> KeggPathwayView(gene.data = genedata[,c("Control","Treatment")], pathway.id = keggID)

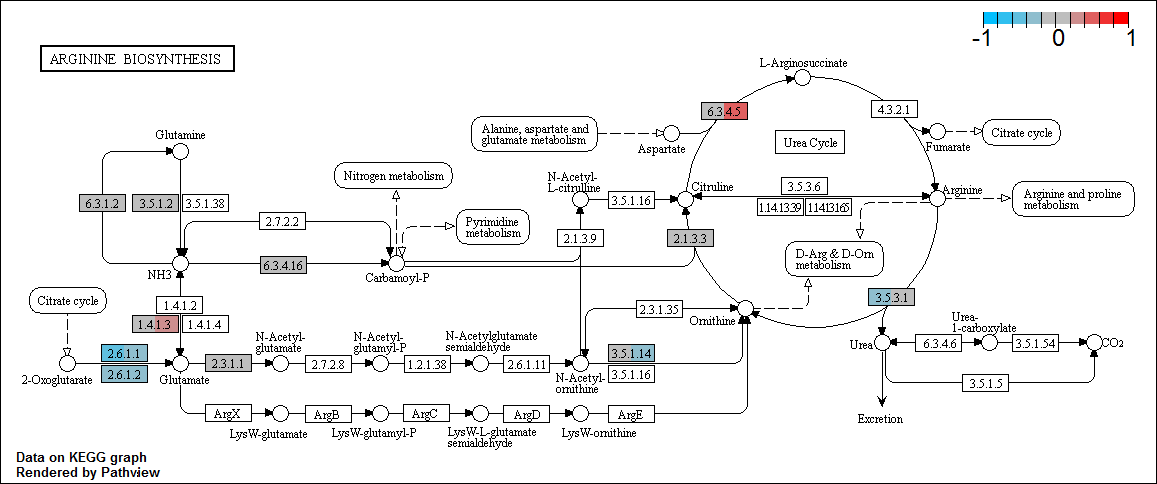


Figure 12 Visualization of Arginine biosynthesis pathway

> keggID = enrich\_result2$kegg2$enrichRes@result$ID[1]

> KeggPathwayView(gene.data = genedata[,c("Control","Treatment")], pathway.id = keggID)

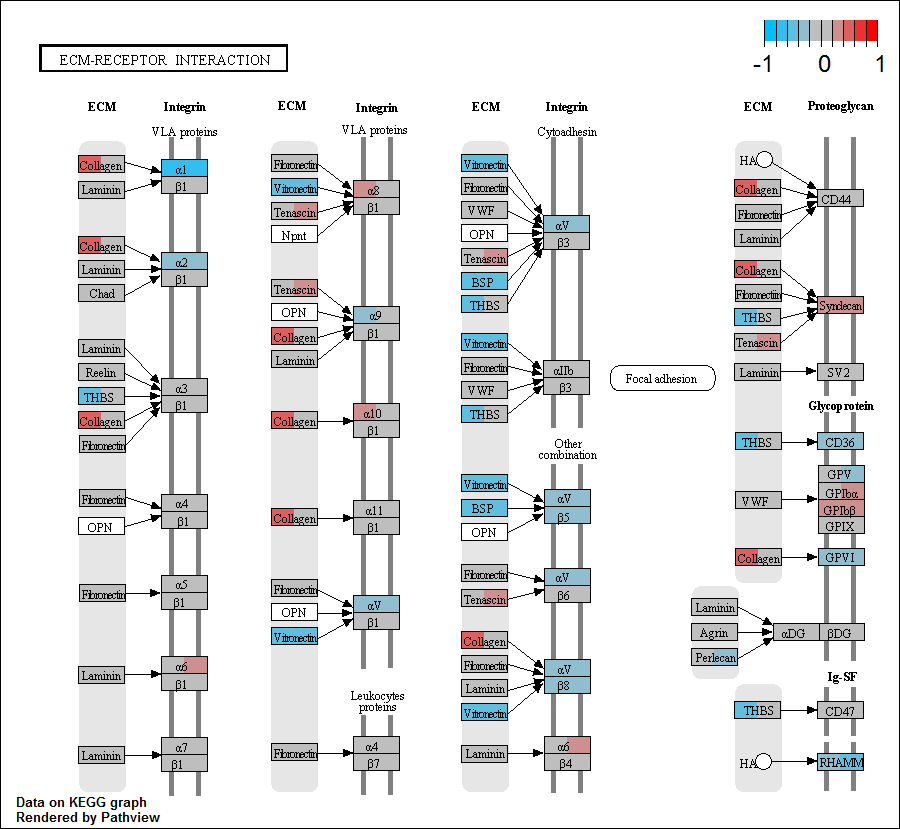


Figure 13 Visualization of ECM-receptor interaction pathway

> keggID = enrich\_result2$kegg3$enrichRes@result$ID[1]

> KeggPathwayView(gene.data = genedata[,c("Control","Treatment")], pathway.id = keggID)

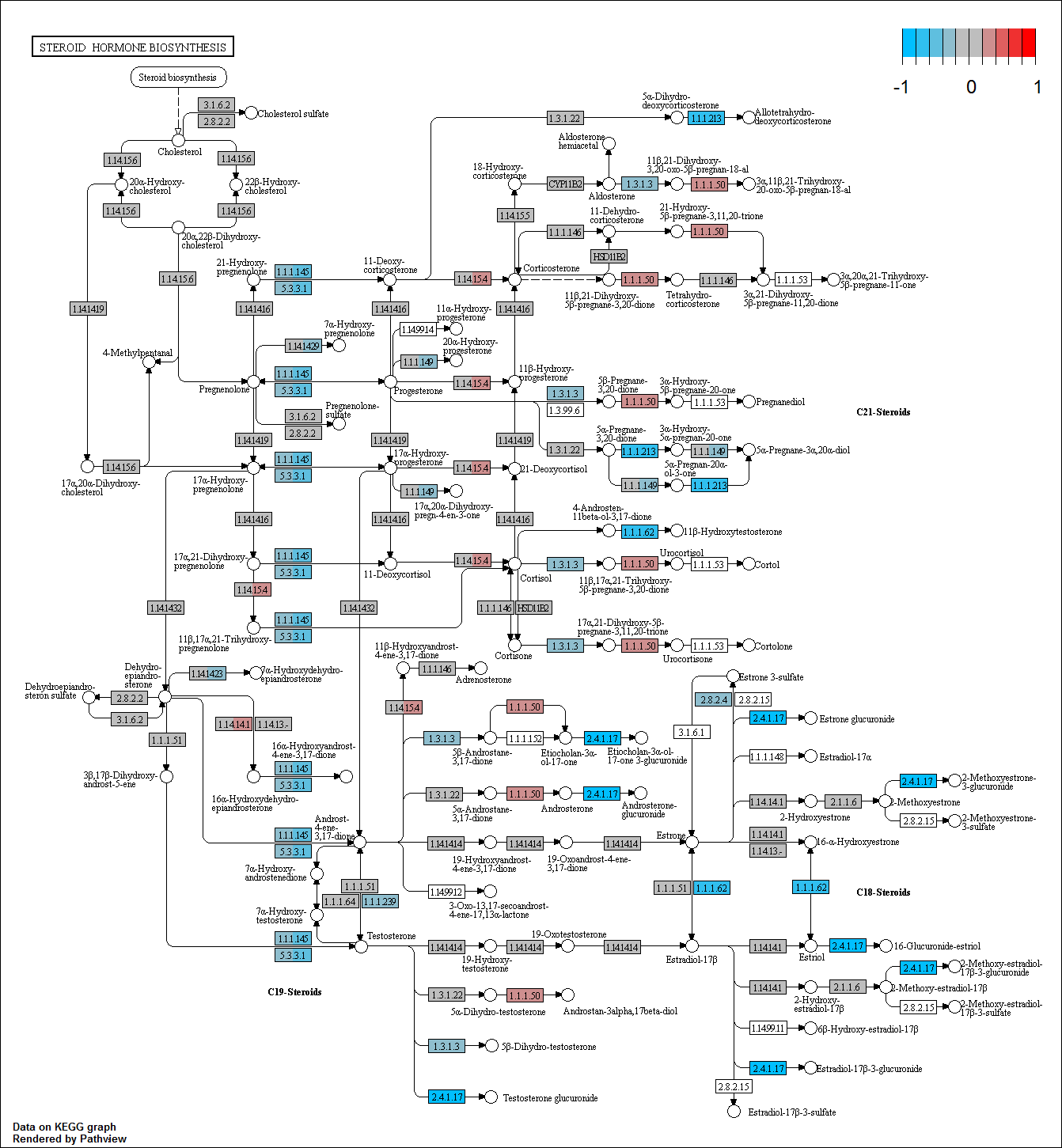


Figure 14 Visualization of Steroid hormone biosynthesis pathway

> keggID = enrich\_result2$kegg4$enrichRes@result$ID[1]

> KeggPathwayView(gene.data = genedata[,c("Control","Treatment")], pathway.id = keggID)

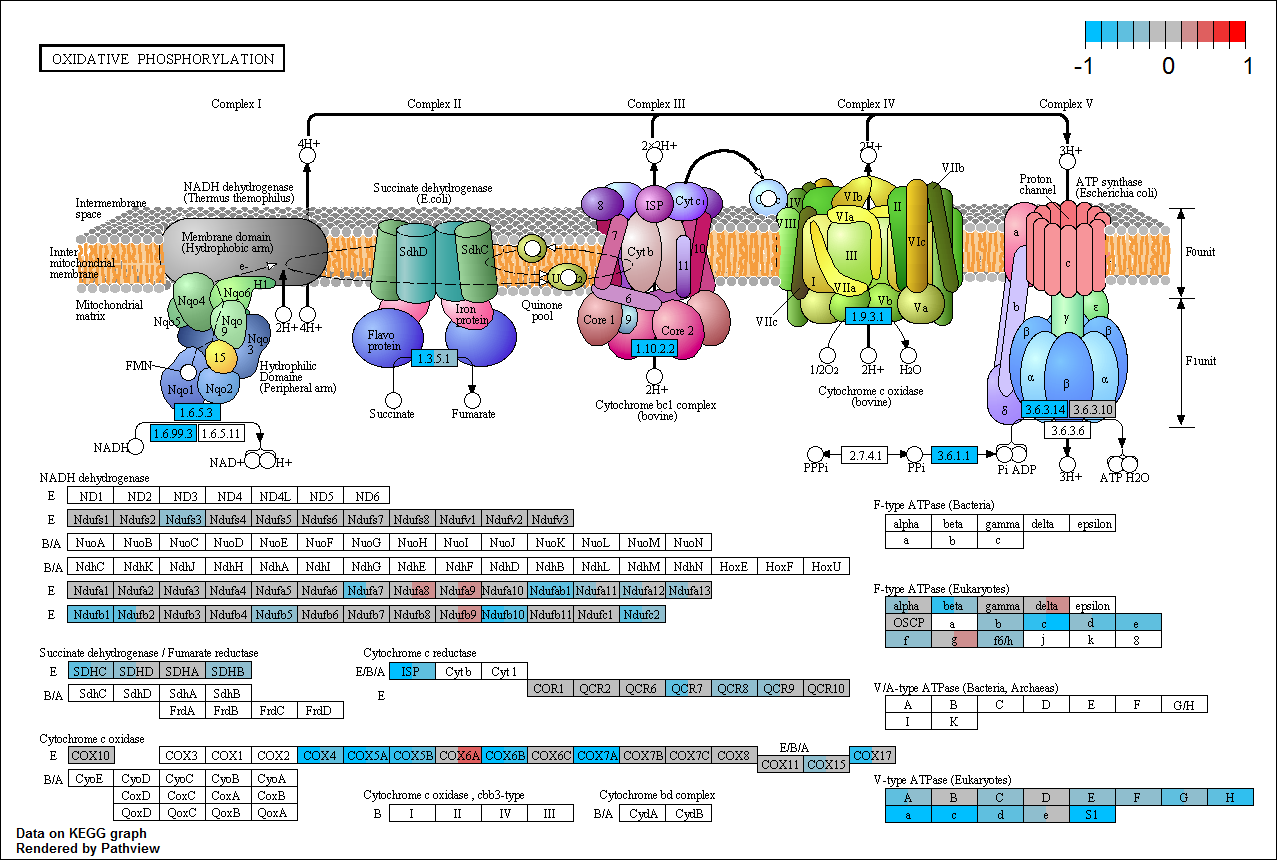


Figure 15 Visualization of Oxidative phosphorylation pathway

5 Run pipeline with MAGeCK-RRA results

5.1 Gene summary file in MAGeCK-RRA results

For experiments which only contain two conditions, we recommend to use MAGeCK-RRA to identify essential genes from CRISPR/Cas9 knockout screens and tests the statistical significance of each observed change between two conditions. We use the term 'essential' to refer to positively or negatively selected genes. Gene summary file in MAGeCK-RRA results summarize the statistical significance of positive selection and negative selection.

> data("BRAF\_rra.gene\_summary")

> head(BRAF\_rra.gene\_summary)

id num neg.score neg.p.value neg.fdr neg.rank neg.goodsgrna neg.lfc pos.score pos.p.value

1 CMIP 6 4.8104e-10 2.8420e-07 0.001650 1 5 -0.66277 0.76014 0.73823

2 MCL1 3 1.9829e-09 2.8420e-07 0.001650 2 3 -1.18000 1.00000 1.00000

3 ITGB2 4 2.9590e-08 2.8420e-07 0.001650 3 4 -1.20780 0.99939 0.99908

4 SCN2A 4 1.9322e-07 8.5260e-07 0.003713 4 3 -0.61912 0.19756 0.27533

5 GRB2 3 2.6728e-06 1.5631e-05 0.048680 5 2 -0.72466 0.82825 0.80865

6 EGFR 4 3.0225e-06 1.6768e-05 0.048680 6 3 -0.45355 0.95912 0.94822

pos.fdr pos.rank pos.goodsgrna pos.lfc

1 0.977480 12939 1 -0.66277

2 1.000000 17140 0 -1.18000

3 0.999967 17115 0 -1.20780

4 0.886655 5329 1 -0.61912

5 0.984303 14036 0 -0.72466

6 0.993803 16331 0 -0.45355

> dd.rra = ReadRRA(BRAF\_rra.gene\_summary)

> head(dd.rra)

Official neg.fdr pos.fdr ENTREZID

1 A1BG 1 0.931838 1

2 A1CF 1 0.989673 29974

3 A2M 1 0.932524 2

4 A2ML1 1 0.919760 144568

5 A4GALT 1 0.741204 53947

6 A4GNT 1 0.977480 51146

5.2 Negative selection and positive selection

Take 0.05 as cutoff, get negative selection and positive selection genes and do enrichment analysis on KEGG pathway and GO BP terms.

> geneList = dd.rra$neg.fdr

> names(geneList)=dd.rra$ENTREZID

> genes = dd.rra[dd.rra$neg.fdr<0.05, "ENTREZID"]

> kegg.neg=enrichment\_analysis(geneList = geneList, genes=genes, type = "KEGG", pvalueCutoff = 0.05, plotTitle="KEGG: neg")

> bp.neg=enrichment\_analysis(geneList = geneList, genes=genes, type = "BP", pvalueCutoff = 0.05, plotTitle="BP: neg")

> print(kegg.neg$gridPlot)

> print(bp.neg$gridPlot)

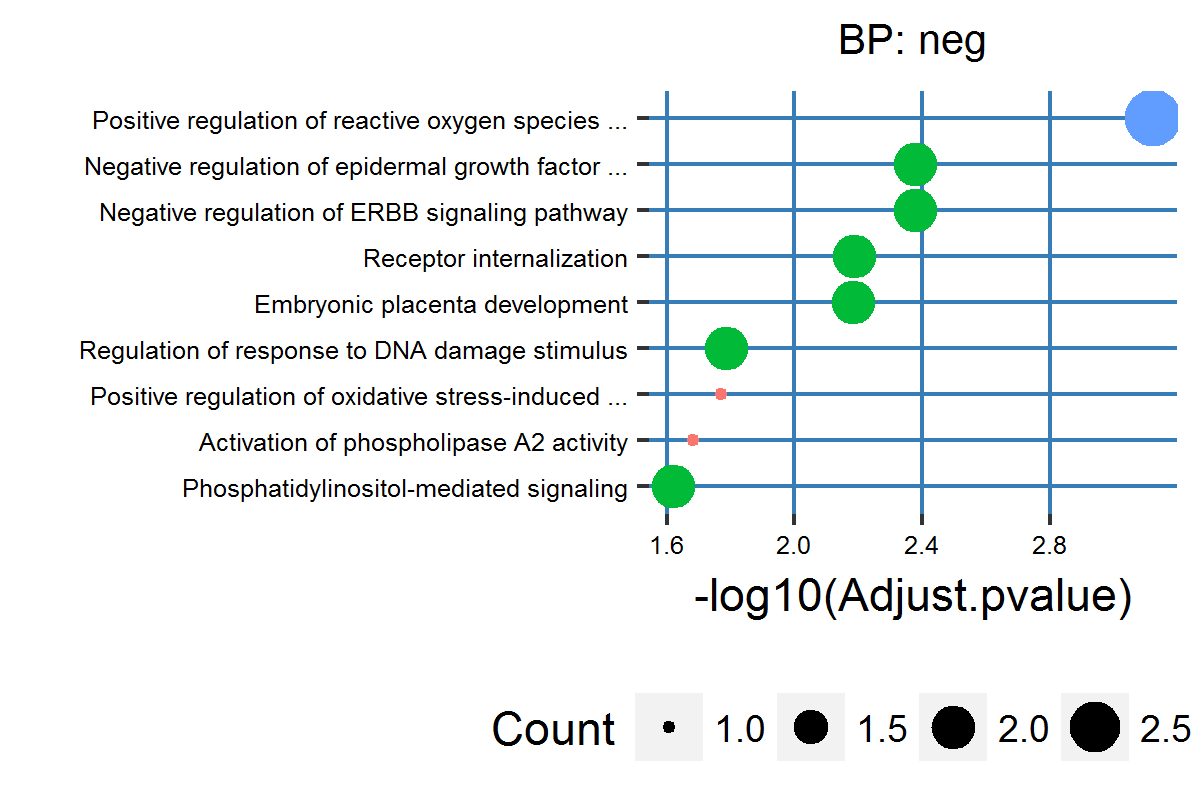
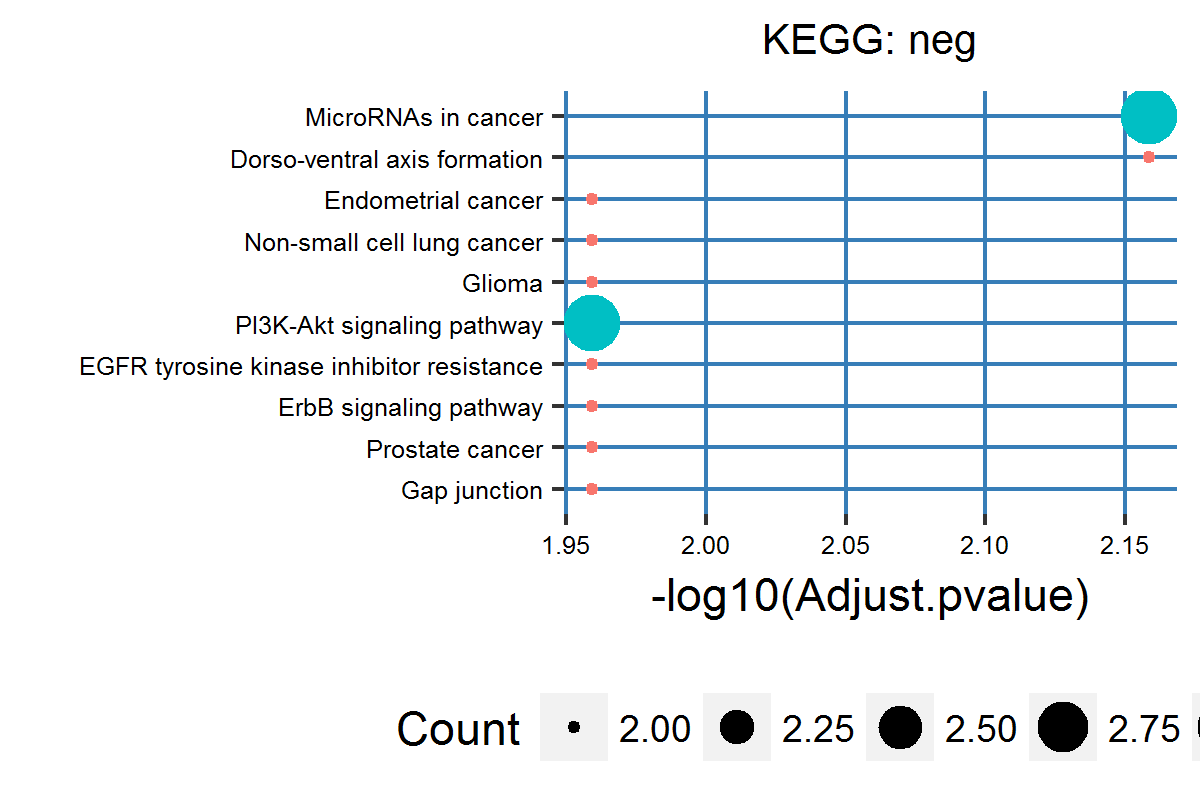


Figure 16 Grid plot of enriched terms for negative selection genes

> geneList = dd.rra$pos.fdr

> names(geneList)=dd.rra$ENTREZID

> genes = dd.rra[dd.rra$pos.fdr<0.05, "ENTREZID"]

> kegg.pos=enrichment\_analysis(geneList = geneList, genes=genes, type = "KEGG", pvalueCutoff = 0.05, plotTitle="KEGG: pos", gridColour="#e41a1c" )

> bp.pos=enrichment\_analysis(geneList = geneList, genes=genes, method = "ORT", type = "BP", pvalueCutoff = 0.05, plotTitle="BP: pos", gridColour="#e41a1c")

> print(kegg.pos$gridPlot)

> print(bp.pos$gridPlot)

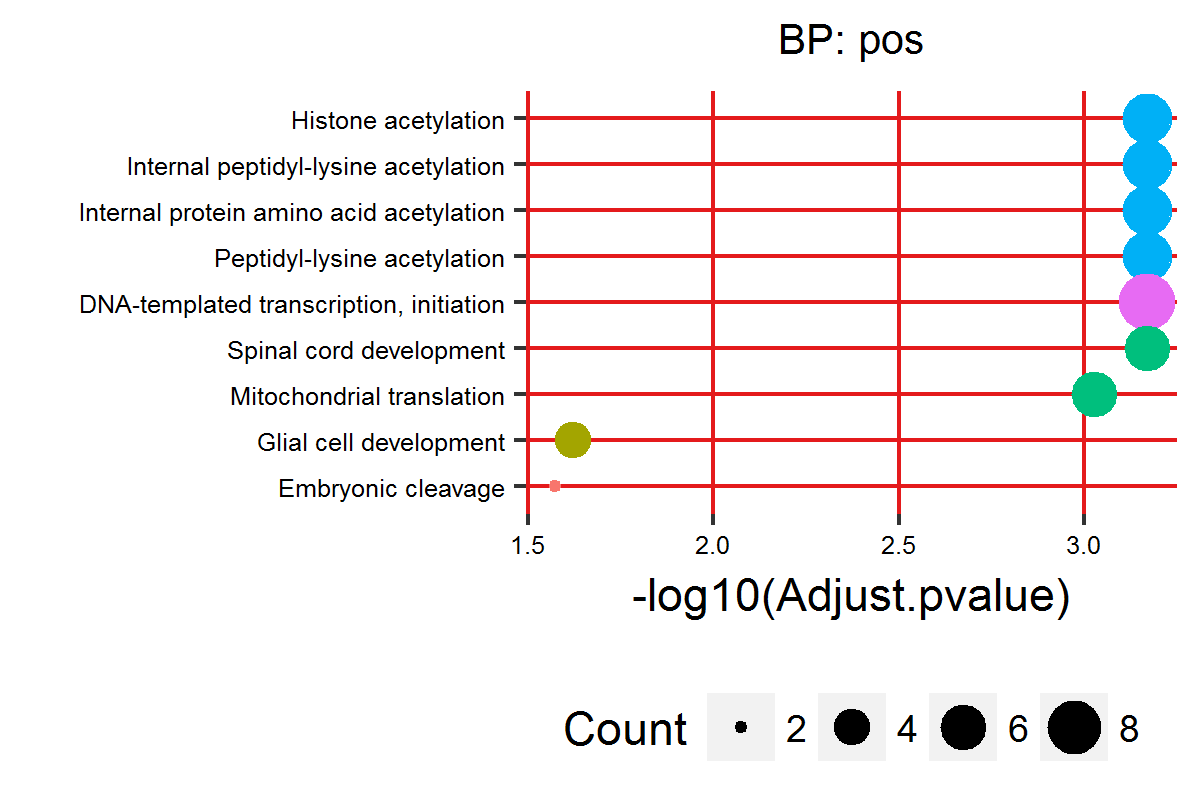
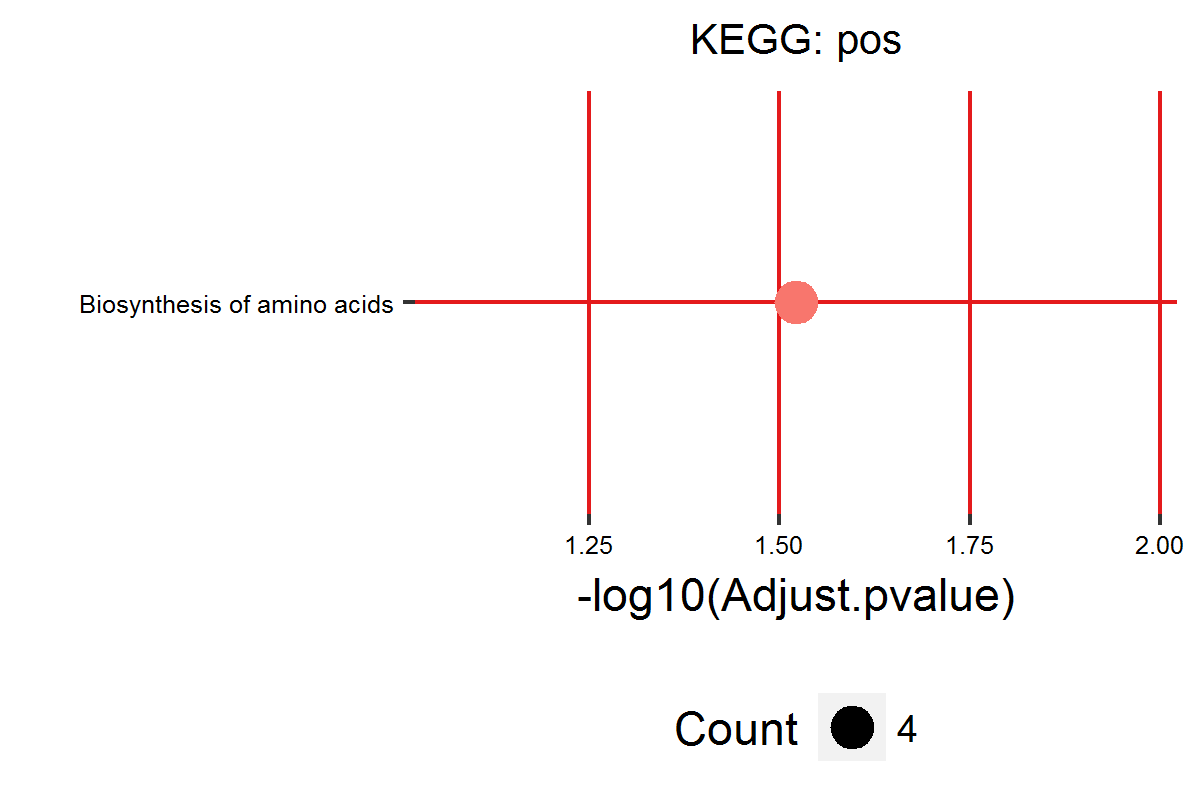


Figure 17 Grid plot of enriched terms for positive selection genes

6 Session Info

> sessionInfo("MAGeCKFlute")

R version 3.3.1 (2016-06-21)

Platform: x86\_64-w64-mingw32/x64 (64-bit)

Running under: Windows 7 x64 (build 7601) Service Pack 1

locale:

[1] LC\_COLLATE=Chinese (Simplified)\_People's Republic of China.936

[2] LC\_CTYPE=Chinese (Simplified)\_People's Republic of China.936

[3] LC\_MONETARY=Chinese (Simplified)\_People's Republic of China.936

[4] LC\_NUMERIC=C

[5] LC\_TIME=Chinese (Simplified)\_People's Republic of China.936

attached base packages:

character(0)

other attached packages:

[1] MAGeCKFlute\_0.1.0

loaded via a namespace (and not attached):

[1] KEGGgraph\_1.30.0 ggrepel\_0.6.5 Rcpp\_0.12.8 lattice\_0.20-33

[5] GO.db\_3.3.0 tidyr\_0.6.0 png\_0.1-7 Biostrings\_2.40.2

[9] assertthat\_0.1 digest\_0.6.10 R6\_2.1.3 plyr\_1.8.4

[13] stats4\_3.3.1 RSQLite\_1.1-1 httr\_1.2.1 ggplot2\_2.2.0

[17] BiocInstaller\_1.22.3 utils\_3.3.1 zlibbioc\_1.18.0 lazyeval\_0.2.0

[21] SparseM\_1.74 data.table\_1.10.4 annotate\_1.50.1 Rgraphviz\_2.16.0

[25] S4Vectors\_0.10.3 Matrix\_1.2-7.1 preprocessCore\_1.34.0 qvalue\_2.4.2

[29] GOstats\_2.38.1 splines\_3.3.1 stringr\_1.1.0 topGO\_2.24.0

[33] pathview\_1.12.0 igraph\_1.0.1 RCurl\_1.95-4.8 munsell\_0.4.3

[37] stats\_3.3.1 BiocGenerics\_0.18.0 KEGGREST\_1.12.3 tibble\_1.2

[41] gridExtra\_2.2.1 matrixStats\_0.51.0 IRanges\_2.6.1 fitdistrplus\_1.0-8

[45] grDevices\_3.3.1 XML\_3.98-1.5 AnnotationForge\_1.14.2 reshape\_0.8.6

[49] MASS\_7.3-45 bitops\_1.0-6 grid\_3.3.1 RBGL\_1.48.1

[53] xtable\_1.8-2 GSEABase\_1.34.1 gtable\_0.2.0 affy\_1.50.0

[57] DBI\_0.5-1 magrittr\_1.5 datasets\_3.3.1 scales\_0.4.1

[61] graph\_1.50.0 stringi\_1.1.1 GOSemSim\_1.30.3 XVector\_0.12.1

[65] reshape2\_1.4.2 genefilter\_1.54.2 affyio\_1.42.0 DO.db\_2.9

[69] clusterProfiler\_3.0.5 graphics\_3.3.1 base\_3.3.1 ggsci\_2.4

[73] org.Hs.eg.db\_3.3.0 tools\_3.3.1 Biobase\_2.32.0 Category\_2.38.0

[77] parallel\_3.3.1 survival\_2.39-5 AnnotationDbi\_1.34.4 colorspace\_1.2-6

[81] DOSE\_2.10.7 memoise\_1.0.0 methods\_3.3.1