Statistical analysis and visualization of functional profiles for genes and gene clusters

Guangchuang Yu

School of Public Health

The University of Hong Kong

guangchuangyu@gmail.com

September 23, 2015

Abstract

clusterProfiler supports enrichment analysis of Gene Ontology (GO) and Kyoto Encyclopedia of genes and Genomes (KEGG) with either hypergeometric test or Gene Set Enrichment Analysis (GSEA). *clusterProfiler* adjust the estimated significance level to account for multiple hypothesis testing and also *q-values* were calculated for FDR control. It supports several visualization methods, including `barplot`, `cnetplot`, `enrichMap` and `gseaplot`. *clusterProfiler* also supports comparing functional profiles among gene clusters. It supports comparing biological themes of GO, KEGG, Disease Ontology (via *DOSE*) and Reactome pathways (via *ReactomePA*).

clusterProfiler version: 2.2.7

If you use *clusterProfiler* in published research, please cite:

G Yu, LG Wang, Y Han, QY He. **clusterProfiler: an R package for comparing biological themes among gene clusters**.
Journal of Integrative Biology 2012, 16(5):284-287.
http://dx.doi.org/10.1089/omi.2011.0118

Contents

1 Introduction

In recently years, high-throughput experimental techniques such as microarray, RNA-Seq and mass spectrometry can detect cellular molecules at systems-level. These kinds of analyses generate huge quantitaties of data, which need to be given a biological interpretation. A commonly used approach is via clustering in the gene dimension for grouping different genes based on their similarities [1].

To search for shared functions among genes, a common way is to incorporate the biological knowledge, such as Gene Ontology (GO) and Kyoto Encyclopedia of Genes and Genomes (KEGG), for identifying predominant biological themes of a collection of genes.

After clustering analysis, researchers not only want to determine whether there is a common theme of a particular gene cluster, but also to compare the biological themes among gene clusters. The manual step to choose interesting clusters followed by enrichment analysis on each selected cluster is slow and tedious. To bridge this gap, we designed *clusterProfiler* [2], for comparing and visualizing functional profiles among gene clusters.

2 bitr: Biological Id TranslatoR

Many new R user may find traslating ID is a tedious task and I have received many feedbacks from *clusterProfiler* users that they don't know how to convert gene symbol, uniprot ID or other ID types to Entrez gene ID that used in *clusterProfiler* for most of the species.

To remove this obstacle, We provide `bitr` function for translating among different gene ID types.

```
x <- c("GPX3",  "GLRX",   "LBP",   "CRYAB", "DEFB1", "HCLS1",   "SOD2",   "HSPA2",
       "ORM1",  "IGFBP1", "PTHLH", "GPC3",  "IGFBP3","TOB1",    "MITF",   "NDRG1",
       "NR1H4", "FGFR3",  "PVR",   "IL6",   "PTPRM", "ERBB2",   "NID2",   "LAMB1",
       "COMP",  "PLS3",   "MCAM",  "SPP1",  "LAMC1", "COL4A2",  "COL4A1", "MYOC",
       "ANXA4", "TFPI2",  "CST6",  "SLPI",  "TIMP2", "CPM",     "GGT1",   "NNMT",
       "MAL",   "EEF1A2", "HGD",   "TCN2",  "CDA",   "PCCA",    "CRYM",   "PDXK",
       "STC1",  "WARS",  "HMOX1", "FXYD2", "RBP4",   "SLC6A12", "KDELR3", "ITM2B")
eg = bitr(x, fromType="SYMBOL", toType="ENTREZID", annoDb="org.Hs.eg.db")
head(eg)
```

```
##   SYMBOL ENTREZID
## 1   GPX3     2878
## 2   GLRX     2745
## 3    LBP     3929
## 4  CRYAB     1410
## 5  DEFB1     1672
## 6  HCLS1     3059
```

User should provides an annotation package, both *fromType* and *toType* can accept any types that supported.

User can use `idType` to list all supporting types.

```
idType("org.Hs.eg.db")
```

```
##  [1] "ENTREZID"     "PFAM"         "IPI"          "PROSITE"      "ACCNUM"
##  [6] "ALIAS"        "ENZYME"       "MAP"          "PATH"         "PMID"
## [11] "REFSEQ"       "SYMBOL"       "UNIGENE"      "ENSEMBL"      "ENSEMBLPROT"
## [16] "ENSEMBLTRANS" "GENENAME"     "UNIPROT"      "GO"           "EVIDENCE"
## [21] "ONTOLOGY"     "GOALL"        "EVIDENCEALL"  "ONTOLOGYALL"  "OMIM"
## [26] "UCSCKG"
```

We can translate from one type to other types.

```
ids <- bitr(x, fromType="SYMBOL", toType=c("UNIPROT", "ENSEMBL"), annoDb="org.Hs.eg.db")
head(ids)
```

```
##   SYMBOL    UNIPROT         ENSEMBL
## 1   GPX3     P22352 ENSG00000211445
## 2   GLRX A0A024RAM2 ENSG00000173221
## 3   GLRX     P35754 ENSG00000173221
## 4    LBP     P18428 ENSG00000129988
## 5    LBP     Q8TCF0 ENSG00000129988
## 6  CRYAB     P02511 ENSG00000109846
```

3 Gene Ontology analysis

3.1 Supported organisms

At present, GO analysis in *clusterProfiler* supports about 20 species internally as shown below:

- *Arabidopsis*
- *Anopheles*
- *Bovine*
- *Canine*
- *Chicken*
- *Chimp*
- *Coelicolor*
- *E coli strain K12*
- *E coli strain Sakai*
- *Fly*
- *Gondii*
- *Human*
- *Malaria*
- *Mouse*
- *Pig*
- *Rat*

- *Rhesus*
- *Worm*
- *Xenopus*
- *Yeast*
- *Zebrafish*

For un-supported organisms, user can use their own GO annotation data (in data.frame format with one column of GO and another column of gene ID) and passed it to `buildGOmap` function, which will generate annotation file that suitable for GO analysis in *clusterProfiler*. In future version, we may add functions to help user query annotation from public available database.

3.2 Gene Ontology Classification

In *clusterProfiler*, `groupGO` is designed for gene classification based on GO distribution at a specific level.

```
library("DOSE")
data(geneList)
gene <- names(geneList)[abs(geneList) > 2]
head(gene)
```

```
## [1] "4312"  "8318"  "10874" "55143" "55388" "991"
```

```
ggo <- groupGO(gene     = gene,
               organism = "human",
               ont      = "BP",
               level    = 3,
               readable = TRUE)
head(summary(ggo))
```

```
##                    ID                                Description Count GeneRatio
## GO:0019953 GO:0019953                        sexual reproduction    10    10/207
## GO:0019954 GO:0019954                      asexual reproduction     0     0/207
## GO:0032504 GO:0032504      multicellular organism reproduction     9     9/207
## GO:0032505 GO:0032505 reproduction of a single-celled organism     0     0/207
## GO:0051321 GO:0051321                         meiotic cell cycle     6     6/207
## GO:0006807 GO:0006807      nitrogen compound metabolic process    63    63/207
##
## GO:0019953
## GO:0019954
## GO:0032504
## GO:0032505
## GO:0051321
## GO:0006807 CDC45/MCM10/S100A9/FOXM1/MYBL2/S100A8/TOP2A/NCAPH/E2F8/CXCL10/RRM2/UGT8/NUSAP
```

The input parameters of *gene* is a vector of gene IDs. It expects entrezgene for most of the organisms. For yeast, it should be ORF IDs; *organism* should be the common name of supported species. If *readable*

is setting to TRUE, the input gene IDs will be converted to gene symbols.

3.3 GO over-representation test

Over-representation test [3] is a widely used approach to identify biological themes. Here we implement hypergeometric model to assess whether the number of selected genes associated with disease is larger than expected.

To determine whether any terms annotate a specified list of genes at frequency greater than that would be expected by chance, *clusterProfiler* calculates a p-value using the hypergeometric distribution:

$$p = 1 - \sum_{i=0}^{k-1} \frac{\binom{M}{i}\binom{N-M}{n-i}}{\binom{N}{n}}$$

In this equation, N is the total number of genes in the background distribution, M is the number of genes within that distribution that are annotated (either directly or indirectly) to the node of interest, n is the size of the list of genes of interest and k is the number of genes within that list which are annotated to the node. The background distribution by default is all the genes that have annotation. User can set the background via *universe* parameter.

P-values were adjusted for multiple comparison, and q-values were also calculated for FDR control.

```
ego <- enrichGO(gene          = gene,
                universe      = names(geneList),
                organism      = "human",
                ont           = "CC",
                pAdjustMethod = "BH",
                pvalueCutoff  = 0.01,
                qvalueCutoff  = 0.05,
                readable      = TRUE)
head(summary(ego))
```

```
##                    ID                                Description GeneRatio
## GO:0005819 GO:0005819                                    spindle    24/196
## GO:0005876 GO:0005876                          spindle microtubule    11/196
## GO:0000793 GO:0000793                         condensed chromosome    17/196
## GO:0000779 GO:0000779 condensed chromosome, centromeric region    13/196
## GO:0015630 GO:0015630                      microtubule cytoskeleton    37/196
## GO:0005875 GO:0005875                microtubule associated complex    14/196
##              BgRatio   pvalue p.adjust   qvalue
## GO:0005819 222/11978 1.83e-13 6.00e-11 5.33e-11
## GO:0005876  43/11978 6.19e-11 1.02e-08 9.03e-09
## GO:0000793 147/11978 2.52e-10 2.75e-08 2.45e-08
## GO:0000779  78/11978 3.52e-10 2.88e-08 2.56e-08
## GO:0015630 750/11978 1.15e-09 6.51e-08 5.79e-08
## GO:0005875 103/11978 1.19e-09 6.51e-08 5.79e-08
```

```
##
## GO:0005819                                                                    KI
## GO:0005876
## GO:0000793
## GO:0000779
## GO:0015630 KIF20A/TACC3/CENPE/CHEK1/KIF18B/SKA1/TPX2/KIF4A/ASPM/AK5/BIRC5/KIF11/KIFC1/MA
## GO:0005875
##            Count
## GO:0005819    24
## GO:0005876    11
## GO:0000793    17
## GO:0000779    13
## GO:0015630    37
## GO:0005875    14
```

The input parameter *universe* is the background gene list. If user not explicitly setting this parameter, it will use all the genes that have GO annotation. *pAdjustMethod* specify the method for adjusting p-values. The *pvalueCutoff* parameter is use to restrict the result based on the p-values and the adjusted p values while *qvalueCutoff* is used to control q-values.

3.4 GO Gene Set Enrichment Analysis

A common approach in analyzing gene expression profiles was identifying differential expressed genes that are deemed interesting. The enrichment analysis we demonstrated previous were based on these differential expressed genes. This approach will find genes where the difference is large, but it will not detect a situation where the difference is small, but evidenced in coordinated way in a set of related genes. Gene Set Enrichment Analysis (GSEA) [4] directly addresses this limitation. All genes can be used in GSEA; GSEA aggregates the per gene statistics across genes within a gene set, therefore making it possible to detect situations where all genes in a predefined set change in a small but coordinated way. Since it is likely that many relevant phenotypic differences are manifested by small but consistent changes in a set of genes.

Genes are ranked based on their phenotypes. Given a priori defined set of gens S (e.g., genes shareing the same *GO* or *KEGG* category), the goal of GSEA is to determine whether the members of S are randomly distributed throughout the ranked gene list (L) or primarily found at the top or bottom.

There are three key elements of the GSEA method:

- Calculation of an Enrichment Score.
  The enrichment score (ES) represent the degree to which a set S is over-represented at the top or bottom of the ranked list L. The score is calculated by walking down the list L, increasing a running-sum statistic when we encounter a gene in S and decreasing when it is not. The magnitude of the increment depends on the gene statistics (e.g., correlation of the gene with phenotype). The ES is the maximum deviation from zero encountered in the random walk; it

corresponds to a weighted Kolmogorov-Smirnov-like statistic [4].

- Esimation of Significance Level of *ES*.
  The *p-value* of the *ES* is calculated using permutation test. Specifically, we permute the gene labels of the gene list *L* and recompute the *ES* of the gene set for the permutated data, which generate a null distribution for the *ES*. The *p-value* of the observed ES is then calculated relative to this null distribution.
- Adjustment for Multiple Hypothesis Testing.
  When the entire *GO* or *KEGG* gene sets is evaluated, *clusterProfiler* adjust the estimated significance level to account for multiple hypothesis testing and also *q-values* were calculated for FDR control.

```
ego2 <- gseGO(geneList     = geneList,
              organism     = "human",
              ont          = "CC",
              nPerm        = 100,
              minGSSize    = 120,
              pvalueCutoff = 0.01,
              verbose      = FALSE)
head(summary(ego2))
```

```
## [1] ID              Description     setSize         enrichmentScore
## [5] pvalue          p.adjust        qvalues
## <0 rows> (or 0-length row.names)
```

GSEA use permutation test, user can set *nPerm* for number of permutations. Gene Set size below *minGSSize* will be omitted.

3.5 GO Semantic Similarity Analysis

GO semantic similarity can be calculated by *GOSemSim* [1]. We can use it to cluster genes/proteins into different clusters based on their functional similarity and can also use it to measure the similarities among GO terms to reduce the redundancy of GO enrichment results.

4 KEGG analysis

The annotation package, KEGG.db, is not updated since 2012. It's now pretty old and in *clusterProfiler*, `enrichKEGG` supports downloading latest online version of KEGG data for enrichment analysis. Using KEGG.db is also supported by explicitly setting *use_internal_data* parameter to `TRUE`, but it's not recommended.

With this new feature, organism is not restricted to those supported in previous release, it can be any species that have KEGG annotation data available in KEGG database. User should pass abbreviation of academic name to the `organism` parameter. The full list of KEGG supported organisms can be accessed via http://www.genome.jp/kegg/catalog/org_list.html.

4.1 KEGG over-representation test

To speed up the compilation of this document, we set *use_internal_data = TRUE*.

```
kk <- enrichKEGG(gene         = gene,
                 organism     = "human",
                 pvalueCutoff = 0.05,
                 readable     = TRUE,
                 use_internal_data = TRUE)
head(summary(kk))
```

```
##                ID                                Description GeneRatio  BgRatio
## hsa04110 hsa04110                                  Cell cycle     11/74 128/5894
## hsa04114 hsa04114                              Oocyte meiosis     10/74 114/5894
## hsa03320 hsa03320                   PPAR signaling pathway      7/74  70/5894
## hsa04914 hsa04914 Progesterone-mediated oocyte maturation      6/74  87/5894
## hsa04115 hsa04115                     p53 signaling pathway      5/74  69/5894
## hsa04062 hsa04062              Chemokine signaling pathway      8/74 189/5894
##             pvalue p.adjust   qvalue
## hsa04110 4.31e-07 4.83e-05 4.76e-05
## hsa04114 1.25e-06 7.01e-05 6.92e-05
## hsa03320 2.35e-05 8.78e-04 8.66e-04
## hsa04914 7.21e-04 2.02e-02 1.99e-02
## hsa04115 1.64e-03 3.67e-02 3.62e-02
## hsa04062 2.37e-03 4.43e-02 4.37e-02
##                                                                 geneID Count
## hsa04110 CDC45/CDC20/CCNB2/CCNA2/CDK1/MAD2L1/TTK/CHEK1/CCNB1/MCM5/PTTG1    11
## hsa04114     CDC20/CCNB2/CDK1/MAD2L1/CALML5/AURKA/CCNB1/PTTG1/ITPR1/PGR    10
## hsa03320                         MMP1/FADS2/ADIPOQ/PCK1/FABP4/HMGCS2/PLIN1     7
## hsa04914                                 CCNB2/CCNA2/CDK1/MAD2L1/CCNB1/PGR     6
## hsa04115                                      CCNB2/RRM2/CDK1/CHEK1/CCNB1     5
## hsa04062            CXCL10/CXCL13/CXCL11/CXCL9/CCL18/CCL8/CXCL14/CX3CR1     8
```

4.2 KEGG Gene Set Enrichment Analysis

```
kk2 <- gseKEGG(geneList     = geneList,
               organism     = "human",
               nPerm        = 100,
               minGSSize    = 120,
               pvalueCutoff = 0.01,
               verbose      = FALSE,
               use_internal_data = TRUE)
head(summary(kk2))
```

```
## [1] ID              Description     setSize         enrichmentScore
```

```
## [5] pvalue          p.adjust        qvalues
## <0 rows> (or 0-length row.names)
```

5 Disease Ontology analysis

DOSE [5] supports Disease Ontology (DO) Semantic and Enrichment analysis, please refer to the package vignettes. The `enrichDO` function is very useful for identifying disease association of interesting genes, and function `gseAnalyzer` function is designed for gene set enrichment analysis of *DO*.

6 Reactome pathway analysis

With the demise of KEGG (at least without subscription), the KEGG pathway data in Bioconductor will not update and we encourage user to analyze pathway using *ReactomePA* which use Reactome as a source of pathway data. The function call of `enrichPathway` and `gsePathway` in *ReactomePA* is consistent with `enrichKEGG` and `gseKEGG`.

7 DAVID functional analysis

clusterProfiler provides enrichment and GSEA analysis with GO, KEGG, DO and Reactome pathway supported internally, some user may prefer GO and KEGG analysis with DAVID [6] and still attracted by the visualization methods provided by *clusterProfiler* [**?**]. To bridge the gap between DAVID and clusterProfiler, we implemented `enrichDAVID`. This function query enrichment analysis result from DAVID webserver via RDAVIDWebService [7] and stored the result as an `enrichResult` instance, so that we can use all the visualization functions in *clusterProfiler* to visualize DAVID results. `enrichDAVID` is fully compatible with `compareCluster` function and comparing enrichment results from different gene clusters is now available with DAVID.

```
david <- enrichDAVID(gene = gene,
                     idType = "ENTREZ_GENE_ID",
                     listType = "Gene",
                     annotation = "KEGG_PATHWAY")
```

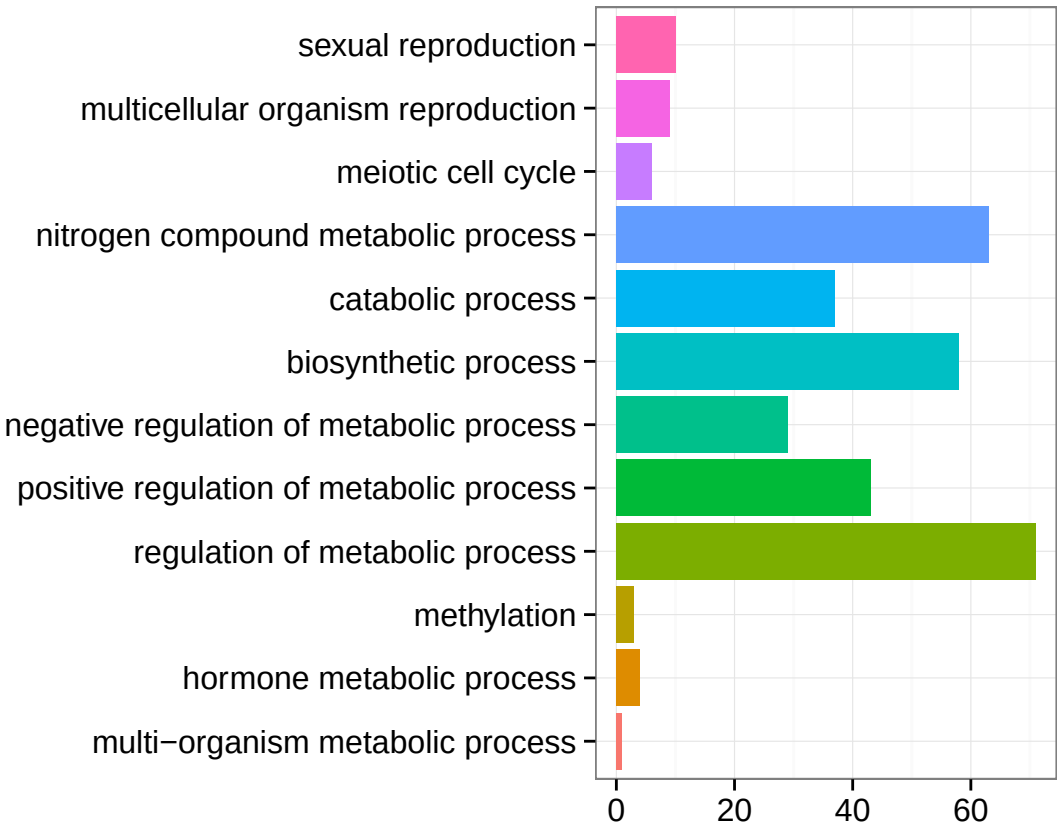
8 Visualization

The function calls of `groupGO`, `enrichGO`, `enrichKEGG`, `enrichDO` and `enrichPathway` are consistent and all the output can be visualized by bar plot, enrichment map and category-gene-network plot. It

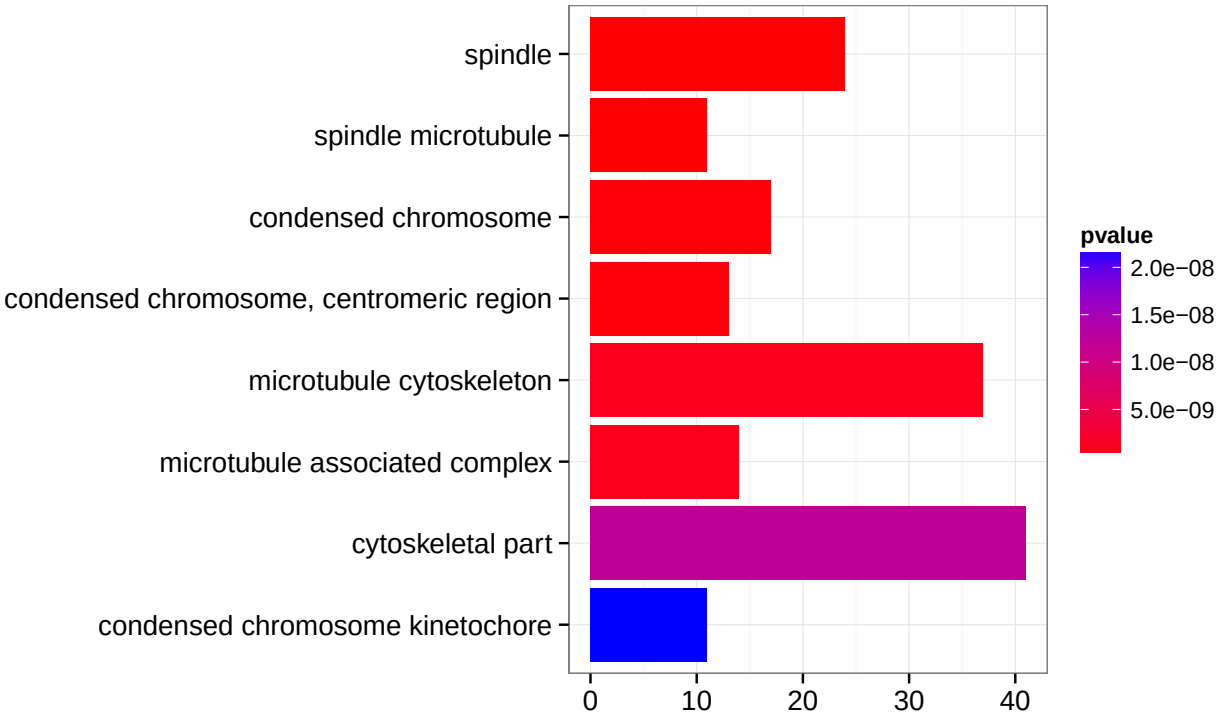
is very common to visualize the enrichment result in bar or pie chart. We believe the pie chart is misleading and only provide bar chart.

8.1 barplot

```
barplot(ggo, drop=TRUE, showCategory=12)
```




```
barplot(ego, showCategory=8)
```



8.2 enrichMap

Enrichment map can be viusalized by `enrichMap`, which also support results obtained from hypergeometric test and gene set enrichment analysis.

```
enrichMap(ego)
```

8.3 cnetplot

In order to consider the potentially biological complexities in which a gene may belong to multiple annotation categories and provide information of numeric changes if available, we developed `cnetplot` function to extract the complex association.

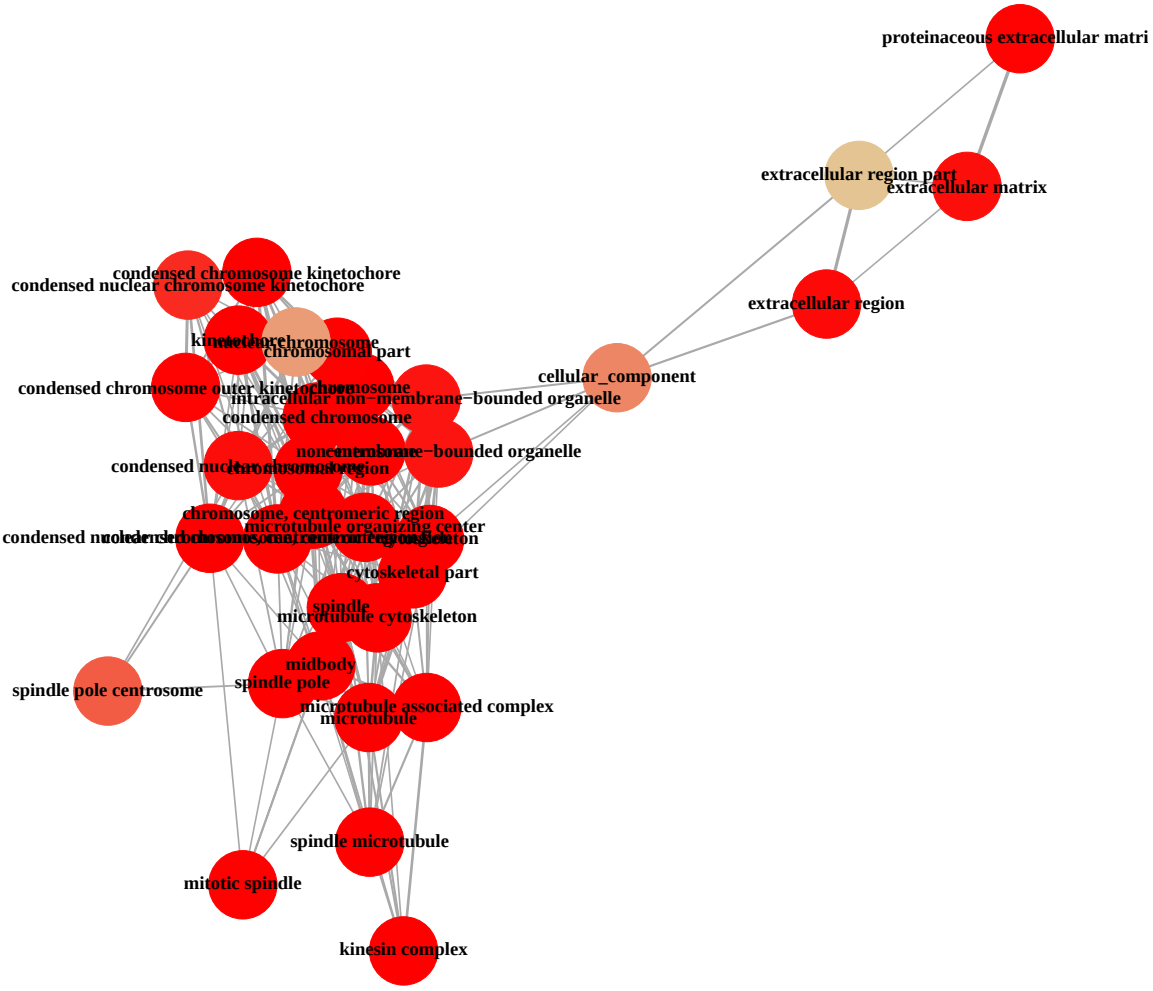
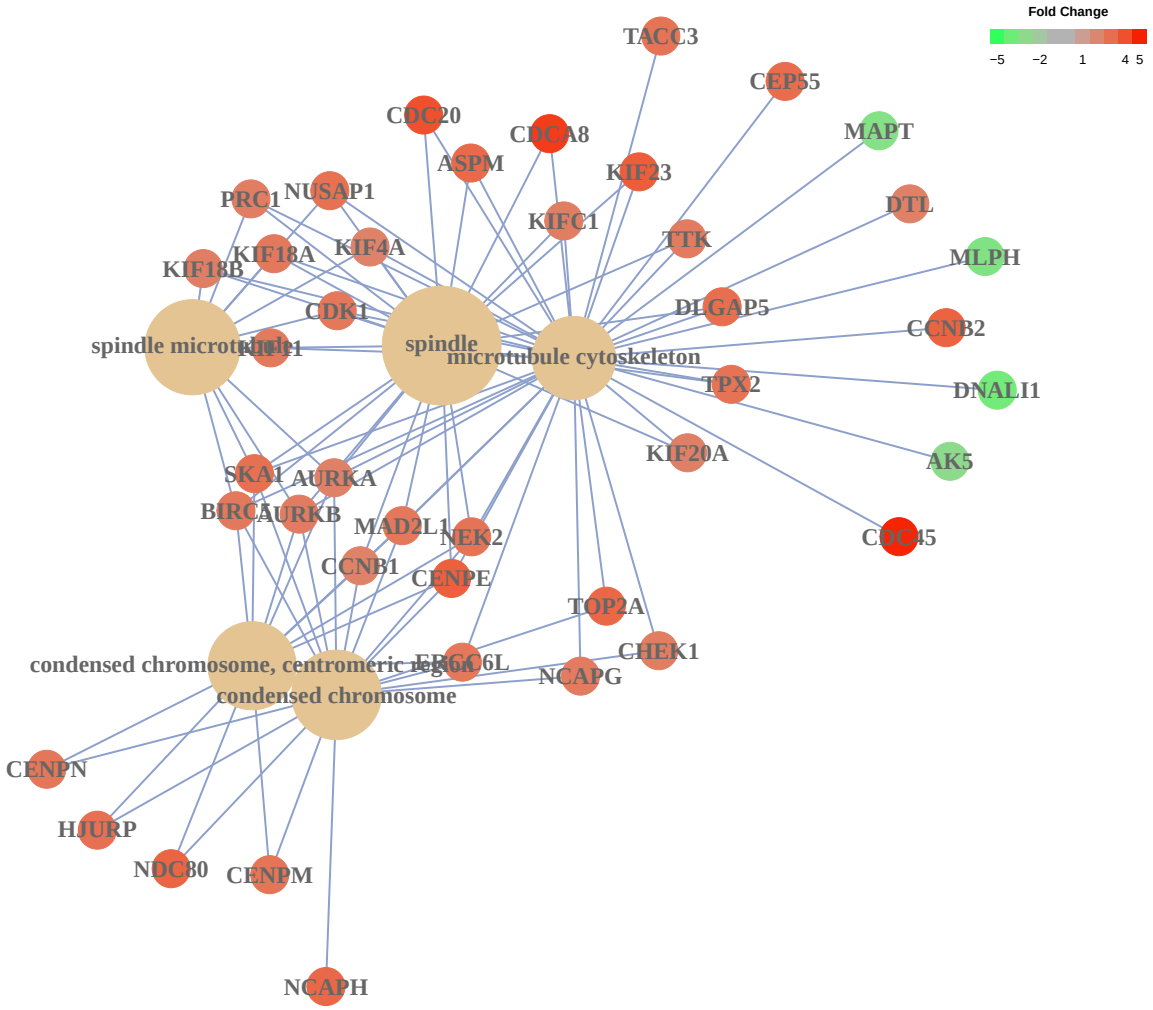
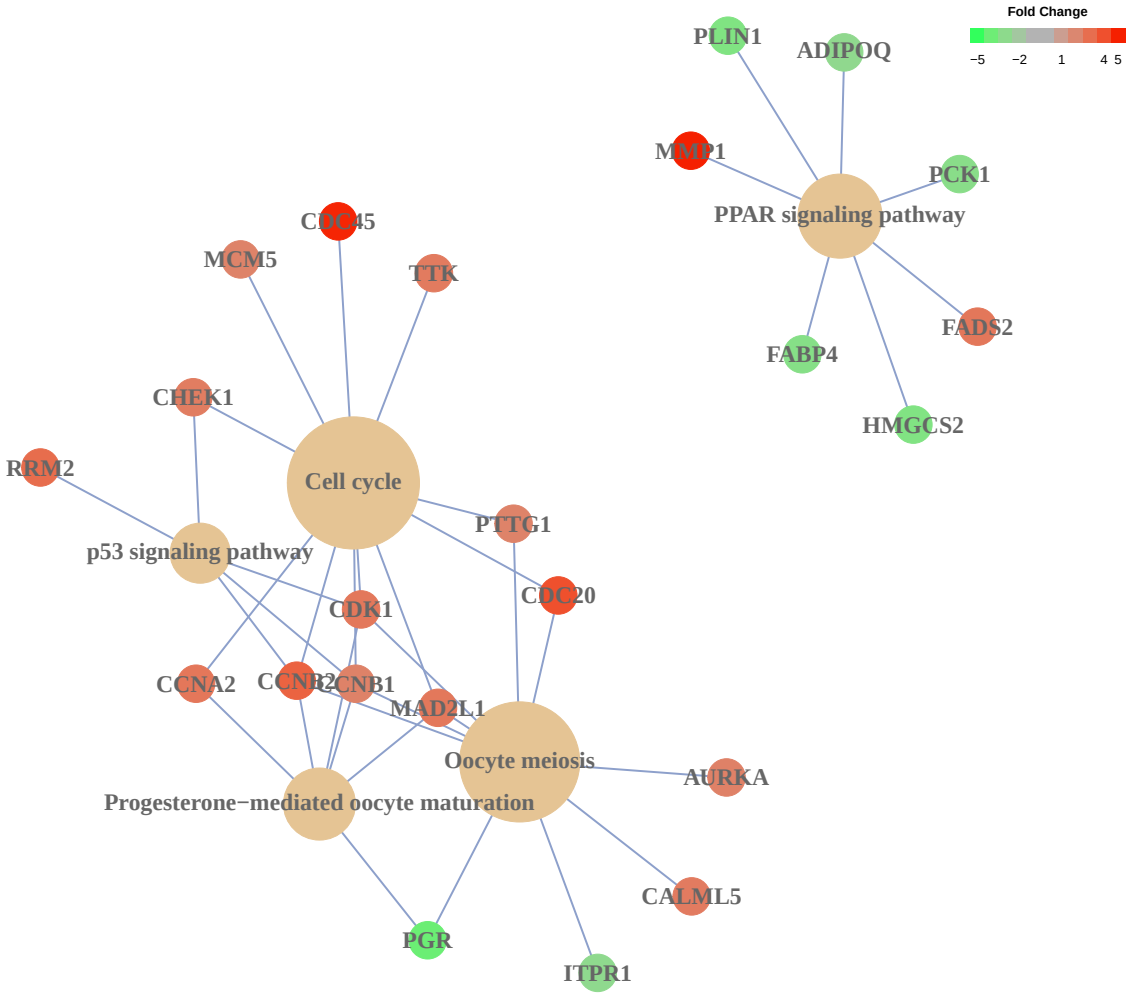


Figure 1: enrichment map of enrichment result

```
cnetplot(ego, categorySize="pvalue", foldChange=geneList)
```




```
cnetplot(kk, categorySize="geneNum", foldChange=geneList)
```



8.4 gseaplot

Running score of gene set enrichment analysis and its association of phenotype can be visualized by `gseaplot`.

```
gseaplot(kk2, geneSetID = "hsa04145")
```

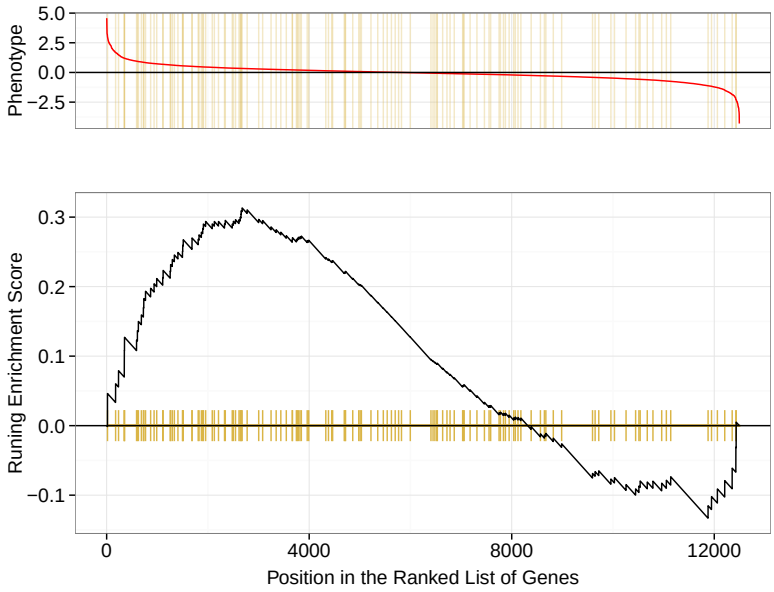


Figure 2: plotting gsea result

8.5 pathview from pathview package

clusterProfiler users can also use `pathview` from the *pathview* [8] to visualize KEGG pathway.

The following example illustrate how to visualize "hsa04110" pathway, which was enriched in our previous analysis.

```
library("pathview")
hsa04110 <- pathview(gene.data  = geneList,
                     pathway.id = "hsa04110",
                     species    = "hsa",
                     limit      = list(gene=max(abs(geneList)), cpd=1))
```

For further information, please refer to the vignette of *pathview* [8].

9 Biological theme comparison

clusterProfiler was developed for biological theme comparison [2], and it provides a function, `compareCluster`, to automatically calculate enriched functional categories of each gene clusters.

```
data(gcSample)
lapply(gcSample, head)
```

```
## $X1
## [1] "4597"  "7111"  "5266"  "2175"  "755"   "23046"
##
```

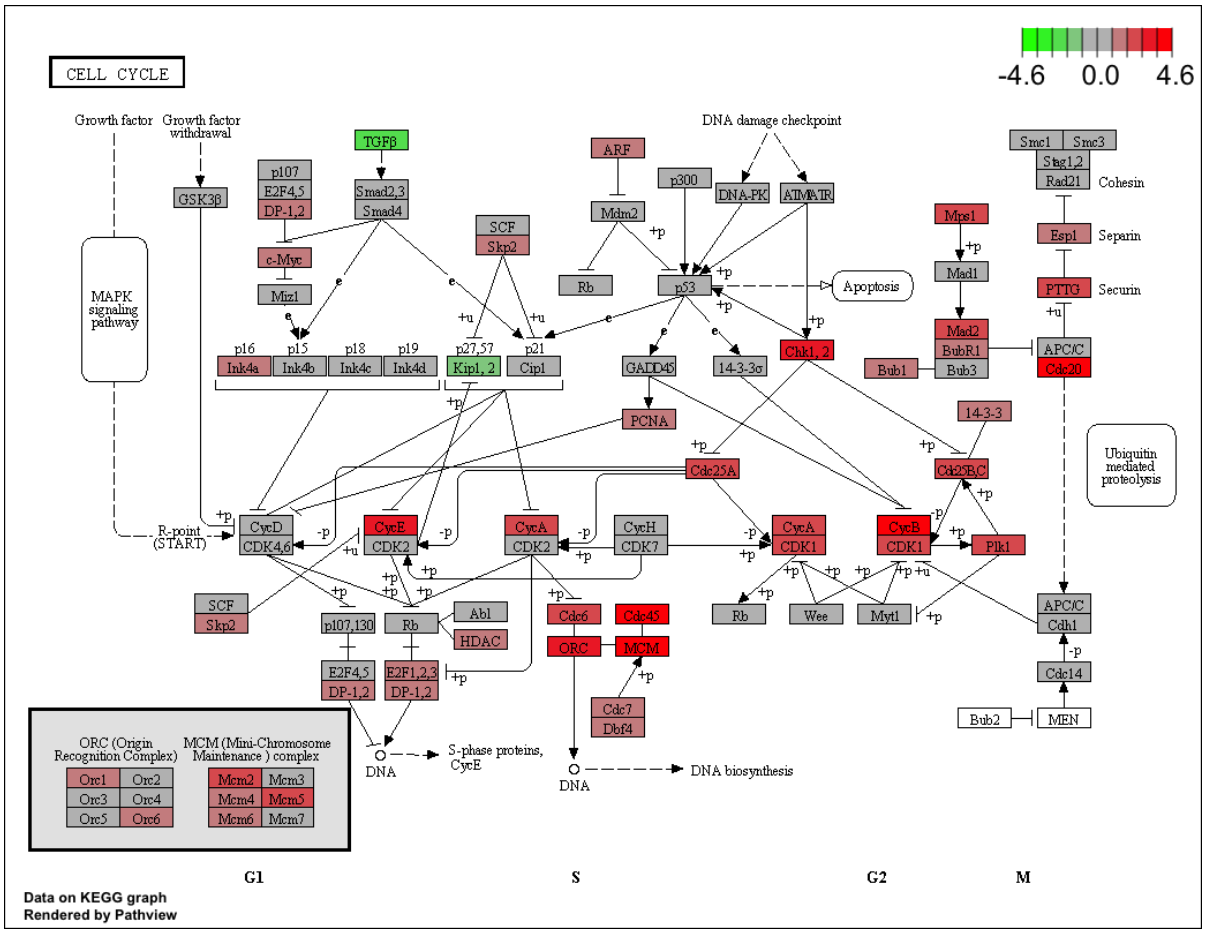


Figure 3: visualize KEGG pathway using pathview

```
## $X2
## [1] "23450" "5160"  "7126"  "26118" "8452"  "3675"
##
## $X3
## [1] "894"   "7057"  "22906" "3339"  "10449" "6566"
##
## $X4
## [1] "5573"  "7453"  "5245"  "23450" "6500"  "4926"
##
## $X5
## [1] "5982" "7318" "6352" "2101" "8882" "7803"
##
## $X6
## [1] "5337"  "9295"  "4035"  "811"   "23365" "4629"
##
## $X7
## [1] "2621" "2665" "5690" "3608" "3550" "533"
```

```
##
## $X8
## [1] "2665" "4735" "1327" "3192" "5573" "9528"
```

The input for *geneCluster* parameter should be a named list of gene IDs.

```
ck <- compareCluster(geneCluster = gcSample, fun = "enrichKEGG", use_internal_data = TRUE)
head(summary(ck))
```

```
##   Cluster       ID              Description GeneRatio  BgRatio   pvalue
## 1      X2 hsa04110               Cell cycle    18/297 128/5894 6.38e-05
## 2      X2 hsa05340 Primary immunodeficiency     8/297  35/5894 2.70e-04
## 3      X3 hsa04512 ECM-receptor interaction     9/152  85/5894 3.08e-04
## 4      X4 hsa03030          DNA replication    10/326  36/5894 1.63e-05
## 5      X4 hsa04110               Cell cycle    20/326 128/5894 2.00e-05
## 6      X4 hsa00240    Pyrimidine metabolism    16/326  99/5894 8.99e-05
##   p.adjust  qvalue
## 1  0.01206 0.01142
## 2  0.02550 0.02415
## 3  0.04739 0.04276
## 4  0.00188 0.00159
## 5  0.00188 0.00159
## 6  0.00428 0.00362
##
## 1          991/1869/890/1871/701/990/10926/9088/8317/9700/9134/1029/2810/699/11200/23594
## 2                                                                          100/6891/3932/973/916/925
## 3                                                                  7057/3339/3695/1101/3679/3910/3696
## 4                                                          5425/4172/4175/4171/10535/5984/2237/4176/
## 5 6500/9184/4172/994/4175/4171/1387/10274/8697/902/4616/5591/4176/8881/7043/983/1022/102
## 6                5425/7296/4860/6241/7298/5440/7372/5430/9583/4832/54107/953/5435/1635/5
##   Count
## 1    18
## 2     8
## 3     9
## 4    10
## 5    20
## 6    16
```

9.1 Formula interface of compareCluster

`compareCluster` also supports passing a formula[1] of type $Entrez \sim group$ or $Entrez \sim group + othergroup$.

[1] The code to support formula has been contributed by Giovanni Dall'Olio.

```
## formula interface
mydf <- data.frame(Entrez=c('1', '100', '1000', '100101467',
                        '100127206', '100128071'),
                   group = c('A', 'A', 'A', 'B', 'B', 'B'),
                   othergroup = c('good', 'good', 'bad', 'bad',
                       'good', 'bad'))
xx.formula <- compareCluster(Entrez~group, data=mydf, fun='groupGO')
head(summary(xx.formula))
```

```
##   Cluster         ID          Description Count GeneRatio     geneID
## 1       A GO:0016020             membrane     2       2/3   100/1000
## 2       A GO:0005576 extracellular region     3       3/3 1/100/1000
## 3       A GO:0005581      collagen trimer     0       0/3
## 4       A GO:0005623                 cell     2       2/3   100/1000
## 5       A GO:0009295             nucleoid     0       0/3
## 6       A GO:0019012               virion     0       0/3
```

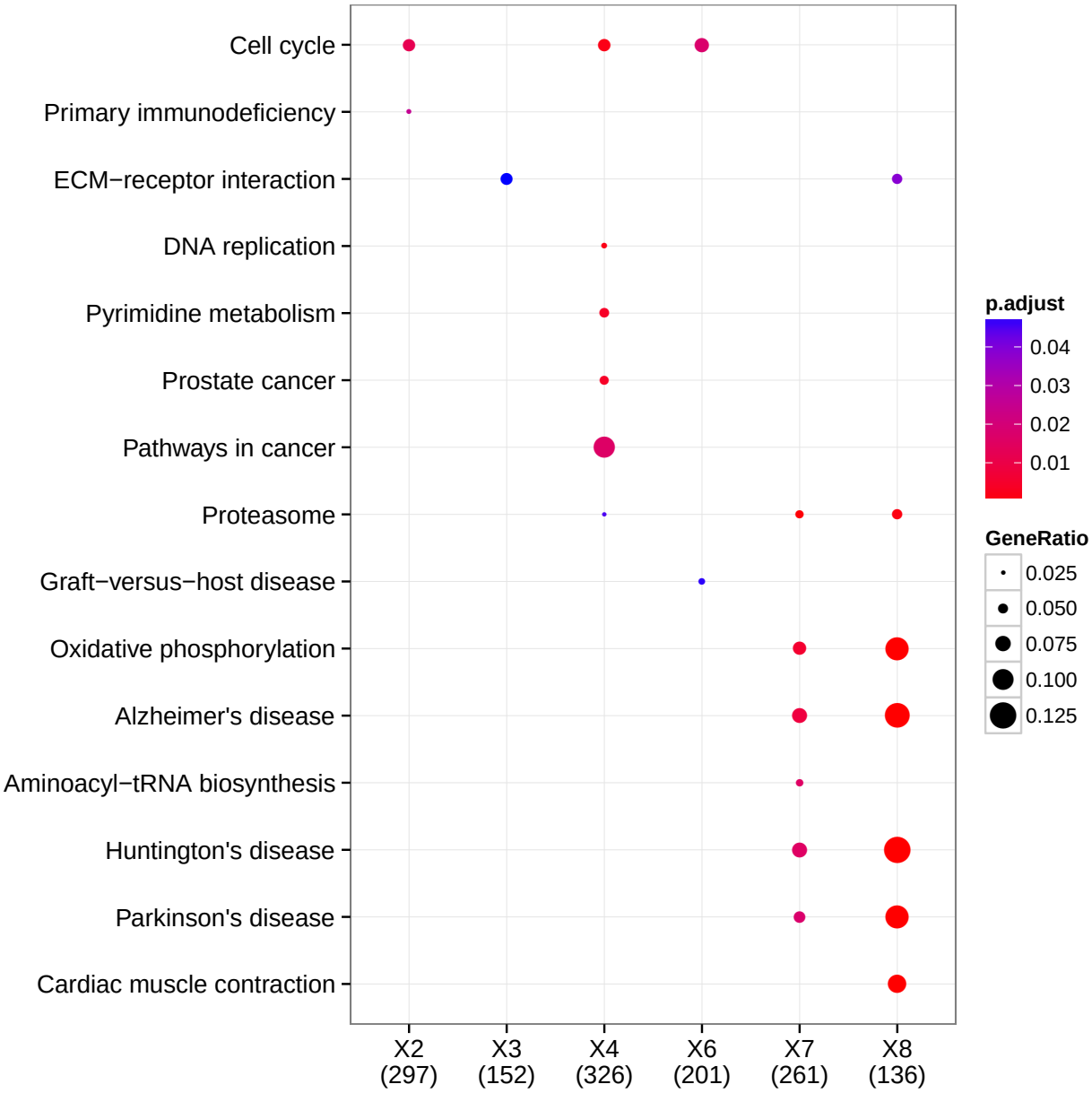
```
## formula interface with more than one grouping variable
xx.formula.twogroups <- compareCluster(Entrez~group+othergroup,
                                       data=mydf, fun='groupGO')
head(summary(xx.formula.twogroups))
```

```
##   Cluster         ID          Description Count GeneRatio geneID
## 1   A.bad GO:0016020             membrane     1       1/1   1000
## 2   A.bad GO:0005576 extracellular region     1       1/1   1000
## 3   A.bad GO:0005581      collagen trimer     0       0/1
## 4   A.bad GO:0005623                 cell     1       1/1   1000
## 5   A.bad GO:0009295             nucleoid     0       0/1
## 6   A.bad GO:0019012               virion     0       0/1
```

9.2 Visualization of profile comparison

We can visualize the result using `plot` method.

```
plot(ck)
```



By default, only top 5 (most significant) categories of each cluster was plotted. User can changes the parameter *showCategory* to specify how many categories of each cluster to be plotted, and if *showCategory* was set to *NULL*, the whole result will be plotted.

The `plot` function accepts a parameter *by* for setting the scale of dot sizes. The default parameter *by* is setting to "geneRatio", which corresponding to the "GeneRatio" column of the output. If it was setting to *count*, the comparison will be based on gene counts, while if setting to *rowPercentage*, the dot sizes will be normalized by *count/(sum of each row)*

To provide the full information, we also provide number of identified genes in each category (numbers

in parentheses) when *by* is setting to *rowPercentage* and number of gene clusters in each cluster label (numbers in parentheses) when *by* is setting to *geneRatio*, as shown in Figure 3. If the dot sizes were based on *count*, the row numbers will not shown.

The p-values indicate that which categories are more likely to have biological meanings. The dots in the plot are color-coded based on their corresponding p-values. Color gradient ranging from red to blue correspond to in order of increasing p-values. That is, red indicate low p-values (high enrichment), and blue indicate high p-values (low enrichment). P-values and adjusted p-values were filtered out by the threshold giving by parameter *pvalueCutoff*, and FDR can be estimated by *qvalue*.

User can refer to the example in [2]; we analyzed the publicly available expression dataset of breast tumour tissues from 200 patients (GSE11121, Gene Expression Omnibus) [9]. We identified 8 gene clusters from differentially expressed genes, and using `compareCluster` to compare these gene clusters by their enriched biological process.

Another example was shown in [10], we calculated functional similarities among viral miRNAs using method described in [11], and compared significant KEGG pathways regulated by different viruses using `compareCluster`.

The comparison function was designed as a framework for comparing gene clusters of any kind of ontology associations, not only `groupGO`, `enrichGO`, and `enrichKEGG` provided in this package, but also other biological and biomedical ontologies, for instance, `enrichDO` from *DOSE* [5] and `enrichPathway` from *ReactomePA* work fine with `compareCluster` for comparing biological themes in disease and reactome pathway perspective. More details can be found in the vignettes of *DOSE* [5] and *ReactomePA*.

10 External documents

- Why clusterProfiler fails
- KEGG enrichment analysis with latest online data using clusterProfiler
- DAVID functional analysis with clusterProfiler
- Enrichment map
- a formula interface for GeneOntology analysis

11 Session Information

Here is the output of `sessionInfo()` on the system on which this document was compiled:

- R version 3.2.2 (2015-08-14), `x86_64-pc-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, stats4, utils

- Other packages: AnnotationDbi 1.30.1, Biobase 2.28.0, BiocGenerics 0.14.0, DBI 0.3.1, DOSE 2.6.6, GO.db 3.1.2, GenomeInfoDb 1.4.3, IRanges 2.2.7, RSQLite 1.0.0, S4Vectors 0.6.6, clusterProfiler 2.2.7, org.Hs.eg.db 3.1.2
- Loaded via a namespace (and not attached): BiocStyle 1.6.0, Biostrings 2.36.4, DO.db 2.9, GOSemSim 1.26.0, KEGG.db 3.1.2, KEGGREST 1.8.0, MASS 7.3-44, R6 2.1.1, Rcpp 0.12.1, XVector 0.8.0, colorspace 1.2-6, digest 0.6.8, evaluate 0.8, formatR 1.2.1, ggplot2 1.0.1, grid 3.2.2, gtable 0.1.2, highr 0.5.1, httr 1.0.0, igraph 1.0.1, knitr 1.11, labeling 0.3, magrittr 1.5, munsell 0.4.2, plyr 1.8.3, png 0.1-7, proto 0.3-10, qvalue 2.0.0, reshape2 1.4.1, scales 0.3.0, splines 3.2.2, stringi 0.5-5, stringr 1.0.0, tools 3.2.2, zlibbioc 1.14.0

References

[1] Guangchuang Yu, Fei Li, Yide Qin, Xiaochen Bo, Yibo Wu, and Shengqi Wang. Gosemsim: an r package for measuring semantic similarity among go terms and gene products. *Bioinformatics*, 26(7):976–978, 2010. PMID: 20179076. URL: http://bioinformatics.oxfordjournals.org/cgi/content/abstract/26/7/976, doi:10.1093/bioinformatics/btq064.

[2] Guangchuang Yu, Le-Gen Wang, Yanyan Han, and Qing-Yu He. clusterprofiler: an r package for comparing biological themes among gene clusters. *OMICS: A Journal of Integrative Biology*, 16(5):284–287, May 2012. URL: http://online.liebertpub.com/doi/abs/10.1089/omi.2011.0118, doi:10.1089/omi.2011.0118.

[3] Elizabeth I Boyle, Shuai Weng, Jeremy Gollub, Heng Jin, David Botstein, J Michael Cherry, and Gavin Sherlock. GO::TermFinder–open source software for accessing gene ontology information and finding significantly enriched gene ontology terms associated with a list of genes. *Bioinformatics (Oxford, England)*, 20(18):3710–3715, December 2004. PMID: 15297299. URL: http://www.ncbi.nlm.nih.gov/pubmed/15297299, doi:10.1093/bioinformatics/bth456.

[4] Aravind Subramanian, Pablo Tamayo, Vamsi K. Mootha, Sayan Mukherjee, Benjamin L. Ebert, Michael A. Gillette, Amanda Paulovich, Scott L. Pomeroy, Todd R. Golub, Eric S. Lander, and Jill P. Mesirov. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(43):15545–15550, October 2005. URL: http://www.pnas.org/content/102/43/15545.abstract, doi:10.1073/pnas.0506580102.

[5] Guangchuang Yu, Li-Gen Wang, Guang-Rong Yan, and Qing-Yu He. DOSE: an r/bioconductor package for disease ontology semantic and enrichment analysis. 31(4):608–609. URL: http://bioinformatics.oxfordjournals.org.eproxy2.lib.hku.hk/content/31/4/608, doi:10.1093/bioinformatics/btu684.

[6] Da Huang, Brad T Sherman, Qina Tan, Jack R Collins, W Gregory Alvord, Jean Roayaei, Robert Stephens, Michael W Baseler, H Clifford Lane, and Richard A Lempicki. The DAVID gene functional classification tool: a novel biological module-centric algorithm to functionally analyze large gene lists. 8(9):R183. URL: http://genomebiology.com/2007/8/9/R183, doi:10.1186/gb-2007-8-9-r183.

[7] Cristbal Fresno and Elmer A. Fernndez. RDAVIDWebService: a versatile r interface to DAVID. 29(21):2810–2811. URL: http://bioinformatics.oxfordjournals.org.eproxy1.lib.hku.hk/content/29/21/2810, doi:10.1093/bioinformatics/btt487.

[8] Weijun Luo and Cory Brouwer. Pathview: an R/Bioconductor package for pathway-based data integration and visualization. 29:1830–1831. PMID: 23740750. URL: http://bioinformatics.oxfordjournals.org/content/29/14/1830, doi:10.1093/bioinformatics/btt285.

[9] Marcus Schmidt, Daniel B?hm, Christian von T?rne, Eric Steiner, Alexander Puhl, Henryk Pilch, Hans-Anton Lehr, Jan G. Hengstler, Heinz K?lbl, and Mathias Gehrmann. The humoral immune system has a key prognostic impact in node-negative breast cancer. *Cancer Research*, 68(13):5405 –5413, July 2008. URL: http://cancerres.aacrjournals.org/content/68/13/5405.abstract, doi:10.1158/0008-5472.CAN-07-5206.

[10] Guangchuang Yu and Qing-Yu He. Functional similarity analysis of human virus-encoded miRNAs. *Journal of Clinical Bioinformatics*, 1(1):15, May 2011. URL: http://www.jclinbioinformatics.com/content/1/1/15, doi:10.1186/2043-9113-1-15.

[11] Guangchuang Yu, Chuan-Le Xiao, Xiaochen Bo, Chun-Hua Lu, Yide Qin, Sheng Zhan, and Qing-Yu He. A new method for measuring functional similarity of microRNAs. *Journal of Integrated OMICS*, 1(1):49–54, February 2011. URL: http://www.jiomics.com/index.php/jio/article/view/21, doi:10.5584/jiomics.v1i1.21.