cogena, a workflow for co-expressed gene-set enrichment analysis

Zhilong Jia, Michael R. Barnes

2018-04-30

Contents

1 Abstract

Co-expressed **g**ene-set **en**richment **a**nalysis, cogena, is a

workflow for gene set enrichment analysis of co-expressed genes. The cogena worklow (Figure 1) proceeds from co-expression analysis using a range of clustering methods, through to gene set enrichment analysis based on a range of pre-defined gene sets. Cogena can be applied to a number of different analytical scenarios dependent on the gene set used. Currently cogena is pre-built with gene sets from Msigdb and Connectivity

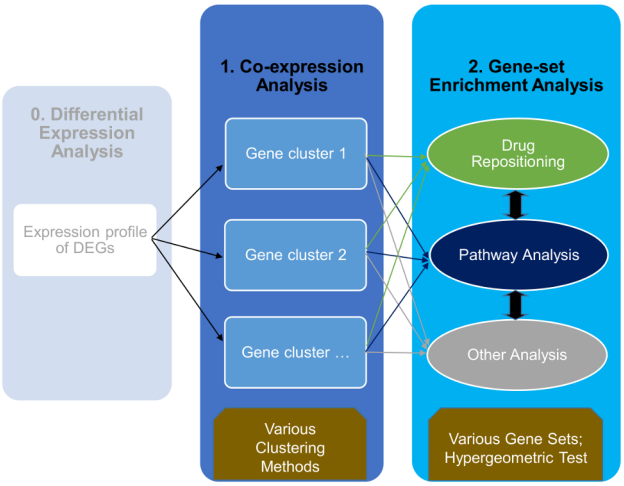

Figure 1: Overview of the cogena workflow

Map, allowing pathway analysis and drug repositioning analysis respectively, the user can also add custom genes sets to expand analytical functionality further.

The following sections outline a typical example of the cogena workflow, describing the input data and typical analysis steps required for pathway analysis and drug repositioning analysis.

2 A quick start

See examples using the `?cogena` command in R.

3 Data Input

```
Note: all the gene names should be gene SYMBOLs since they are used in
the gene set files. Other kinds of gene identifiers can be used
according to the identifiers used in the user-defined gene-set files.
```

3.1 Input data required

- A set of differentially expressed genes: These should be in a matrix with genes in rows and samples in columns, data.frame or ExpressionSet object.
- The sample labels indicating the labels, like control and disease, of each sample. A vector with sample names.

3.2 Example dataset

The `cogena` package has an example dataset, from the NCBI GEO database GSE13355. The samples are derived from lesional and non-lesional skin of psoriasis patients. There are two objects in the Psoriasis dataset. See `?Psoriasis` for more details.

```
library(cogena)
data(Psoriasis)
# objects in the Psoriasis dataset.
# Note: label of interest should follow the control label as this
# will affect the direction of gene regulation.
# For instance use factor (c("Normal", "Cancer", "Normal"),
# levels=c("Normal", "Cancer")), instead of factor(c("Normal",
# "Cancer","Normal")) since "Cancer" is the label of interest.
```

```
ls()
```

```
##  [1] "DEexprs"                "annoGMT"
##  [3] "annofile"               "clMethods"
##  [5] "clen_res"               "cmapDn100_cogena_result"
##  [7] "enrichment.table"       "gec"
##  [9] "geneC"                  "genecl_result"
## [11] "method"                 "metric"
## [13] "nClust"                 "ncore"
## [15] "sampleLabel"
```

4 Various Analyses

4.1 What kind of analysis can be done?

The gene set used determines the type of analysis.

There are a variety of gene sets in the `cogena` package, partly collected from MSigDB and CMap. Gene sets are summarized in Table 2.

Table 2. Cogena Gene Sets

Gene Set	Description
c2.cp.biocarta.v5.0.symbols.gmt.xz	Biocarta gene sets
c2.cp.kegg.v5.0.symbols.gmt.xz	KEGG gene sets
c2.cp.reactome.v5.0.symbols.gmt.xz	Reactome gene sets
c2.cp.v5.0.symbols.gmt.xz	all canonical pathways
c5.bp.v5.0.symbols.gmt.xz	GO biological processes
c5.mf.v5.0.symbols.gmt.xz	GO molecular functions
CmapDn100.gmt.xz	Connectivity map gene sets: top 100 down regulated per drug
CmapUp100.gmt.xz	Connectivity map gene sets: top 100 up regulated per drug

User-defined gene-sets must be formatted gmt and/or compressed by xz, (such as c2.cp.kegg.v5.0.symbols.gmt or c2.cp.kegg.v5.0.symbols.gmt.xz). Gene sets should be copied to the extdata directory in the installation directory of cogena, such as ~/R/x86_64-pc-linux-gnu- library/3.2/cogena/extdata in Linux), or kindly send to the author of cogena to share with others.

4.2 Types of analyses

- Pahtway Analysis
- GO Analysis
- Drug repositioning
- User defined

5 Pathway Analysis

Firstly, KEGG Pathway Analysis, will be demonstrated to show the utility of cogena. The other analyses based on cogena are similar to the process of pathway analysis. Here we used the KEGG pathway gene set (c2.cp.kegg.v5.0.symbols.gmt.xz), hierarchical and Pam clustering methods, 10 clusters, 2 CPU cores and "correlation" distance metric to set up the pathway analysis.

5.1 Parameter setting

```
# KEGG Pathway gene set
annoGMT <- "c2.cp.kegg.v5.0.symbols.gmt.xz"
# GO biological process gene set
# annoGMT <- "c5.bp.v5.0.symbols.gmt.xz"
annofile <- system.file("extdata", annoGMT, package="cogena")

# the number of clusters. It can be a vector.
# nClust <- 2:20
nClust <- 10

# Making factor "Psoriasis" behind factor "ct" means Psoriasis Vs Control

# is up-regualted
sampleLabel <- factor(sampleLabel, levels=c("ct", "Psoriasis"))

# the number of cores.
# ncore <- 8
ncore <- 2

# the clustering methods
# clMethods <- c("hierarchical","kmeans","diana","fanny","som","model",
# "sota","pam","clara","agnes") # All the methods can be used together.
clMethods <- c("hierarchical","pam")



# the distance metric
metric <- "correlation"

# the agglomeration method used for hierarchical clustering
# (hierarchical and agnes)
method <- "complete"
```

5.2 Cogena running

There are two steps for cogena analysis, co-expression analysis and then gene set enrichment analysis (here is pathway anlysis).

```
# Co-expression Analysis
genecl_result <- coExp(DEexprs, nClust=nClust, clMethods=clMethods,
                       metric=metric, method=method, ncore=ncore)


# Enrichment (Pathway) analysis for the co-expressed genes
clen_res <- clEnrich(genecl_result, annofile=annofile, sampleLabel=sampleLabel)
```

5.3 Results of pathway analysis

5.3.1 Summary of cogena result

After completing the cogena analysis, the user can use `summary` to see the summary of the result of cogena. And `enrichment` caculates the enrichment score of certain clustering methods and certain numbers of cluster.

Cogena does not automatically set the clustering method or the number of clusters. Here we show some principles to guide the user towards optimal selection of method and number of clusters:

- Different clusters should account for different gene sets.
- A gene set should enriched only in one cluster but not two or more.
- The number of genes in a gene set enriched cluster should be the smallest small possible to achieve the highest enrichment score.

```
summary(clen_res)
```

```
##
## Clustering Methods:
##  hierarchical pam
##
## The Number of Clusters:
##  10
##
## Metric of Distance Matrix:
##  correlation
##
## Agglomeration method for hierarchical clustering (hclust and agnes):
##  complete
##
## Gene set:
##  c2.cp.kegg.v5.0.symbols.gmt.xz
```

```
# Here we consider the "pam" method and 10 clusters.
# Always make the number as character, please!
enrichment.table <- enrichment(clen_res, "pam", "10")
```

5.3.2 Heatmap of expression profiling with clusters

`heatmapCluster` is developed to show the co-expression of differentially expressed genes. Figure 2 produced by `heatmapCluster` is an enhanced heatmap with co-expression information. Moreover, it is obvious to know which cluster contains up-regulated or down-regulated genes based on the colour.

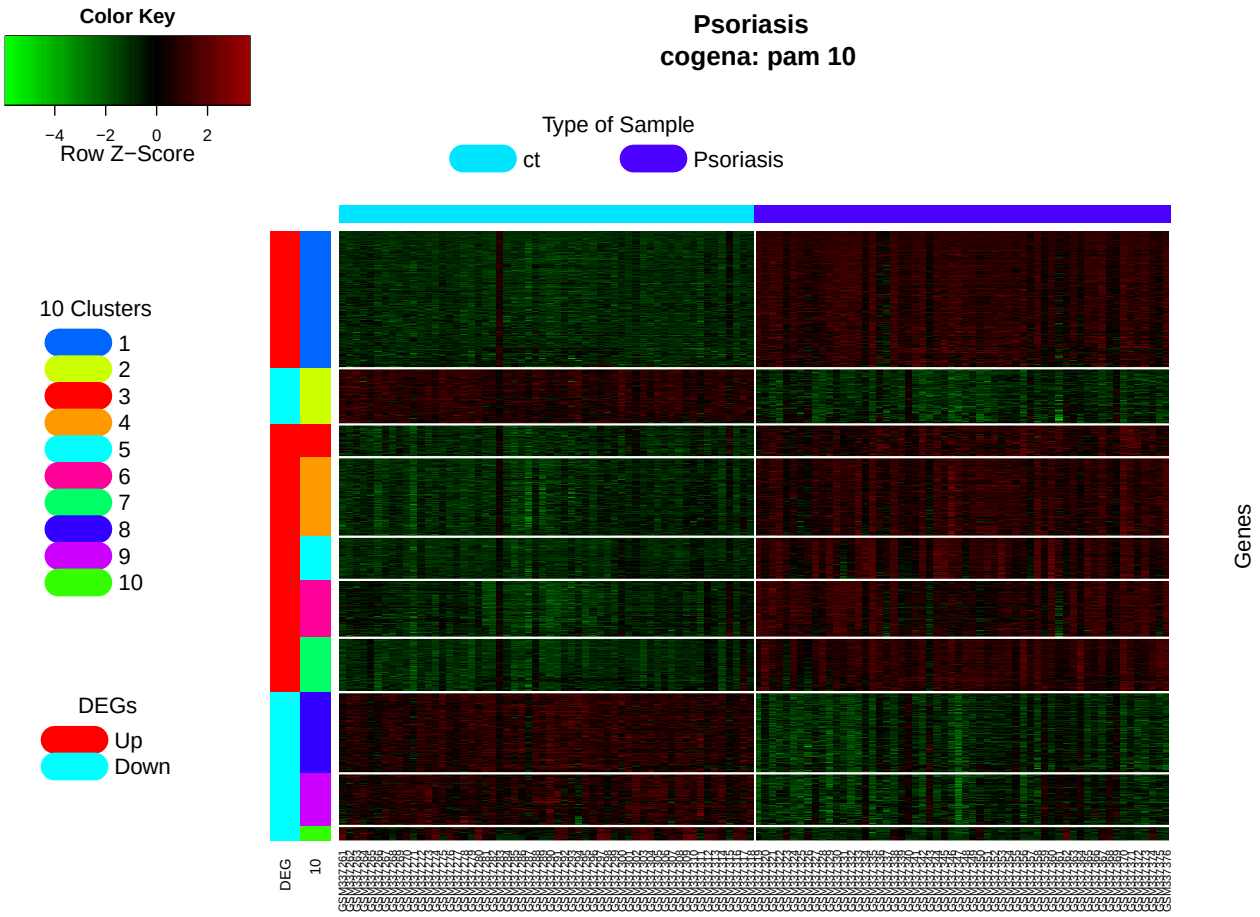

Figure 2: Heatmap of expression profiling with clusters

```
# Always make the number as character, please!
heatmapCluster(clen_res, "pam", "10", maintitle="Psoriasis")
```

```
## The number of genes in each cluster:
## upDownGene
##   1   2
## 468 238
## cluster_size
##   1   2   3   4   5   6   7   8   9  10
## 158  65  38  92  50  67  63  94  61  18
```

5.3.3 Enrichment heatmap of co-expressed genes

`heatmapPEI` can be used to show the enrichment graph. See Figure 3. See `?heatmapPEI` for more details. Many parameters are configurable, while generally the default will be fine.

```
# The enrichment score for 10 clusters, together with Down-regulated,
# Up-regulated and All DE genes. The values shown in Figure 2 is the -log2(FDR).
#
# Always make the number as character, please!
```

```
heatmapPEI(clen_res, "pam", "10", printGS=FALSE, maintitle="Pathway analysis for Psoriasis")
```

```
## Warning: Ignoring unknown aesthetics: fill
```

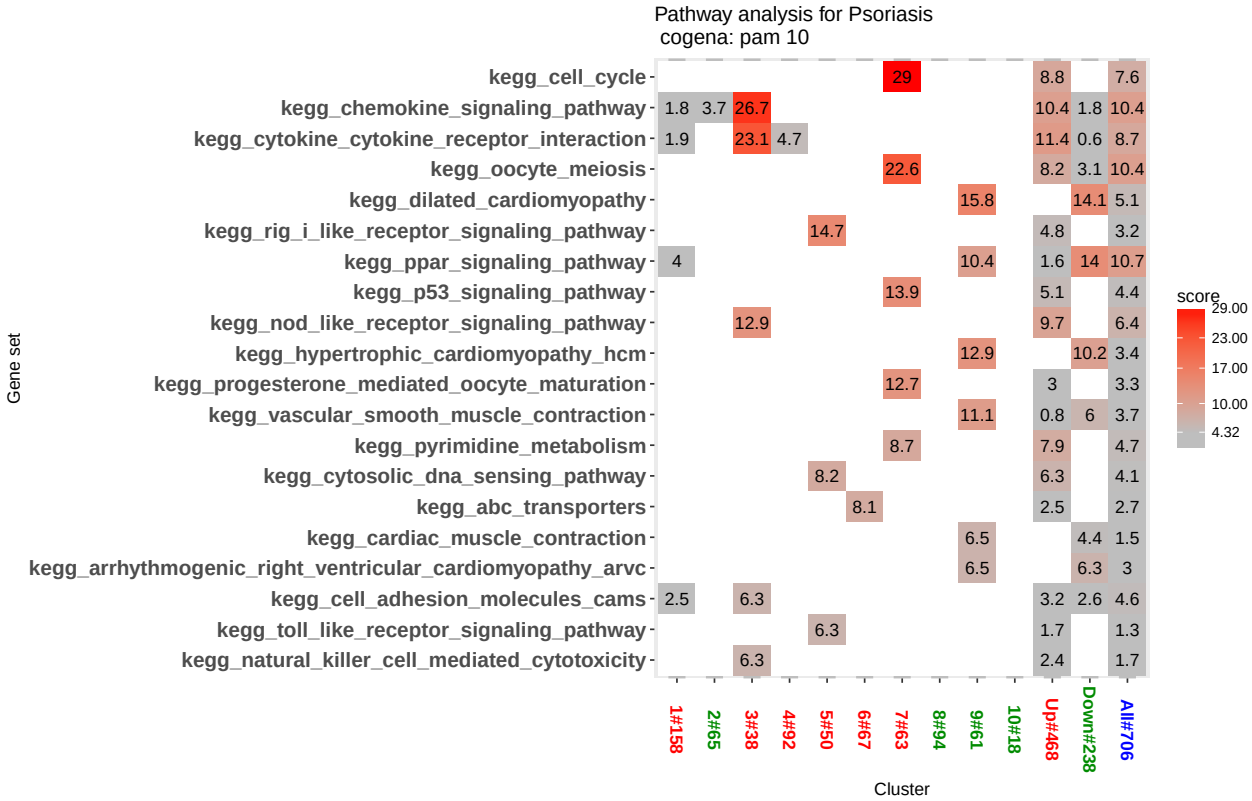

Figure 3 shows the pathway enrichment for each cluster as well as up-regulated, down-regulated and all the differentially expressed genes. The enrichment scores can be ranked by a certain cluster or the max or average scores of all the scores for each pathway.

6 Drug repositioning

Pathway analysis demonstrates that specific disease pathways are often represented by a single cluster. Accordingly, we recommend that drug repositioning is performed based on co-expressed gene clusters instead of all the differentially expressed genes. If the input of cogena is disease related data, the drugs enriched should recover the gene expression changed by the disease (the drug should induce an opposite direction in expression to the disease), while if the input is drug related, the enriched drugs should show similar gene expression changes caused by the drug studied. Here we show drugs for treating psoriasis, an autoimmune disease.

6.1 Drug repositioning analysis running

The drug repositioning gene set choice of *CmapDn100.gmt.xz* or *CmapUp100.gmt.xz* should be made based on the regulation direction of clusters. For example, as the 7th cluster contains up-regulated genes for psoriasis, the *CmapDn100.gmt.xz* is chosen for drug repositioning of psoriasis to recover the gene expression changes caused by the disease.

```
# A comprehensive way
# cmapDn100_cogena_result <- clEnrich(genecl_result,
```

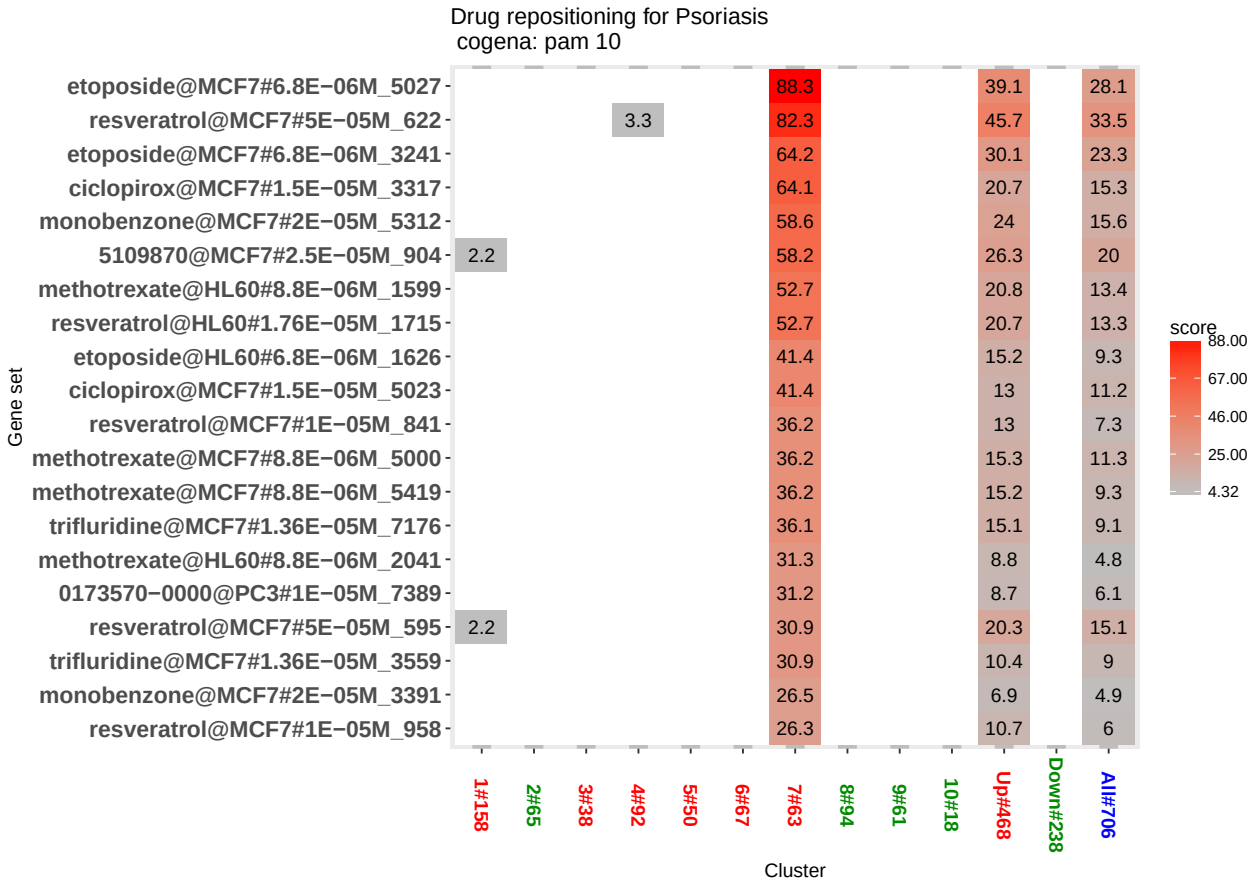

Figure 3: Drug repositioning

```
# annofile=system.file("extdata", "CmapDn100.gmt.xz", package="cogena"),
# sampleLabel=sampleLabel)

# A quick way
# Based on the pathway analysis results
cmapDn100_cogena_result <- clEnrich_one(genecl_result, "pam", "10",
    annofile=system.file("extdata", "CmapDn100.gmt.xz", package="cogena"),
    sampleLabel=sampleLabel)
```

6.2 Original result of drug repositioning

Showing the results ordered by the 7th cluster in Figure 5. The parameter `orderMethod` is used to order the results.

```
heatmapPEI(cmapDn100_cogena_result, "pam", "10", printGS=FALSE,
           orderMethod = "7", maintitle="Drug repositioning for Psoriasis")
```

```
## Warning: Ignoring unknown aesthetics: fill
```

```
# Results based on cluster 5.
# heatmapPEI(cmapDn100_cogena_result, "pam", "10", printGS=FALSE,
#           orderMethod = "5", maintitle="Drug repositioning for Psoriasis")
```

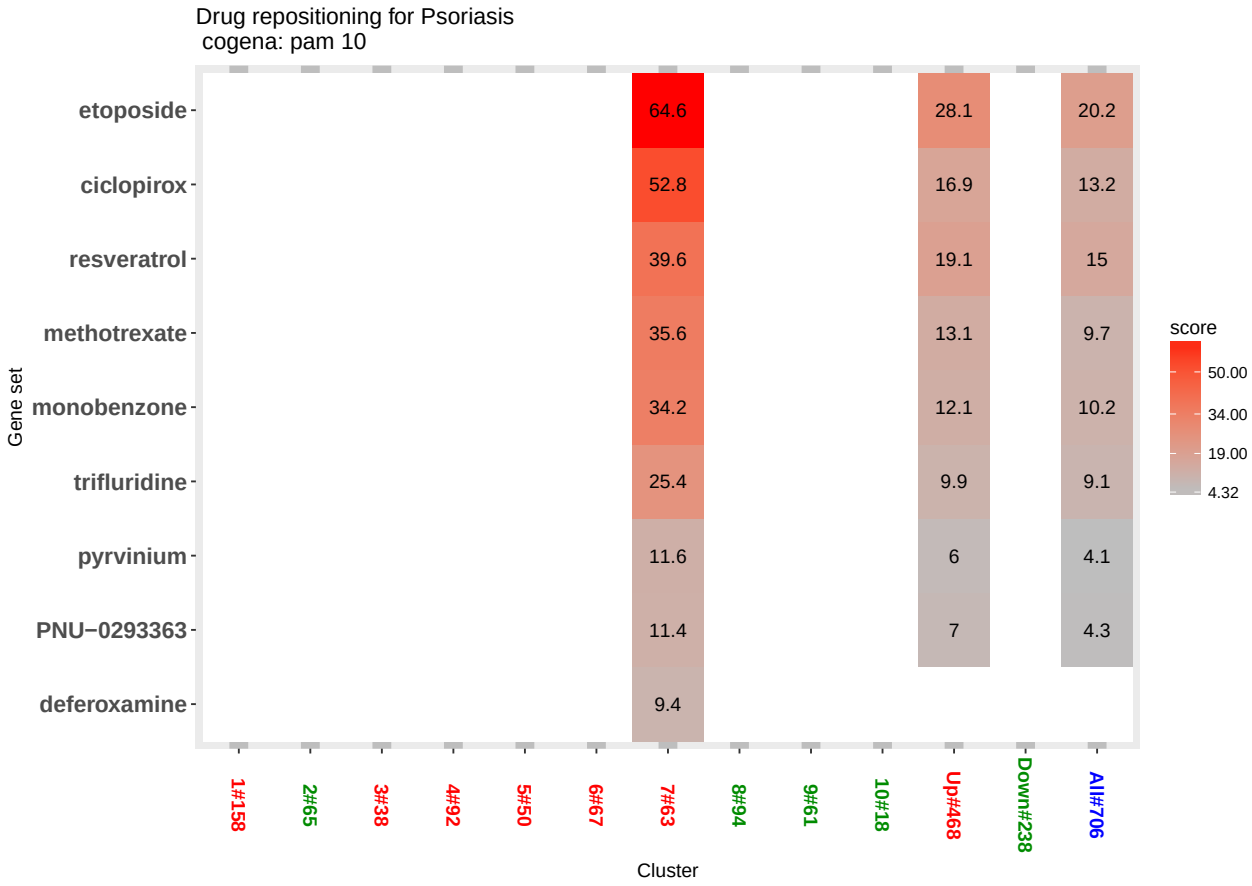

Figure 4: Drug repositioning (multi-instance merged)

```
# Results based on cluster 9, containing down-regulated genes.
# heatmapPEI(cmapUp100_cogena_result, "pam", "10", printGS=FALSE,
#            orderMethod = "9", maintitle="Drug repositioning for Psoriasis")
```

6.3 Multi-instance merged result of drug repositioning

Usually there is more than one instance for a drug with different doses or time-points in the Cmap gene set. `heatmapCmap` can merge the multi-instance results based on parameter *mergeMethod* ("mean" or "max"). Figure 6 shows the multi-instance merged results ordered by the 7th cluster.

```
heatmapCmap(cmapDn100_cogena_result, "pam", "10", printGS=FALSE,
            orderMethod = "7", maintitle="Drug repositioning for Psoriasis")
```

```
## `summarise_each()` is deprecated.
## Use `summarise_all()`, `summarise_at()` or `summarise_if()` instead.
## To map `funs` over all variables, use `summarise_all()`
```

```
## Warning: Setting row names on a tibble is deprecated.
```

```
## Warning: Ignoring unknown aesthetics: fill
```

6.4 Other useful functions

6.4.1 Querying genes in a certain cluster

The user can obtain the genes in a certain cluster via `geneInCluster` , enabling other analyses, such as drug target identification.

```
# Always make the number as character, please!
geneC <- geneInCluster(clen_res, "pam", "10", "4")
head(geneC)
```

```
## [1] "CD47"      "SERPINB13" "PNP"       "MPZL2"     "KCNJ15"    "SOX7"
```

6.4.2 Gene expression profiling with cluster infromation

It can be obtained by `geneExpInCluster`. There are two items, `clusterGeneExp` and `label`, in the returned object of `geneExpInCluster`. It can be used for other application.

```
# Always make the number as character, please!
gec <- geneExpInCluster(clen_res, "pam", "10")
gec$clusterGeneExp[1:3, 1:4]
```

```
##         cluster_id GSM337261 GSM337262 GSM337263
## PI3              1  6.556556  6.040479  7.033708
## S100A7A          1  4.989918  4.971686  5.677227
## S100A12          1  4.873823  5.168421  5.255036
```

```
gec$label[1:4]
```

```
## GSM337261 GSM337262 GSM337263 GSM337264
##        ct        ct        ct        ct
## Levels: ct Psoriasis
```

6.4.3 The gene correlation in a cluster

The correlation among a cluster can be checked and visulised by `corInCluster`. See Figure 4.

```
# Always make the number as character, please!
corInCluster(clen_res, "pam", "10", "10")
```

7 Bug Report

https://github.com/zhilongjia/cogena/issues

8 Citation

Jia Z. et al. *Cogena, a tool for co-expressed gene-set enrichment analysis and visualization.*

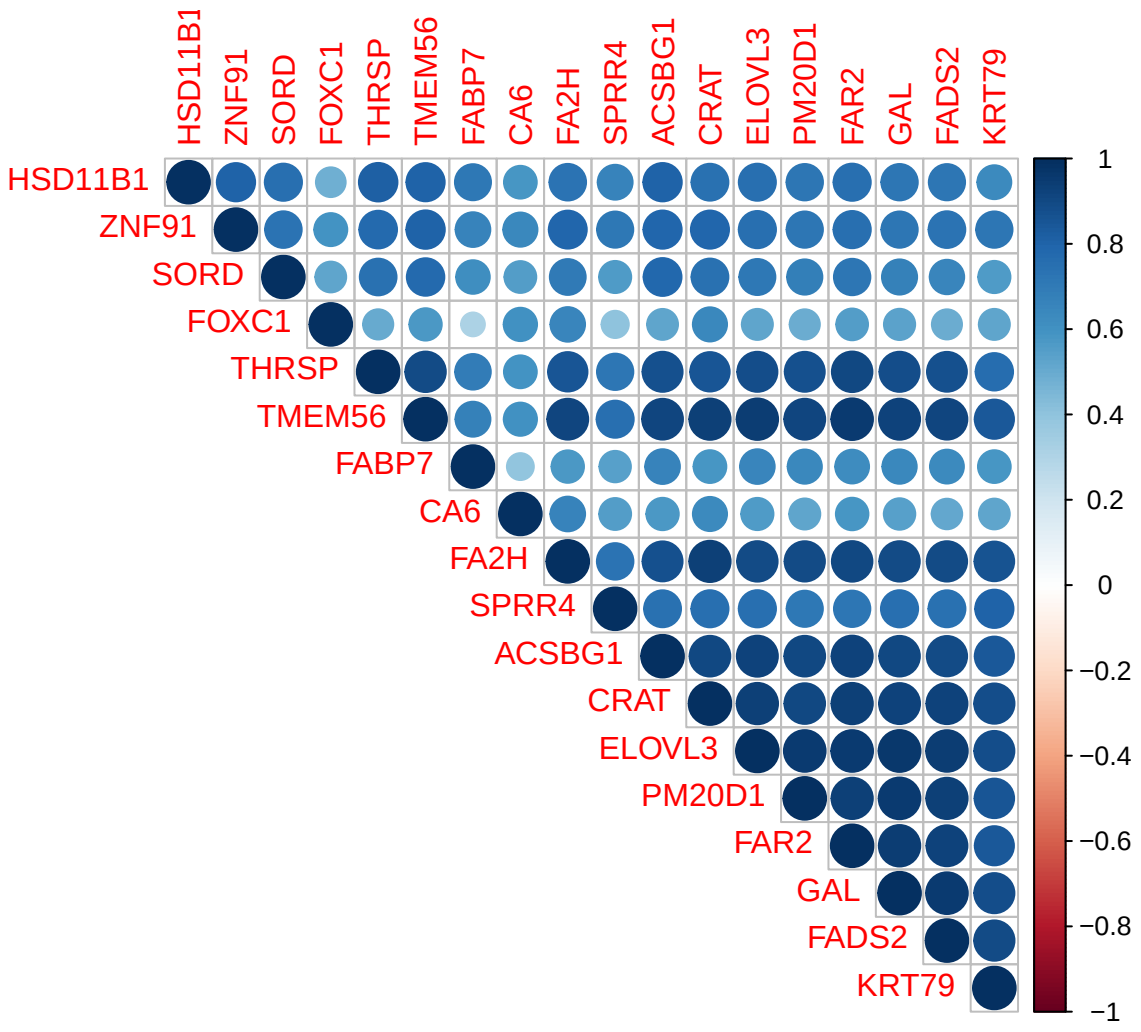

Figure 5: Correlation of genes in a cluster

9 Other Information

System info

```
## R version 3.5.0 (2018-04-23)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04.4 LTS
##
## Matrix products: default
## BLAS: /home/biocbuild/bbs-3.7-bioc/R/lib/libRblas.so
## LAPACK: /home/biocbuild/bbs-3.7-bioc/R/lib/libRlapack.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] bindrcpp_0.2.2  cogena_1.14.0   kohonen_3.0.4   ggplot2_2.2.1
## [5] cluster_2.0.7-1
##
## loaded via a namespace (and not attached):
##  [1] mclust_5.4          Rcpp_0.12.16        mvtnorm_1.0-7
##  [4] lattice_0.20-35     class_7.3-14        gtools_3.5.0
##  [7] assertthat_0.2.0    rprojroot_1.3-2     digest_0.6.15
## [10] foreach_1.4.4       biwt_1.0            R6_2.2.2
## [13] plyr_1.8.4          backports_1.1.2     stats4_3.5.0
## [16] pcaPP_1.9-73        evaluate_0.10.1     highr_0.6
## [19] pillar_1.2.2        gplots_3.0.1        rlang_0.2.0
## [22] lazyeval_0.2.1      gdata_2.18.0        Matrix_1.2-14
## [25] rmarkdown_1.9       apcluster_1.4.5     devtools_1.13.5
## [28] stringr_1.3.0       munsell_0.4.3       compiler_3.5.0
## [31] pkgconfig_2.0.1     BiocGenerics_0.26.0 htmltools_0.3.6
## [34] tibble_1.4.2        codetools_0.2-15    rrcov_1.4-3
## [37] dplyr_0.7.4         withr_2.1.2         MASS_7.3-50
## [40] bitops_1.0-6        grid_3.5.0          gtable_0.2.0
## [43] magrittr_1.5        scales_0.5.0        KernSmooth_2.23-15
## [46] amap_0.8-14         stringi_1.1.7       reshape2_1.4.3
## [49] doParallel_1.0.11   robustbase_0.93-0   fastcluster_1.1.24
## [52] iterators_1.0.9     tools_3.5.0         Biobase_2.40.0
## [55] glue_1.2.0          DEoptimR_1.0-8      parallel_3.5.0
## [58] yaml_2.1.18         colorspace_1.3-2    caTools_1.17.1
## [61] corrplot_0.84       memoise_1.1.0       knitr_1.20
## [64] bindr_0.1.1
```