HTSanalyzeR2: An R package for gene set enrichment and network analysis of various high-throughput data

***Lina Zhu*[1], *Feng Gao*[1], *Xiupei Mei*[2], *and Xin Wang*[1]**

[1]Department of Biomedical Sciences, City University of Hong Kong, Hong Kong
[2]Department of Computer Science, City University of Hong Kong, Hong Kong

2017-12-15

Abstract

This package provides gene set over-representation, enrichment and network analyses for various preprocessed high-throughput data as well as corresponding time-series data including CRISPR, RNA-seq, micro-array and RNAi. It could also generate a dynamic shiny report encompassing all the results and visualizations, facilitating the users maximally for downloading, modifying the visualization parts with personal preference and sharing with others by publishing the report to Shinyapps.io.

Package

HTSanalyzeR2 0.99.10

Contents

1 An overview of HTSanalyzeR2

While high-throughput experiments are no longer bottlenecks for biologists to dissect the functional mechanisms genome-widely, how to efficiently interpretate and visualize the results remains a challenge. There is also no software so far claimed to be able to perform functional annotation for time series data with interactive visualization. Here, we have implemented a versatile R package, **HTSanalyzeR2**, which has several advantages as below:

- **HTSanalyzeR2** can perform gene set over-representation, enrichment and network analyses for pre-processed data generated by various popular high-throughput experiments including RNA-seq, CRISPR, micro-array and RNAi.
- For time series data or a similar experiment coming from different groups, **HTSanalyzeR2** can perform either longitudinal or horizontal 'time-series' analysis for mutual comparing.
- **HTSanalyzeR2** could generate an interactive report for users downloading, modifying the visualizations as well as sharing with others.

Before starting the demonstration, you need to load the following packages:

```
library(HTSanalyzeR2)
library(org.Hs.eg.db)
library(KEGGREST)
library(GO.db)
library(igraph)
```

2 Case study1: Single dataset analysis for gene expression data

This case study is using **HTSanalyzeR2** to perform gene set over-representation, enrichment and network analyses on single gene expression data. Basically, this dataset is from a micro-array experiment on 90 colon cancer patients with GEO number named GSE33113. Using the Colon Cancer Consensus Molecular Subtyping classifier generated by Guinney J et al. in 2015(Guinney J (2015)), we can easily get the subtype label of each patient. Motivated by the poorest prognosis of CMS4 patients, we want to detect the enriched pathways of CMS4 patients compared to non-CMS4 patients. To this end, first we need to do the differential expression analysis using the most popular R packge 'limma' tailored for micro-array data. To make this vignette simple enough, we skip to exlpain this step and start from the result gotten by 'limma'.

2.1 Hypergeometric test and Gene set enrichment analysis

2.1.1 Prepare the input data

To perform gene set enrichment analysis for single dataset, you must preprare the following inputs:

1. a named numeric vector of phenotypes(normally this would be a vector of genes with log2 fold change).

2. a list of gene set collections.

First you need to prepare a named phenotype.

```
data(GSE33113_limma)
phenotype <- as.vector(GSE33113_limma$logFC)
names(phenotype) <- rownames(GSE33113_limma)
```

Then, if you also want to do hypergeometric test on a list of interested genes, you need to define the hits as your interested genes. For example, here we define the hits as genes with absolute log2 fold change greater than 1. In this case, the names of phenotpe, namely all the input genes, would be taken as the background gene list to perform hypergeometric test.

Note:In cases if you want to do hypergeometric test with only a list of hits and no phenotype, **HTSanalyzeR2** can also realize it. For details please go to Part 5:Special using of HTSanalyzeR2.

```
## define hits if you want to do hypergeometric test
hits <-  names(phenotype[which(abs(phenotype) > 1)])
```

Then we must define the gene set collections. A gene set collection is a list of gene sets, each of which consists of a group of genes with a similar known function. **HTSanalyzeR2** provides facilities which greatly simplify the creation of up-to-date gene set collections including three Gene Ontology terms: Molecular Function(MF), Biological Process(BP), Cellular Component(CC), KEGG pathways. Gene sets in a comprehensive molecular signatures database, MSigDB(Arthur Liberzon (2011)), for Homo Sapiens is also provided, containing the most commonly used two collections:'c2'(curated gene sets) and 'c5'(GO gene sets). Here to simplify the demonstration, we will only use one GO, KEGG and one MSigDB gene set collection. To work properly, you need to choose the right species for your input genes. Besides, these gene set collections must be provided as a named list as below:

```
GO_MF <- GOGeneSets(species="Hs", ontologies=c("MF"))
PW_KEGG <- KeggGeneSets(species="Hs")
MSig_C2 <- MSigDBGeneSets(collection = "c2")
ListGSC <- list(GO_MF=GO_MF, PW_KEGG=PW_KEGG, MSig_C2=MSig_C2)
```

2.1.2 Initialize and preprocess

An S4 class named 'GSCA' is developed to perform hypergeometric test in order to find the gene sets sharing significant overlapping with hits. Gene set enrichment analysis, as described by Subramanian et al.(Subramanian A (2005)), can also be conducted.

To initialize a new 'GSCA' object, the previous prepared phenotype and a named list of gene sets collections are needed. In addition, as said before, if you also want to do hypergeometric test, hits is needed.

```
gsca <- GSCA(listOfGeneSetCollections=ListGSC,
             geneList=phenotype, hits=hits)
```

Then a preprocess step including invalid input data removing, duplication removing by different methods, initial gene identifiers converting to Entrez ID and phenotype ordering needs to be performed to fit for the next analysis. See the help documentation of funciton *preprocess* for details.

```
gsca1 <- preprocess(gsca, species="Hs", initialIDs="SYMBOL",
                    keepMultipleMappings=TRUE, duplicateRemoverMethod="max",
                    orderAbsValue=FALSE)
```

2.1.3 Perform analysis

After getting a preprocessed 'GSCA' object, you can perform hypergeometric test and gene set enrichment analysis using the function named *analyze*. This funciton needs an argument called *para*, which is a list of parameters including:

- *pValueCutoff* : a single numeric value specifying the cutoff for adjusted pvalues considered significant.
- *pAdjustMethod*: a single character value specifying the pvalue adjustment method.
- *nPermutations*: a single numeric value specifying the number of permutations for deriving p-values of GSEA.
- *minGeneSetSize*: a single numeric value specifying the minimum number of genes shared by a gene set and the background genes, namely the phenotype. Gene sets with fewer than this number are removed from both hypergeometric test and GSEA.
- *exponent*: a single integer or numeric value used in weighting phenotypes in GSEA, as described by Subramanian et al(Subramanian A (2005)).

```
gsca2 <- analyze(gsca1,
                 para=list(pValueCutoff=0.05, pAdjustMethod="BH",
                           nPermutations=100, minGeneSetSize=180,
                           exponent=1),
                 doGSOA = TRUE, doGSEA = TRUE)
```

In this case study, we only use 100 permutations and set a very large *minGeneSetSize* just for a fast compilation of this vignette. In real applications, you may want a much smaller threshold (e.g. 10) and more permutations(e.g. 1000) to get a more robust GSEA result.

During the enrichment analysis of gene sets, the function evaluates the statistical significance of the gene set scores by performing a large number of permutations. To perform it more efficiently, our package allows parallel calculation based on the *doParallel* package. To do this, the user simply needs to register and claim to use multiple cores **before** running *analyze*.

```
## analyze using multiple cores
if (requireNamespace("doParallel", quietly=TRUE)) {
    doParallel::registerDoParallel(cores=2)
} else {
}

gsca2 <- analyze(gsca1,
                 para=list(pValueCutoff=0.05, pAdjustMethod="BH",
                           nPermutations=100, minGeneSetSize=180,
                           exponent=1),
                 doGSOA = TRUE, doGSEA = TRUE)
```

After analyzing, all the results would be stored in slot *result*. If hypergeometric test and GSEA are both performed, gene sets which are both significant in this two analysis based on either pvalue or adjusted pvalue can be gotten.

```
head(getResult(gsca2)$HyperGeo.results$GO_MF, 3)
##            Universe Size Gene Set Size Total Hits Expected Hits
## GO:0005509         18978           617        477     15.507904
## GO:0042803         18978           720        477     18.096744
## GO:0003714         18978           184        477      4.624723
##            Observed Hits       Pvalue Adjusted.Pvalue
## GO:0005509            43 1.685852e-09    1.260615e-08
## GO:0042803            31 2.641243e-03    8.763329e-03
## GO:0003714            10 1.820557e-02    4.825921e-02
head(getResult(gsca2)$GSEA.results$PW_KEGG, 3)
##          Observed.score Pvalue Adjusted.Pvalue
## hsa05016     -0.5195224      0               0
## hsa04510      0.6267835      0               0
## hsa05205      0.5475122      0               0
head(getResult(gsca2)$Sig.pvals.in.both$MSig_C2, 3)
##                                                    HyperGeo.Pvalue
## TURASHVILI_BREAST_DUCTAL_CARCINOMA_VS_DUCTAL_NORMAL_DN    1.984818e-17
## TONKS_TARGETS_OF_RUNX1_RUNX1T1_FUSION_HSC_UP              1.471194e-07
## MOHANKUMAR_TLX1_TARGETS_DN                                1.316622e-05
##                                                    GSEA.Pvalue
## TURASHVILI_BREAST_DUCTAL_CARCINOMA_VS_DUCTAL_NORMAL_DN           0
## TONKS_TARGETS_OF_RUNX1_RUNX1T1_FUSION_HSC_UP                     0
## MOHANKUMAR_TLX1_TARGETS_DN                                       0
head(getResult(gsca2)$Sig.adj.pvals.in.both$MSig_C2, 3)
##                                                    HyperGeo.Adj.Pvalue
## TURASHVILI_BREAST_DUCTAL_CARCINOMA_VS_DUCTAL_NORMAL_DN       2.924687e-16
## TONKS_TARGETS_OF_RUNX1_RUNX1T1_FUSION_HSC_UP                 9.099610e-07
## MOHANKUMAR_TLX1_TARGETS_DN                                   6.871123e-05
##                                                    GSEA.Adj.Pvalue
## TURASHVILI_BREAST_DUCTAL_CARCINOMA_VS_DUCTAL_NORMAL_DN               0
## TONKS_TARGETS_OF_RUNX1_RUNX1T1_FUSION_HSC_UP                         0
## MOHANKUMAR_TLX1_TARGETS_DN                                           0
```

In addition, to make the results more understandable, you are highly recommended to annotate the gene sets ID to terms by function *appendGSTerms*.

```
gsca3 <- appendGSTerms(gsca2, goGSCs=c("GO_MF"),
                       keggGSCs=c("PW_KEGG"),
                       msigdbGSCs = c("MSig_C2"))
head(getResult(gsca3)$GSEA.results$PW_KEGG, 3)
##                    Gene.Set.Term Observed.score Pvalue Adjusted.Pvalue
## hsa05016    Huntington's disease     -0.5195224      0               0
## hsa04510          Focal adhesion      0.6267835      0               0
## hsa05205 Proteoglycans in cancer      0.5475122      0               0
```

2.1.4 Summarize results

A *summarize* method could be performed to get a general summary for an analyzed 'GSCA' object including the gene set collections, genelist, hits, parameters for analysis and the summary of result.

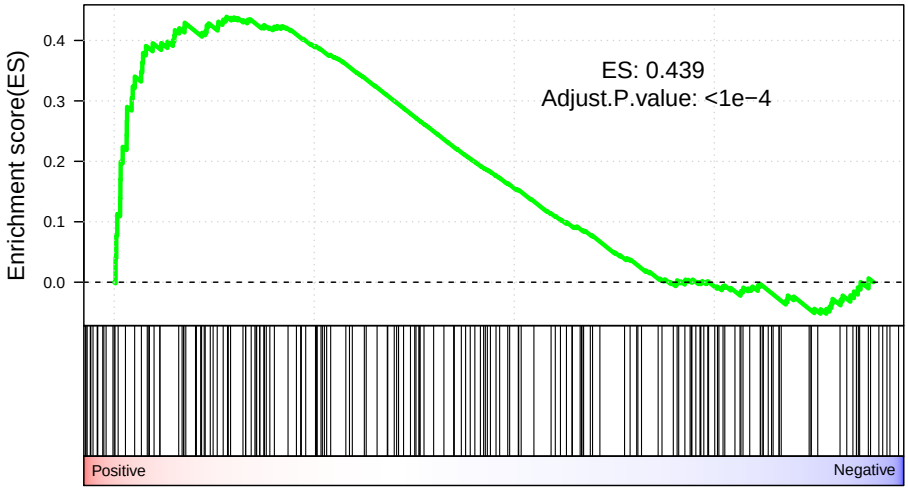
```
HTSanalyzeR2::summarize(gsca3)
##
## -No of genes in Gene set collections:
##         input above min size
## GO_MF    4110             41
## PW_KEGG   323             19
## MSig_C2  3762            441
##
##
## -No of genes in Gene List:
##           input valid duplicate removed converted to entrez
## Gene List 21656 21655             21655               18978
##
##
## -No of hits:
##      input preprocessed
## Hits   496          477
##
##
## -Parameters for analysis:
##               minGeneSetSize pValueCutoff pAdjustMethod
## HyperGeo Test 180            0.05         BH
##
##      minGeneSetSize pValueCutoff pAdjustMethod nPermutations exponent
## GSEA 180            0.05         BH            100           1
##
##
## -Significant gene sets (adjusted p-value< 0.05 ):
##          GO_MF PW_KEGG MSig_C2
## HyperGeo     3       4     185
## GSEA        16      12     339
## Both         3       4     171
```

2.1.5 Plot gene sets

To better view the GSEA result for a single gene set, you can use *viewGSEA* to plot the positions of the genes of the gene set in the ranked phenotypes and the location of the enrichment score. To realize it, you must first get the gene set ID by *getTopGeneSets*, which can return all or the top significant gene sets from GSEA results. Basically, the user needs to specify the type of results–"HyperGeo.results" or "GSEA.results", the name(s) of the gene set collection(s) as well as the type of selection– all (by parameter 'allSig') or top (by parameter 'ntop') significant gene sets.

```
topGS <- getTopGeneSets(gsca3, resultName="GSEA.results",
                        gscs=c("GO_MF", "PW_KEGG"), allSig=TRUE)
topGS
## $GO_MF
##   GO:0003714   GO:0004872   GO:0004871   GO:0005096   GO:0003779
## "GO:0003714" "GO:0004872" "GO:0004871" "GO:0005096" "GO:0003779"
##   GO:0045296   GO:0001077   GO:0004930   GO:0005102   GO:0003682
```

```
## "GO:0045296" "GO:0001077" "GO:0004930" "GO:0005102" "GO:0003682"
##   GO:0005509   GO:0042803   GO:0003723   GO:0003677   GO:0005524
## "GO:0005509" "GO:0042803" "GO:0003723" "GO:0003677" "GO:0005524"
##   GO:0005515
## "GO:0005515"
##
## $PW_KEGG
##   hsa05016   hsa04510   hsa05205   hsa04015   hsa04810   hsa04014   hsa04060
## "hsa05016" "hsa04510" "hsa05205" "hsa04015" "hsa04810" "hsa04014" "hsa04060"
##   hsa04010   hsa05165   hsa04151   hsa05200   hsa01100
## "hsa04010" "hsa05165" "hsa04151" "hsa05200" "hsa01100"
viewGSEA(gsca3, gscName="GO_MF", gsName=topGS[["GO_MF"]][1])
```



You can also plot all or the top significant gene sets in one time and store them as png or pdf format into a specificed path by using *plotGSEA*.

```
plotGSEA(gsca3, gscs=c("GO_MF", "PW_KEGG"), ntop=3, filepath=".")
```

2.1.6 Enrichment Map

To get a comprehensive view of the hypergeometric test result or GSEA result instead of a list of significant gene sets with no relations, our package provides *viewEnrichMap* function to draw an enrichment map for better interpretation(Merico D (2010)). More specifically, in the enrichment map, nodes represent significant gene sets sized by the genes it contains and the edge represents the Jaccard similarity coefficient between two gene sets. Nodes color are scaled according to the adjusted pvalues(the darker, the more significant). For hypergeometric test, there is only one color for nodes whereas for GSEA enrichment map, the default color is setted by the sign of enrichment scores(red:+, blue:-). You can also set your favourite format by changing the parameter named 'options'.

However, you are always highly recommended to use *report* to visualize and modify the enrichment map with personal preference, such as different layout, color and size of nodes, types of labels and etc. More details please go to Part4: Shiny report of results.

```
## the enrichment map for top 5 significant gene sets in 'PW_KEGG' and 'GO_MF'
viewEnrichMap(gsca3, gscs=c("PW_KEGG", "GO_MF"),
              allSig = FALSE, gsNameType = "term", ntop = 5)
tmp <- getTopGeneSets(gsca3, resultName = "GSEA.results", gscs=c("GO_MF", "PW_KEGG"),
                      allSig = T)
GO_MF_geneset <- tmp$GO_MF[c(1,3,5,9,12)]
PW_KEGG_geneset <- tmp$PW_KEGG[c(7,2,5,1,9)]
specificGeneset <- list("GO_MF"=GO_MF_geneset, "PW_KEGG"=PW_KEGG_geneset)
report(gsca3, specificGeneset=specificGeneset)
```

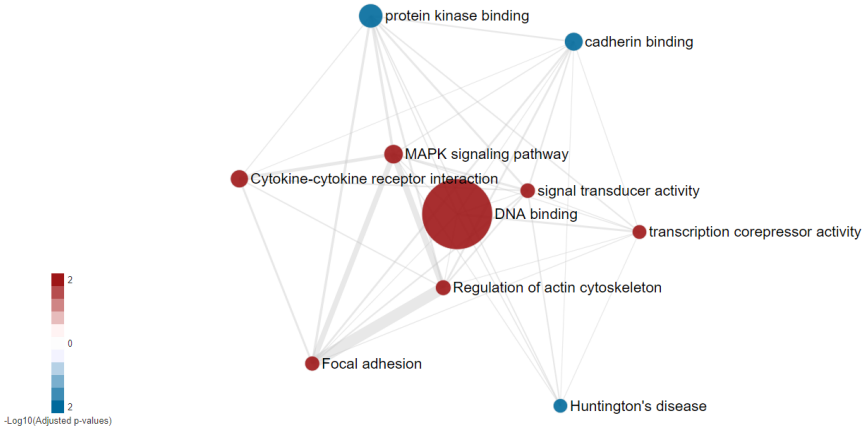


Figure 1:

Arthur Liberzon, Reid Pinchback, Aravind Subramanian. 2011. "Molecular Signatures Database (Msigdb) 3.0." *Bioinformatics* 27 (12): 1739–40. doi:10.1093/bioinformatics/btr260.

Guinney J, Wang X, Dienstmann R. 2015. "The Consensus Molecular Subtypes of Colorectal Cancer." *Nature Medicine* 21 (11): 1350–6. doi:10.1038/nm.3967.

Merico D, Stueker O, Isserlin R. 2010. "Enrichment Map: A Network-Based Method for Gene-Set Enrichment Visualization and Interpretation." *PLoS ONE* 5 (11): e13984. doi:10.1371/journal.pone.0013984.

Subramanian A, Mootha VK, Tamayo P. 2005. "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): 15545–50. doi:10.1073/pnas.0506580102.