

Ontologies - statistics, biases, tools, networks, and interpretation

Day 3 – Enrichment with R (ClusterProfiler, ReactomePA, Dose, PathFindR)

CBNA Course Series 2020

EMBL CBNA Course, 28.10.2020

Dr Matt Rogon, EMBL Heidelberg

DAY 2 – R WORKSHOP	3
INTRODUCTION TO CLUSTERPROFILER, AND REACTOMEPA	4
PICKING BACKGROUND FROM QUANTITATIVE MEASUREMENTS E.G. RNASEQ OR MASS SPECTROMETRY	7
TUTORIAL	10
BREAKOUT ROOM 1.....	11
<i>Exercise 1.....</i>	<i>11</i>
<i>Exercise 2.....</i>	<i>12</i>
<i>Exercise 3.....</i>	<i>13</i>
<i>Exercise 4 Comparing clusters and using custom annotation</i>	<i>14</i>
PATHFINDER	15
<i>Breakout Room 2.....</i>	<i>17</i>
Part 1:	17
Part 2:	17
Part 3:	17

Day 2 – R workshop

Up until now we've been using tools with graphical user interfaces. Either webpages or standalone tools such as Cytoscape.

In this tutorial you will learn how to use R packages for data processing against ontologies and pathways:

- ClusterProfiler
- Dose
- ReactomePA
- PathFindR

All preprocessed files are provided, you will need to modify data frames to correspond to package requirements as we analyze data in the exercises.

Introduction to ClusterProfiler, and ReactomePA

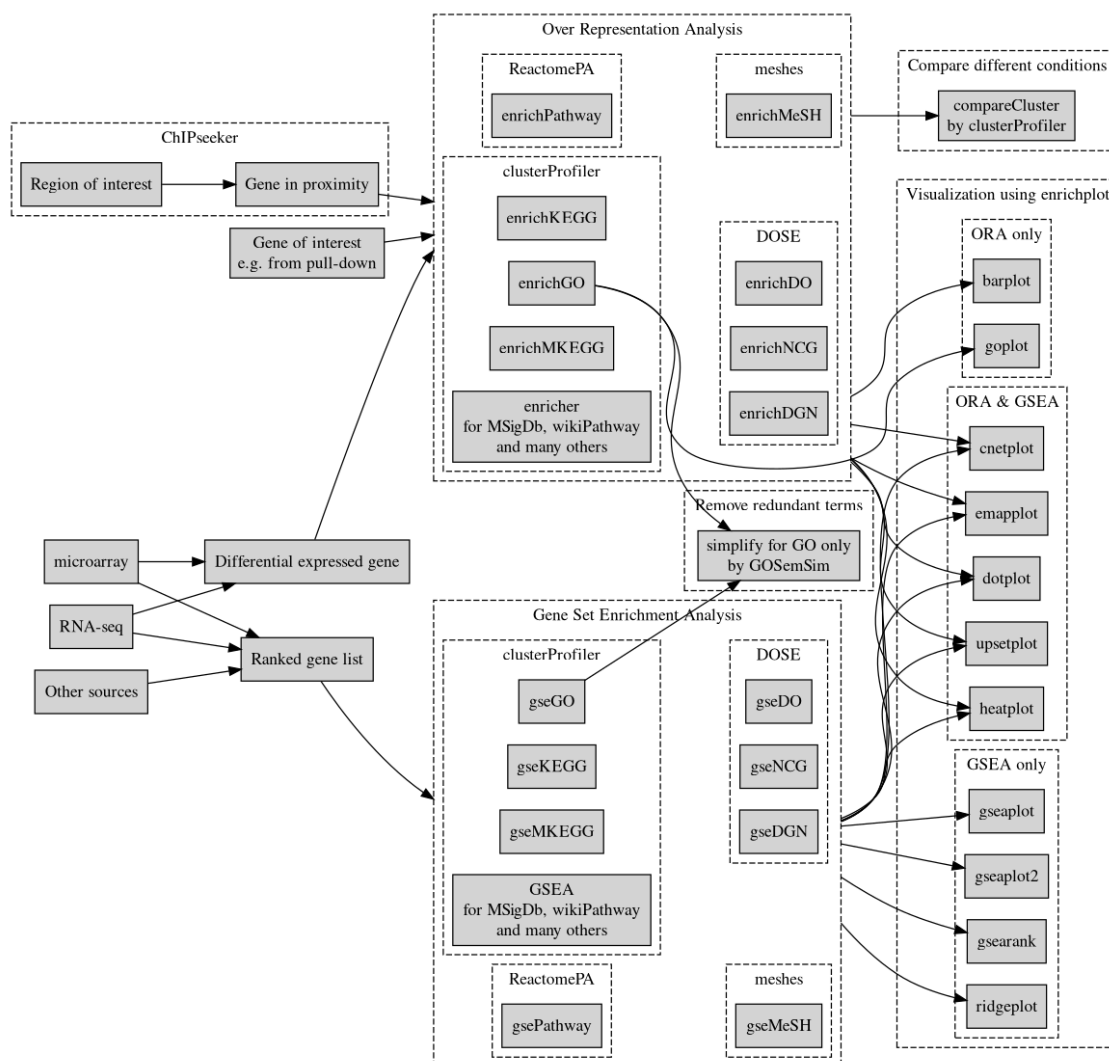
Launch RStudio and load the provided R environment file: **Complete_2020.RData**

(**folder** /Students materials/Part 2 - R-code session ClusterProfiler, ReactomePA, pathfindR/1. Load data and script/)

Open the script for this session: **R_code_reduced_for_RData.R**

- By now you should have all of the packages installed on your system as requested in the introductory email.
- Set your working directory to the location of the source files.

clusterProfiler implements methods to analyze and visualize functional profiles of genomic coordinates (supported by **ChIPseeker**), gene and gene clusters.



Supported Analysis

- Over-Representation Analysis
- Gene Set Enrichment Analysis
- Biological theme comparison

Supported ontologies/pathways

- Disease Ontology (via *DOSE*)
- *Network of Cancer Gene* (via *DOSE*)
- *DisGeNET* (via *DOSE*)
- Gene Ontology (supports many species with GO annotation query online via *AnnotationHub*)
- KEGG Pathway and Module with latest online data (supports more than 4000 species listed in http://www.genome.jp/kegg/catalog/org_list.html)
- Reactome Pathway (via *ReactomePA*)
- DAVID (via *RDAVIDWebService*)
- *Molecular Signatures Database*
 - hallmark gene sets
 - positional gene sets
 - curated gene sets
 - motif gene sets
 - computational gene sets
 - GO gene sets
 - oncogenic signatures
 - immunologic signatures
- Other Annotations
 - from other sources (e.g. *DisGeNET* as *an example*)
 - custom annotation

Visualization

- barplot, cnetplot, dotplot, emapplot, gseaplot, goplot, upsetplot

Please go to <https://yulab-smu.github.io/clusterProfiler-book/> for the full vignette.

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. *OMICS: A Journal of Integrative Biology* 2012, 16(5):284-287.

doi:[10.1089/omi.2011.0118](http://dx.doi.org/10.1089/omi.2011.0118)

Picking background from quantitative measurements e.g. RNASeq or Mass Spectrometry

<https://genomebiology.biomedcentral.com/articles/10.1186/s13059-015-0761-7>

- Background should include '**any gene that COULD HAVE been positive**'.
- **At least be limited to all the genes interrogated by the technology platform.**
- One should remove genes from the background gene set if they are **not expressed in any sample of the experiment**. Thus, in **RNASeq we remove zero-count** genes from the genome, and in **proteomics all non-identified proteins** from the known proteome “universe”.

Several studies identified significant sampling bias in functional enrichment analysis [Hansen et al 2011, Lindholm et al 2014]. Specifically, the ‘discovery’ of significant functional enrichment profiles could be achieved in almost every analysis, regardless of how the regulated gene list was selected.

Sampling bias can arise in 3 ways. Technology, Detection, Biology.

1. **Technology.** Every RNA detection technology, including RNA sequencing, has a biased representation of the gene ontology structure. For example, the Affymetrix Human Genome U133 Plus 2.0 GeneChip has proportionately more genes linked to ‘acetylation’ ($P < 7 \times 10^{-51}$) than does the genome (as defined by the DAVID online gene annotation tool [7]), whereas the Agilent 44 K chip has proportionately more genes linked to ‘mutagenesis site’ ($P < 2 \times 10^{-46}$). In fact, hundreds of gene categories are massively ‘enriched’ in functional classifications on every microarray—this is called technology bias.
2. **Detection.** Not all genes can be detected with equal reliability, to the extent that some genes are never detected as being ‘regulated’ (the signal never changes). This is detection bias, which can reflect aspects of the transcriptomics technology or the sequence of the transcript that is being probed.
3. **Biology.** The transcriptome of a given cell type or tissue is highly specialized, to the point that it can be used to determine the identity of an unknown RNA profile efficiently; this is referred to as biological bias.

Solution?

Not available on the technology level.

- a. There is no technology that can detect all transcripts with equal probability (regardless of differences in abundance).
- b. RNA-seq data are neither unbiased nor global, and because the final RNA-seq data are a statistically derived estimate of expression, any bias that impacts on the likelihood of detection will impact on the functional enrichment analysis

A few proposed solutions:

- *The generation of an estimated background 'universe' in RNA-seq data could be achieved by removing zero-count genes.*
- *Similarly, the application of functional enrichment analysis to 'global' proteomics, where a large part of the 'molecular universe' may not be detected, is fraught with problems. In proteomics we usually take as background the proteins which have any identifications.*
- *Finally, explicitly modeling background – as Huber et al propose the selection of background genes that are similar to differentially expressed genes.*

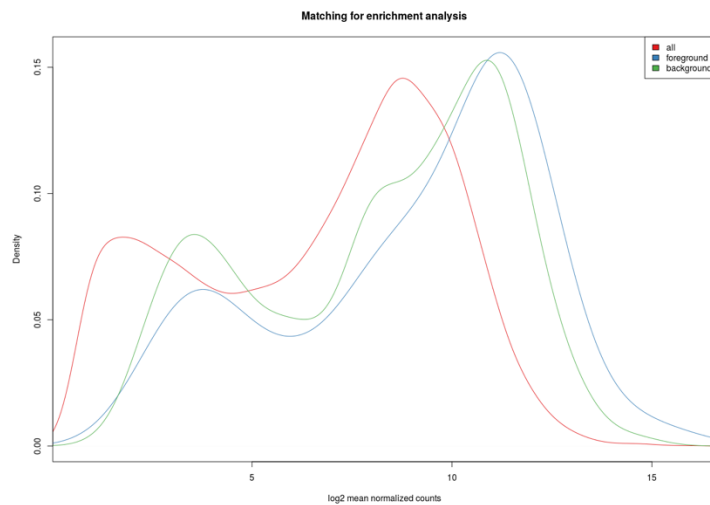
Explicit background modeling approach is shown in:

<https://www.huber.embl.de/users/klaus/Teaching/DESeq2Predoc2014.html#gene-ontology-enrichment-analysis>

Under the assumption that a background is formed by non-DE genes thus genes which show expression similar to differentially expressed significant hits, we can use the *genefinder* function from the R package “*genefilter*”

- Here we find background genes that are similar in expression to the differentially expressed genes. The function tries to identify 10 genes for each DE-gene that match its expression strength.
- We then check whether the background has roughly the same distribution of average expression strength as the foreground by plotting the densities.
- We do this in order not to select a biased background since the gene set testing is performed by a simple Fisher test on a 2x2 table, which uses only the status of a gene, i.e. whether it is differentially expressed or not, and not its fold-change or absolute expression.
- Note that the chance of a gene being identified as DE will most probably depend on its expression for RNA-Seq data (potentially also its gene length etc.). Thus it is important to find a matching background.

- The testing approach here is very similar to web tools like *DAVID*, however, we explicitly model the background here.



- We first get average gene expressions for each of the genes and then find non-DE genes that show a similar expression as the DE-genes. These genes are then our background.
- We now remove DE genes from background and then get the total number of genes in the background.
- Plotting the density of the average expressions, shows that the background matching has worked reasonably well.
- We can now perform the actual testing.
- For this purpose we can use various packages including ClusterProfiler, ReactomePA, PathfindR, TopGO and many more.

Tutorial

- Go to the R file `R_code_reduced_for_RData_2020.R`
- You will analyze the code and familiarize yourselves with the execution and formatting of commands for `ClusterProfiler`, `Dose`, and `ReactomePA`

Breakout Room 1

This breakout room is divided into 5 exercises where you will be exploring in more detail the functionality of the packages.

Exercise 1

You have 10 minutes

- Using the data frame: results
- Perform an enrichment against KEGG Module

How many modules did you find?

Name them:

Exercise 2

You have 20 minutes

Given a list of hits and a background in folder /Exercise 2

background.txt and

id_list.txt

Perform enrichment analysis against:

1. Gene Ontology,
2. Kegg modules, and
3. DisGeNet (diseases)

For each result produce a cnetplot and an upsetplot.

Note - you will need to provide the correct id's for the functions to execute - use converters such as BioMart (example is in the code file), or the function bitr with AnnotationHub

- vary the p-value q-value thresholds in case of an empty result

- make the DisGeNet output readable (gene symbols, not entrez id's)

Exercise 3 Comparing clusters

You have 20 minutes

Here you will be comparing multiple clusters using ClusterProfiler.

For this exercise we will use the 550prot_clusters.txt file from the Exercise 3 directory

The function which does the cluster comparison can be used for many applications, doesn't need to be clusters – could be conditions/treatments, disease vs healthy etc.

Procedure to do this?

- Load the data found in the Exercise folder: 550prot_clusters.txt
- For each cluster you will need a list of Entrez identifiers
 - o Convert the input from UniProt to Entrez use any tool you like – there is a sample with bitr function (code line ~320 in the main R tutorial script).
 - o Create a list of lists which will contain all clusters (one per list)
- Run enrichment against GO, Reactome, KEGG
 - o You will need a function which accepts list of lists and runs comparisons
 - o Create dotplots for the results

Exercise 4 Comparing clusters and building a custom annotation

You have 20 minutes

- set your working directory to the subdirectory /Exercise 4
- Here you will find 3 files:
 - o cluster1.txt and cluster2.txt, which contain the lists of genes of interest, here you will need to convert id's, create a list of lists to pass to compareClusters
 - o all_gene_disease_associations.tsv – this file contains gene to disease annotations.
 - Column 1 contains entrez id's, subsequent columns annotations to diseases
 - Use this file to build a custom annotation files with the following format:
 - Dataframe 1: disease2gene: `c("diseaseId", "geneId")`
 - Dataframe 2: disease2name `c("diseaseId", "diseaseName")`

TODO:

1. perform comparative cluster analysis of the two clusters against the custom annotation file found in all_gene_disease_associations.tsv
 - a. explore the function compareCluster and enricher to complete the task
 - b. compareCluster accepts 'fun=' switch, which works the same as passing fun='enrichGO' or fun='enrichKEGG' i.e. you can pass fun='enricher'
2. produce dotplot comparing the two clusters

Here is an example on how to proceed with bitr function for identifier conversion:

```
keytypes(org.Hs.eg.db)
```

```
hit_ids <- bitr(hit_list$gene, fromType="SYMBOL", toType=c("ENTREZID"), OrgDb="org.Hs.eg.db")
```

The last exercise is based on PathFindR R package. First some introductions:

Exercise 5 PathFindR Tutorial and 3 exercises

You have until end-of-day

As you've now seen the most commonly used pathway analysis methods are:

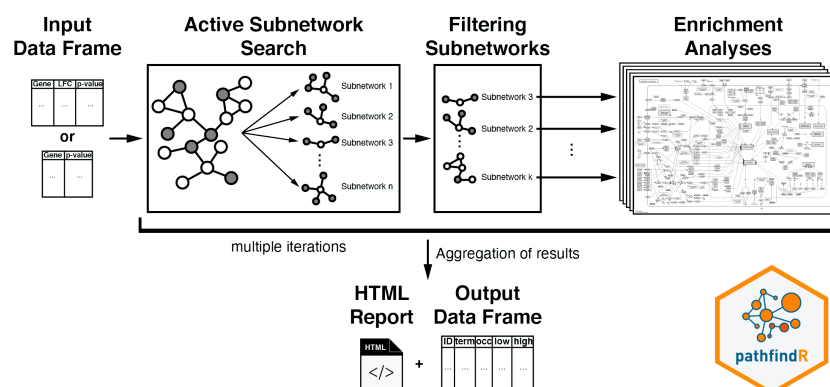
ORA - overrepresentation analyses - for each pathway, ORA statistically evaluates the proportion of altered genes among the pathway genes against the proportion among a set of background genes. Alternatively:

FCS - functional class scoring – here a gene-level statistic is calculated using the measurements from the experiment. These gene-level statistics are then aggregated into a pathway-level statistic for each pathway. Finally, the significance of each pathway-level statistic is assessed, and significant pathways are determined.

Neither method uses interaction data.

pathfindR algorithm executes in multiple stages:

- Identification of active subnetworks
- enrichment analysis using the identified active subnetworks
 - For a list of genes, an active subnetwork is defined as a group of interconnected genes in a protein-protein interaction network (PIN) - a reference network - that predominantly consists of **significantly altered genes**.
 - In other words, active subnetworks define distinct associated sets of interacting genes. These altered subgraphs may be associated with specific pathways, or diseases.



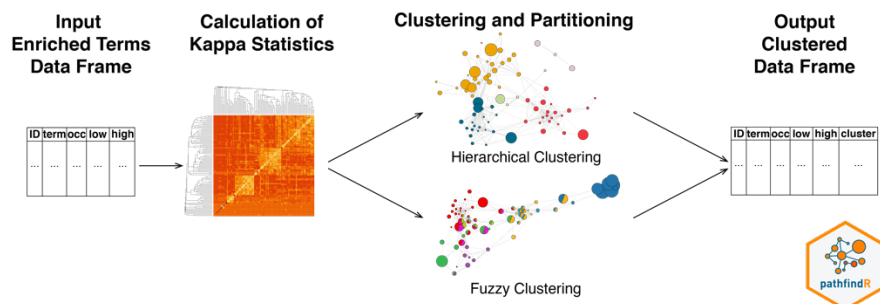
The workflow takes in a data frame consisting of “**gene symbols**”, “**change values**” (optional) and “**associated p values**”:

Gene_symbol	logFC	FDR_p
FAM110A	-0.69	3.4e-06
RNASE2	1.35	1.0e-05

And performs:

- Input testing, Gene symbol mapping to interaction network (if not found – converted to alias and searched again), Mapping log fold changes and p-values onto the selected PIN network
- Active subnetwork search is performed using either:
 - o a greedy algorithm, a simulated annealing algorithm, or a genetic algorithm.
 - o Detailed method description can be found here:
<https://www.frontiersin.org/articles/10.3389/fgene.2019.00858/full>
- Active subnetworks are filtered on their scores and number of significant genes they contain.
- Enrichment analysis - significantly enriched terms (pathways/gene sets) are identified.
- Enriched terms with adjusted p values larger than the given threshold are discarded and the lowest adjusted p value (over all active subnetworks) for each term is kept.
 - o This process of active subnetwork search + enrichment analyses is repeated for a selected number of iterations, performed in parallel. Over all iterations, the lowest and the highest adjusted-p values, as well as number of occurrences over all iterations are reported for each significantly enriched term.
- Finally - Clustering of the Enriched Terms is performed - pairwise kappa statistic between the enriched terms is calculated. Hierarchical clustering follows (by default), and automatically determines the optimal number of clusters by maximizing the average silhouette width and returns a data frame with cluster assignments.

This workflow can be run using the function `run_pathfindR()` and `cluster_enriched_terms()`.



PathfindR can be fully customized with a selection of reference network e.g. IntAct, thresholds of p-values, and algorithms for subnetwork detection.

Breakout Room 2

Part 1:

Load the following file: *pathfinder_input_exercise_noClusters.tsv*

- prepare the data frame for pathfinder - you will need 3 columns:
 - o Gene name
 - o logFC
 - o adj P Val
- Run pathfindR wrapper function on this dataset
- Open results

Part 2:

Create a custom script that will execute pathfindR with the following settings:

- Benjamini-Hochberg FDR
- adjusted p_val_threshold 0.01
- gene sets: GO-BP
- minimum gene set size 5, maximum 100
- PIN protein interaction reference network: IntAct

Generate a fuzzy-clustered result network of terms

Generate visualization for:

- term gene heatmap, term gene graph, and an upset plot

Part 3:

Use the original file *pathfinder_input_exercise_clusters.txt* and compare clusters 1 and 2 with pathfinder

- use *combine_pathfindR_results* function
- produce visualisations