Cw10 Analiza GO

Projekt zaliczeniowy

Wykonanie analiz RNAseq dla bioprojektu PRJNA313294

- Dwie próbki - zika (zakażone) czynnik badany
- Dwie próbki - mock (zakażone) kontrola
- Komórki ludzkie - neutral progenitorl cell (hNPCc)
- Odczyty 75 nt pair end z urządzenia MiSeq - 4 biblioteki 2x2
- Odczyty 75 nt single end z urządzenia NextSeq - 4 biblioteki 2x2
- Za pomocą wybranego piplinu wykonać analizy RNAseq dla 8 bibliotek
- Przygotować skrypt który będzie zawierał następujące elementy
  - Pobieranie danych
  - Kontrola jakości np. FastQC i filtrowanie odczytów
  - Mapowanie odczytów, jako genom/transkryptom referencyjny proszę wykorzystać genom hg19
  - Zliczanie odczytów
  - Analizę DE
  - Porównać wyniki uzyskane dla obu urządzeń
  - W R wykonać PCA oraz heatmapę dla uzyskanych wyników
- Wyniki proszę dostarczyć w formie elektronicznej (preferowany plik pdf) oraz skrypt .sh, .py, .ipynb z odpowiednimi komentarzami
- Termin wysłania projektu 14 luty 2020

Gene ontology - GO terms

To ważna inicjatywa bioinformatyczna mająca na celu ujednolicenie reprezentacji cech genów i produktów genowych we wszystkich gatunkach. Dokładniej, projekt ma na celu:

1. Utrzymanie i rozwój kontrolowanego słownika cech genowych i produktów genowych;
2. Opisywać geny i produkty genów oraz asymilować i rozpowszechniać dane opisowe; oraz
3. Zapewniają narzędzia ułatwiające dostęp do wszystkich aspektów danych dostarczanych przez projekt oraz umożliwiają funkcjonalną interpretację danych eksperymentalnych za pomocą GO, na przykład "analizy wzbogacania" (ang. enrichment analysis).

Źródła informacji o anotacji GO

- bazy danych (np. UniProt)
- narzędzia analizy online (np. Gene Ontology portal)
- DAVID

Narzędzia do analizy GO

- Pakiety Bioconuctor
  - geoseq
  - topGO
  - GOfunR

Ćwiczenia praktyczne

Zadanie 1

Na podstawie informacji w bazie UniProt, proszę utworzyć listę (z nazwami genów Łubinu wąskolistnego), zawierającą informacje o przypisanych numerach GO dla poszczególnych genów. Do utowrzenia listy proszę wykorzystać funkcję `readMappings` z pakietu `topGO`.

Enrichment analysis

```
dane <- read.csv("/Dydaktyka/counts.txt", skip = 1, sep = "\t")
library(goseq)
```

```
## Loading required package: BiasedUrn
```

```
## Loading required package: geneLenDataBase
```

```
##
```

```
library(topGO)
```

```
## Loading required package: BiocGenerics
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter,
##     Find, get, grep, grepl, intersect, is.unsorted, lapply, Map,
##     mapply, match, mget, order, paste, pmax, pmax.int, pmin,
##     pmin.int, Position, rank, rbind, Reduce, rownames, sapply,
##     setdiff, sort, table, tapply, union, unique, unsplit, which,
##     which.max, which.min
```

```
## Loading required package: graph
```

```
## Loading required package: Biobase
```

```
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
## Loading required package: GO.db
```

```
## Loading required package: AnnotationDbi
```

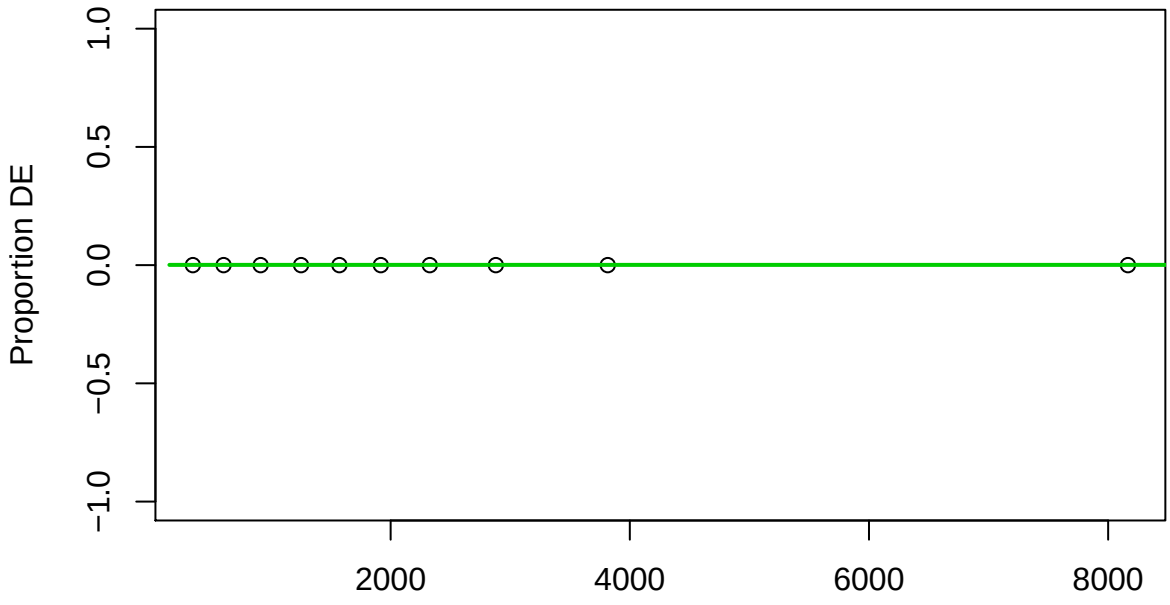
```
## Loading required package: stats4
## Loading required package: IRanges
## Loading required package: S4Vectors
##
## Attaching package: 'S4Vectors'
## The following object is masked from 'package:base':
##
##     expand.grid
## Loading required package: SparseM
##
## Attaching package: 'SparseM'
## The following object is masked from 'package:base':
##
##     backsolve
##
## groupGOTerms:    GOBPTerm, GOMFTerm, GOCCTerm environments built.
##
## Attaching package: 'topGO'
## The following object is masked from 'package:IRanges':
##
##     members
```

```
geneID2GO <- readMappings(file = '/Dydaktyka/geneid2go.map2')
gene.bad.id <- dane$Geneid
popraw.nazwy <- function(gen.id){
  dobry.gen.id <- strsplit(as.character(gen.id), split = ".", fixed = T)[[1]][1]
  return(dobry.gen.id)
}

r <- read.csv('/Dydaktyka/DE_wyniki.csv')

r1 <- as.data.frame(r)
de.gene <- row.names(r1)
de.gene <- sapply(de.gene, popraw.nazwy)

assigned.gene <- sapply(gene.bad.id, popraw.nazwy)
gene.vector <- as.integer(assigned.gene%in%de.gene)
names(gene.vector) <- assigned.gene
print(sum(gene.vector))
```

```
## [1] 0
```

```
length.data <- dane$Length
pwf=nullp(gene.vector,"unnOrg","unnGen", length.data)
```

```
## Warning in newton(lsp = lsp, X = G$X, y = G$y, Eb = G$Eb, UrS = G$UrS, L =
## G$L, : Fitting terminated with step failure - check results carefully
```



Biased Data in 4400 gene bins.

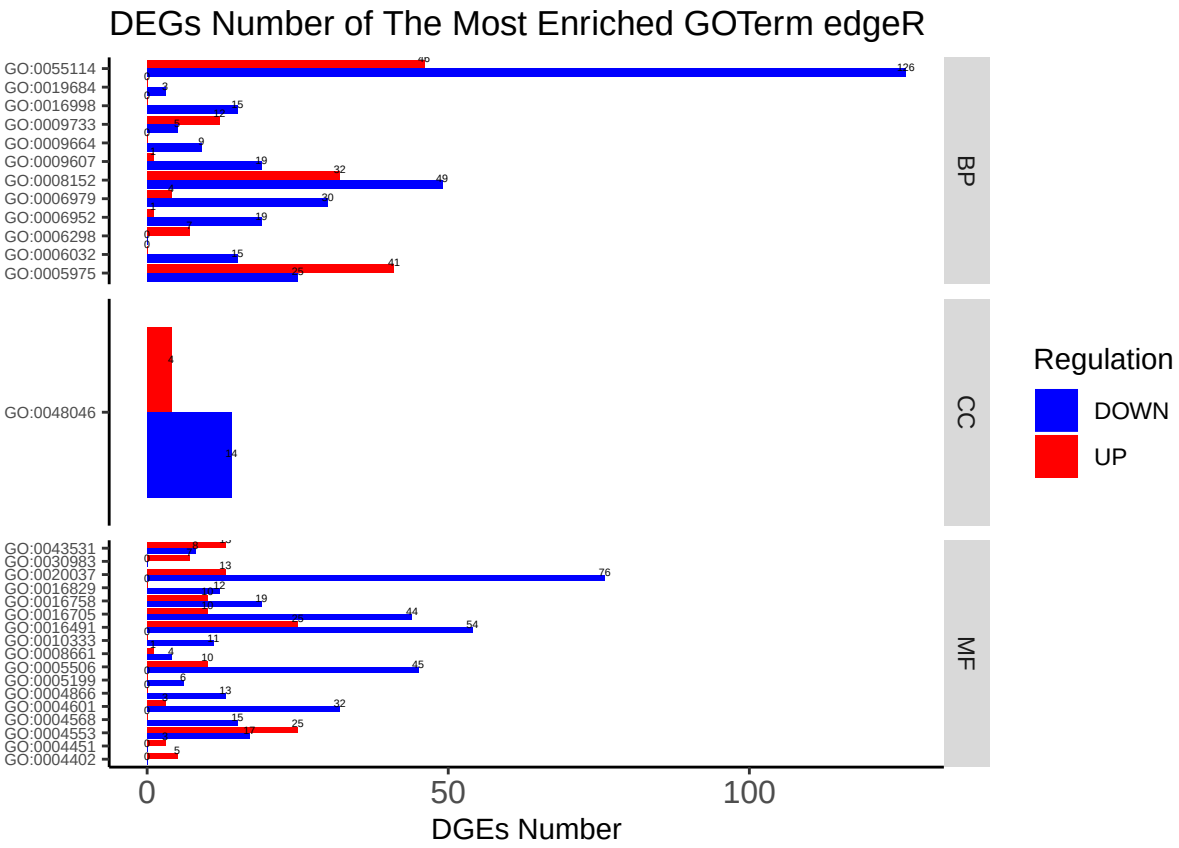
```
GO.wall=goseq(pwf,"unnOrg","unnGen", gene2cat = geneID2GO, test.cats=c("GO:CC", "GO:BP", "GO:MF"), metho
```

```
## Using manually entered categories.
## For 14729 genes, we could not find any categories. These genes will be excluded.
## To force their use, please run with use_genes_without_cat=TRUE (see documentation).
## This was the default behavior for version 1.15.1 and earlier.
## Calculating the p-values...
## 'select()' returned 1:1 mapping between keys and columns
```

```
GO.wall.CC <- GO.wall[GO.wall$ontology=='CC',]
enriched.GO=GO.wall.CC$category[p.adjust(GO.wall.CC$over_represented_pvalue,method="BH")<.05]
GO.wall1.CC <- GO.wall.CC[GO.wall.CC$category%in%enriched.GO,]
GO.wall1.CC$FDR <- p.adjust(GO.wall.CC$over_represented_pvalue,method="BH")[p.adjust(GO.wall.CC$over_rep
GO.wall1.CC <- GO.wall1.CC[!is.na(GO.wall1.CC$FDR),]
GO.wall.MF <- GO.wall[GO.wall$ontology=='MF',]
enriched.GO=GO.wall.MF$category[p.adjust(GO.wall.MF$over_represented_pvalue,method="BH")<.05]
GO.wall1.MF <- GO.wall.MF[GO.wall.MF$category%in%enriched.GO,]
GO.wall1.MF$FDR <- p.adjust(GO.wall.MF$over_represented_pvalue,method="BH")[p.adjust(GO.wall.MF$over_rep
GO.wall1.MF <- GO.wall1.MF[!is.na(GO.wall1.MF$FDR),]
GO.wall.BP <- GO.wall[GO.wall$ontology=='BP',]
enriched.GO=GO.wall.BP$category[p.adjust(GO.wall.BP$over_represented_pvalue,method="BH")<.05]
GO.wall1.BP <- GO.wall.BP[GO.wall.BP$category%in%enriched.GO,]
GO.wall1.BP$FDR <- p.adjust(GO.wall.BP$over_represented_pvalue,method="BH")[p.adjust(GO.wall.BP$over_rep
GO.wall1.BP <- GO.wall1.BP[!is.na(GO.wall1.BP$FDR),]
```

Zadanie 2

Na podstawie wyników analizy GO proszę wykonać poniższy wykres (`ggplot`, `facet_grid`, `geom_bar`)


DEGs Number of The Most Enriched GOTerm edgeR
GO:0055114
GO:0019684
GO:0016998
GO:0009733
GO:0009664
GO:0009607
GO:0008152
GO:0006979
GO:0006952
GO:0006298
GO:0006032
GO:0005975
GO:0048046
GO:0043531
GO:0030983
GO:0020037
GO:0016829
GO:0016758
GO:0016705
GO:0016491
GO:0010333
GO:0008661
GO:0005506
GO:0005199
GO:0004866
GO:0004601
GO:0004568
GO:0004553
GO:0004451
GO:0004402
BP
CC
MF
0
50
100
DGEs Number
Regulation
DOWN
UP