GO_Analysis

2023-05-04

1. Load packages

```
if (!requireNamespace("BiocManager", quietly=TRUE)) {
  install.packages("BiocManager")
}

BiocManager::install("GOfuncR")
BiocManager::install("Homo.sapiens")

library(GOfuncR)
library(tidyverse)
library(ggplot2)
```

2. Data transformation

2.a. Import data sets The `geneNamesUnique.txt` file is provided by Dr. Rebecca L. Young. The `significantMaps.csv` file is manually created from the differentially expressed genes produced from the DGE Analysis

```
GO_input_genes <- read.delim(file = "/stor/work/Bio321G_RY_Spring2023/StudentDirectories/KyNguyen/MiniP
                             header = TRUE)

differentially_expressed_genes_mapping <- read.csv(file = "/stor/work/Bio321G_RY_Spring2023/StudentDire
```

```
colnames(GO_input_genes)[1] <- "gene_id"   # Change column name from "x" to "gene_id"
GO_input_genes <- data.frame(gene_id = unlist(strsplit(as.character(GO_input_genes$gene_id), "_"))) # S


differentially_expressed_genes <- data.frame(gene_name = unlist(strsplit(as.character(differentially_exp
differentially_expressed_genes <- pull(differentially_expressed_genes, gene_name)  # Convert the list o

# Create a new column for whether the gene is the candidate gene or not
```

```
GO_input_genes <- GO_input_genes %>%
  mutate(is_candidate = ifelse(tolower(gene_id) %in% tolower(differentially_expressed_genes), 1, 0))  #
```

2.b. Data Transformation

3. GO Analysis

3.a. GO Enrichment Analysis The hypergeometric test evaluates the over- or under-representation of a set of candidate genes in GO-categories, compared to a set of background genes. The input for the hypergeometric test is a dataframe with two columns

1. A column with gene-symbols
2. A binary column with 1 for a candidate gene and 0 for a background gene.

The output of `go_enrich` is a list of 4 elements. We're only focusing on the first two outputs

1. The results from the enrichment analysis, which is ordered by family-wise error rates (FWER) for over-representation of candidate genes
2. A dataframe with all valid input genes
3. The reference genome for the annotations and the version of the GO-graph
4. A dataframe with the minimum p-values from the permutations, which are used to compute the FWER

```
GO_Enrich_outputs <- go_enrich(GO_input_genes, test = "hyper")

results <- GO_Enrich_outputs$results
genes <- GO_Enrich_outputs$genes
```

```
# Subset over- and under- represented genes with p-value <= 0.05 (i.e., statistically significant genes
overrepresented_genes <- results[results$raw_p_overrep <= 0.05,]
underrepresented_genes<- results[results$raw_p_underrep <= 0.05,]
candidate_genes <- genes[genes$is_candidate == 1,]
genes_annotation <- get_anno_categories(candidate_genes$gene_id)  # Get the gene function (e.g., protei

# Left join the over/underrepresented_genes data.frame with the genes_annotation data.frame
# You can also use the left_join() function from the dplyr library
annotated_overrepresented_genes <- merge(overrepresented_genes, genes_annotation,
                                         by.x = "node_id", by.y = "go_id",
                                         all.x = TRUE, all.y = FALSE)
annotated_underrepresented_genes <- merge(underrepresented_genes, genes_annotation,
                                         by.x = "node_id", by.y = "go_id",
                                         all.x = TRUE, all.y = FALSE)

# Grouped genes with the same function (the "name" column) and remove NA rows
annotated_overrepresented_genes <- annotated_overrepresented_genes %>%
  group_by(node_id) %>%
  mutate(gene = paste(gene, collapse=",")) %>%
  unique %>%
  na.omit
```

```r
annotated_underrepresented_genes <- annotated_underrepresented_genes %>%
  group_by(node_id) %>%
  mutate(gene= paste(gene, collapse=",")) %>%
  unique %>%
  na.omit

# Remove the last two columns
annotated_overrepresented_genes[ ,c(9, 10)] <- NULL
annotated_underrepresented_genes[ ,c(9, 10)] <- NULL

# Convert to table
write.table(annotated_overrepresented_genes, "annotated_overrepresented_genes", quote = F, row.names = 
write.table(annotated_underrepresented_genes, "annotated_underrepresented_genes",quote = F, row.names = 

# Since our annotated_underrepresented_genes table is empty, we only need to work on the overrepresente
annotated_overrepresented_genes <- annotated_overrepresented_genes %>%
  mutate(tally = str_count(gene, ",") + 1)   # Add a tally of the number of genes performing each funct
```

3.b. Data Transformation, again.

4. Visualization - GO Figure

```r
graph_data <- annotated_overrepresented_genes %>%
  dplyr::select(node_name, ontology, gene, FWER_overrep, tally)
graph_data <- graph_data[order(graph_data$FWER_overrep, decreasing = FALSE),]

graph_data$FWER_overrep <- graph_data$FWER_overrep + 0.001   # smallest value is 0 and -log(0) is undef
graph_data <- filter(graph_data, FWER_overrep <= 0.2)        # p-value less than 0.2 (i.e., 80% level o
graph_data$FWER_overrep <- -log(graph_data$FWER_overrep)/log(10)   # log-transform FWER_overrep

head(graph_data, 10)
```

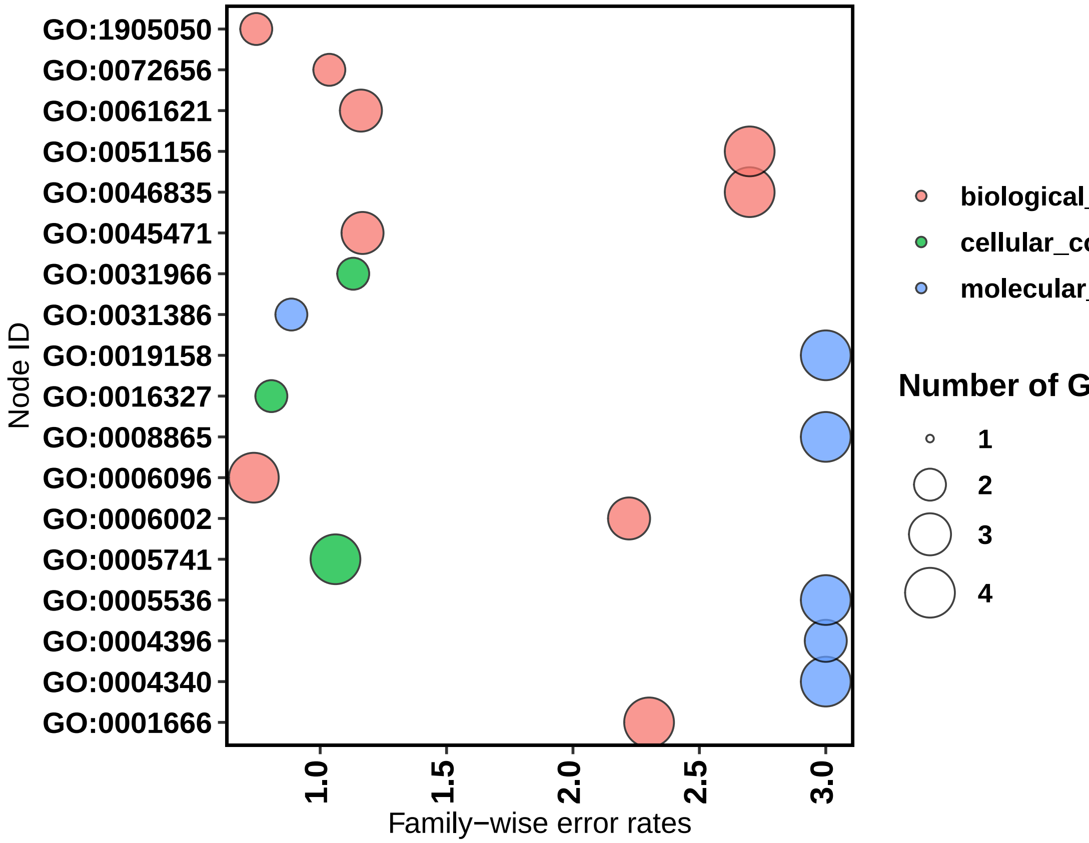
4.a. Prepare graphing data

```
## # A tibble: 10 x 6
## # Groups:   node_id [10]
##    node_id    node_name                        ontology gene  FWER_overrep tally
##    <chr>      <chr>                            <chr>    <chr>        <dbl> <dbl>
##  1 GO:0004340 glucokinase activity             molecul~ HK3,~         3        4
##  2 GO:0004396 hexokinase activity              molecul~ HK1,~         3        3
##  3 GO:0005536 glucose binding                  molecul~ HK2,~         3        4
##  4 GO:0008865 fructokinase activity            molecul~ HK2,~         3        4
##  5 GO:0019158 mannokinase activity             molecul~ HK1,~         3        4
##  6 GO:0046835 carbohydrate phosphorylation     biologi~ HK1,~         2.70     4
##  7 GO:0051156 glucose 6-phosphate metabolic p~ biologi~ HK1,~         2.70     4
##  8 GO:0001666 response to hypoxia              biologi~ HK2,~         2.30     4
##  9 GO:0006002 fructose 6-phosphate metabolic ~ biologi~ HK1,~         2.22     3
## 10 GO:0045471 response to ethanol              biologi~ ND4,~         1.17     3
```

From the GO Analysis result, we can see that

1. Four of the five genes with highest probability of overrepresentation were activity of hexokinase, glucokinase, fructokinase, and mannokinase.
2. These kinases of the liver are enzymes that catalyze the transfer of high-energy ATP molecules (e.g., sugar, proteins) to substrate by adding phosphates to the ATP molecules.
3. Kinases are critical in metabolism and are used extensively to transmit signals and regulate complex processes in cells (and many other complex cellular pathway). Therefore, this might shed some light into how Epipedobates handled their alkaloid diets differently between cryptic and aposematic species

```
ggplot(graph_data, aes(x = FWER_overrep, y = node_id)) +
  geom_point(aes(size = tally, fill = ontology), alpha = 0.75, shape = 21) +
  scale_size_continuous(limits = c(1, 15), range = c(1,17), breaks = c(1, 2, 3, 4)) +
  labs(x= "Family-wise error rates", y = "Node ID", size = "Number of Genes", fill = "")  +
  theme(legend.key=element_blank(),
        axis.text.x = element_text(colour = "black", size = 12, face = "bold", angle = 90, vjust = 0.3,
        axis.text.y = element_text(colour = "black", face = "bold", size = 11),
        legend.text = element_text(size = 10, face ="bold", colour ="black"),
        legend.title = element_text(size = 12, face = "bold"),
        panel.background = element_blank(), panel.border = element_rect(colour = "black", fill = NA, si
        legend.position = "right")
```



4.b. Plotting GO Figure