QCB455/COS551 Homework 3

Motif Finding, BLAST, ChIP-seq, Clustering, and GO Term Enrichment

William Svoboda (wsvoboda)

Last edited November 05, 2021

Contents

Collaboration Statement

I talked with Brendan McManamon (bm18, student), Debby Park (debbyp, student), and Sara Schwartz (sarats, student) about this homework.

1 Motif Finding

1.1 Theoretical Model

1. Build a position frequency matrix to model this transcription factor, using a pseudocount of 0.25.

```
# Given sequences
seq_data <- Biostrings::BStringSet(c("ATGT", "ACGT", "ACCT", "ATCT"))

# Count number of times each nucleotide appears in each column of aligned
# binding sites
counts <- Biostrings::consensusMatrix(seq_data) %>%
    `colnames<-`(1:4)

# Correct for zero frequency case
pseudocount_weight <- 0.25
pseudocounts <- counts + pseudocount_weight

# Normalize each column to sum to 1 (probabilities)
pfm <- apply(pseudocounts, 2, norm <- function(x) {
    return(x/sum(x))
})

# Print position frequency table
knitr::kable(pfm, caption = "Position frequency matrix")
```

Table 1: Position frequency matrix

	1	2	3	4
A	0.85	0.05	0.05	0.05
C	0.05	0.45	0.45	0.05
G	0.05	0.05	0.45	0.05
T	0.05	0.45	0.05	0.85

2. Suppose that the genome this transcription factor is found in has a nucleotide composition of A 40%, C 10%, G 10% and T 40%. Give log-odds scores for searching for binding sites for this transcription factor within the sequence TATGT.

```
# Use given nucleotide composition to find odds
nuc_comps <- c(0.4, 0.1, 0.1, 0.4)
odds <- t(sapply(1:nrow(pfm), function(i) pfm[i, ]/nuc_comps[i])) %>%
    `rownames<-`(c("A", "C", "G", "T"))

# Find position weight matrix
pwm <- t(sapply(1:nrow(odds), function(i) log2(odds[i, ]))) %>%
    `rownames<-`(c("A", "C", "G", "T"))

# Represent each possible window of length 4 for the sequence 'TATGT' and its
# reverse complement 'ACATA'
seq_windows <- c("TATG", "ATGT", "ACAT", "CATA")
seq_windows_df <- read.fwf(file = textConnection(seq_windows), widths = c(1, 1, 1,
    1), colClasses = "character")

# Search and calculate log-odds scores
```

```r
for (i in 1:length(seq_windows)) {
    score <- 0
    for (j in 1:nchar(seq_windows[i])) {
        row_index <- which(rownames(pwm) == substr(seq_windows[i], j, j))
        score <- score + pwm[row_index, j]
    }
    seq_windows_df$Score[i] <- score
}

# Print log-odds scores
knitr::kable(cbind(seq_windows, seq_windows_df$Score) %>%
    `colnames<-`(c("window", "score")), caption = "Log-odds scores")
```

Table 2: Log-odds scores

window	score
TATG	-10
ATGT	4.5147756853853
ACAT	1.34485068394299
CATA	-10

1.2 Uncovering Regulators

1.2.1 MEME

1. Selected 'Classic mode' (provide one set of sequences and MEME discovers motifs enriched in the set)
2. Selected 'DNA, RNA or Protein' for sequence alphabet
3. Uploaded the file **DNA-seqs.txt** for primary sequences
4. Selected 'One Occurrence Per Sequence' for site distribution
5. Selected 1 as the number of motifs to find
6. Clicked 'Submit/Download' to send the discovered motif to Tomtom

- **Figure 1 shows the motif locations discovered by MEME.**

1.2.2 Tomtom

1. Kept the input query motifs on 'Submitted motifs' (from MEME)
2. Selected 'ECOLI (Escherichia coli) DNA' and 'Swiss Regulon' for the motif database
3. Kept 'Search with one motif' selected

- **Table 3 shows the results given by Tomtom. The putative regulator is most likely CRP 25-83.**

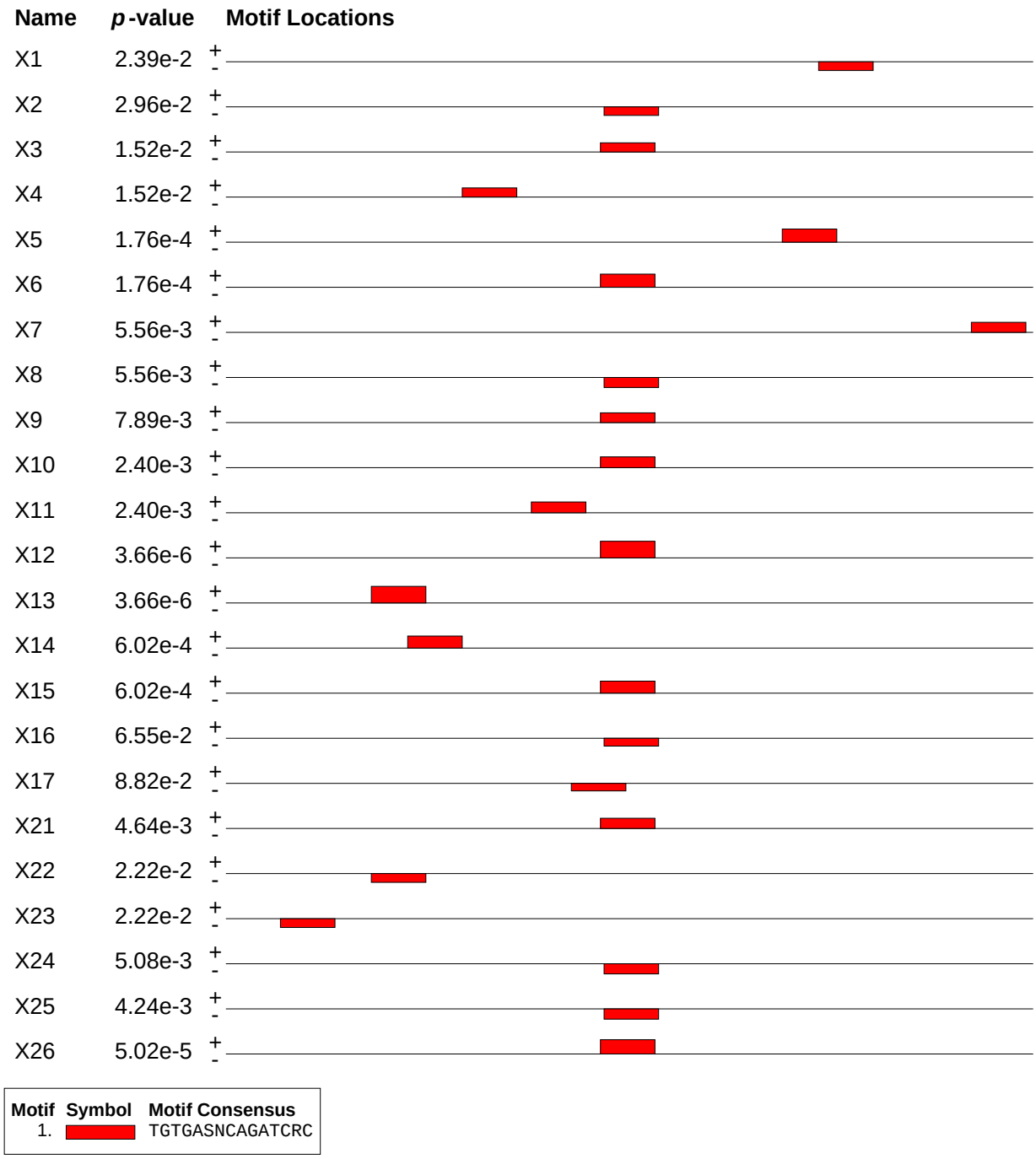


Figure 1: Motif locations

Table 3: Tomtom results

Target_ID	p.value	E.value	q.value	Overlap	Target_consensus
CRP_25-83	0.0000004	0.0000358	0.0000686	15	AAATTTGTGATCTAGATCACAAAAA
CRP_25-152	0.0000013	0.0001279	0.0001226	15	TAAATTGTGATTCATATCACATTTT
CRP_25-12	0.0000057	0.0005567	0.0002667	15	AAAAATGTGACATAGATCACATTTA
CRP_25-18	0.0000440	0.0042663	0.0016349	15	GGAAATGTGATGGAGATCACATAAA
FNR_17-11	0.0011375	0.1103370	0.0302019	14	TTGATGTAAATCAATTT
DeoR_19-5	0.0076562	0.7426570	0.1422980	15	TTTGTGAGAATGCTAACAT
GlpR_20-5	0.0145697	1.4132600	0.2461730	15	AATATGAGCAAAAACGAACA
GntR_18-1	0.0169142	1.6406700	0.2619700	15	AATGTTACCCGTAACATA
H_NS_28-19	0.0290968	2.8223900	0.4159920	15	ATTAAAATGTGAAATATAAAATAATATT
Fur_21-21	0.0511565	4.9621800	0.6338580	15	TAGTAATCGAAATTACCCTCG
DnaA_10-1	0.0604846	5.8670100	0.6381120	10	TTTATCCACA
FNR_17-9	0.0617998	5.9945800	0.6381120	12	AAATGAGATCAAAAAAA
NarL_9-1	0.0698885	6.7791800	0.6598360	9	TACCCATAT
NarP_8-3	0.0710041	6.8874000	0.6598360	8	AGGGGTAT
NorR_11-0	0.0843855	8.1853900	0.7468460	10	GTCATATTGAC
Fis_26-108	0.0976290	9.4700100	0.8247810	15	AAAAACGTGCGAAAAATCGGCAAATT

2 BLAST

2.1 Multiple testing in BLAST

1. The non-redundant BLAST database has $6*10^7$ nucleotide sequences. Suppose you BLAST your sequence and get a sequence hit with a p-value of 10^{-10}. You have performed $6 * 10^7$ sequence comparisons, what is the probability that at least one of the observations will be called significant by chance?

- **We can represent generally the chance of at least one event ocurring in terms of its complement. Using the given p-value and the number of comparisons (events), we can then derive this probability and find its numerical value:**

$$P(\geq 1 \text{ event significant by chance}) = 1 - P(\text{no events significant by chance}) \quad (1)$$
$$= 1 - (1 - 10^{-10})^{6*10^7} \quad (2)$$
$$= \sim 0.006 \quad (3)$$

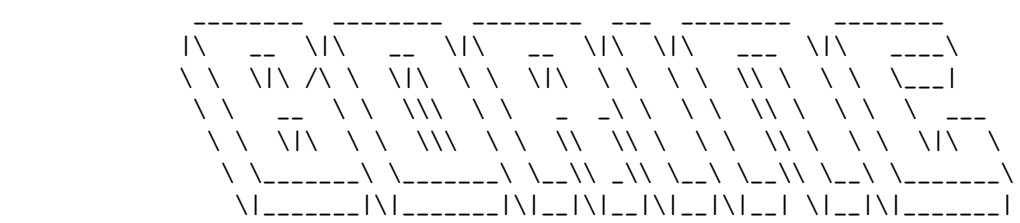
2. Instead of using p-values, BLAST reports E-values. What is it, and what is your estimate of the E-value based on part 1?

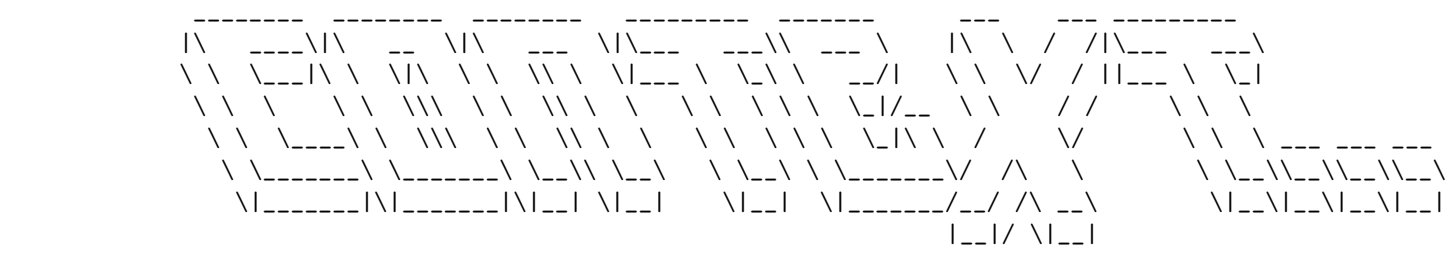
- **An E-value, or Expect value, describes the expected number of hits that will occur by chance when searching a database of a particular size. The lower the E-value for a match, the more significant that match is. We estimate the number of hits that occur by chance by multiplying the given p-value by the number of comparisons:**

$$\text{E-value} = 10^{-10} * 6 * 10^7 \quad (4)$$
$$= 0.006 \quad (5)$$

2.2 Primers for COVID screening by RT-PCR

2.2.1 Background on PCR





2.2.2 Question on primer design

1. Which SARS-CoV-2 gene is amplified by this primer pair?

- **The gene N is amplified.**

2. Are these primers specific to SARS-CoV-2, or do they amplify human mRNA sequences?

- **These primers are specific to SARS-CoV-2.**

3. Consider shortening each of these primers to include only the first 18 nucleotides. Find the human transcripts that could be amplified by this PCR reaction. What are their lengths? The PCR reaction offers some control over the maximum product length. Is there a product length cutoff that allows you to exclude human transcripts?

- **The human transcripts QRICH2 (variants 3, 1, X3, X2, X4; product length 383), CKAP5 (variants 2, 1; product length 2774), and NKD1 (variant 1; product length 2777) are amplified by this PCR reaction. There is no product length cutoff that excludes human transcripts.**

3 P-values in a Simulated ChIP-seq Gene

1. Simulate random Poisson counts. Draw random counts assuming they follow the Poisson distribution with parameter λ as above. Draw $n = 4$ replicates for each condition. Report a table with the counts you obtained on one random trial.

```
counts_c1 <- rpois(n = 4, lambda = 10)
counts_c2 <- rpois(n = 4, lambda = 50)
knitr::kable(cbind(counts_c1, counts_c2) %>%
    `colnames<-`(c("$\\lambda_1$", "$\\lambda_2$")), caption = "Random counts from one trial")
```

Table 4: Random counts from one trial

λ_1	λ_2
12	50
10	52
8	57
11	41

2. Compute and report the sample mean $\bar{c}_i$ and sample variance σ_i^2 in each condition i.

```
mean_c1 <- mean(counts_c1)
mean_c2 <- mean(counts_c2)
var_c1 <- var(counts_c1)
var_c2 <- var(counts_c2)
```

- **For condition 1,** $\bar{c}_1 = 10.25$ **and** $\sigma_1^2 = 2.9166667$**. For condition 2,** $\bar{c}_2 = 50$ **and** $\sigma_2^2 = 44.6666667$**.**

3. Calculate and report the following t-statistic: $t = \frac{\bar{c}_1 - \bar{c}_2}{\sqrt{\sigma_1^2 + \sigma_2^2}}\sqrt{n}$

```
t <- ((mean_c1 - mean_c2)/(sqrt(var_c1 + var_c2))) * sqrt(4)
```

- $t = -11.5249672$

4. Calculate and report the two-tailed p-value of differential protein-DNA interaction using t.

```
pval <- 2 * pt(q = abs(t), df = (2 * (4 - 1)), lower.tail = FALSE)
```

- **The two-tailed p-value is** 2.5648944×10^{-5}**. The size of the p-value is very small, meaning we are correctly rejecting the null hypothesis (which would say that both conditions are the same).**

5. Repeat steps 1-4 with $\lambda_1 = \lambda_2 = 10$. Report the t-statistic and p-value you got. What does the size of your p-value indicate?

```
# Simulate poisson counts
counts_same1 <- rpois(n = 4, lambda = 10)
counts_same2 <- rpois(n = 4, lambda = 10)
# Compute sample mean and variance
mean_same1 <- mean(counts_same1)
var_same1 <- var(counts_same1)
mean_same2 <- mean(counts_same2)
var_same2 <- var(counts_same2)
# Calculate t-statistic
t_same <- ((mean_same1 - mean_same2)/(sqrt(var_same1 + var_same2))) * sqrt(4)
# Calculate two-tailed p-value
pval_same <- 2 * pt(q = abs(t_same), df = (2 * (4 - 1)), lower.tail = FALSE)
```

- **The value of the t-statistic is -0.3611576 and the two-tailed p-value is 0.7303578. The size of the p-value is very large, indicating that we are correctly rejecting the alternative hypothesis (which would say that the two conditions are different).**

6. What is the two-tailed p-value of a t-statistic of 3.5 (with the same degrees of freedom as earlier)?

```
pval_final <- 2 * pt(q = abs(3.5), df = (2 * (4 - 1)), lower.tail = FALSE)
```

- **The two-tailed p-value is 0.0128263.**

4 GO term enrichment

```
# Import datasets
ORFs = read.table("./hw3data/q4/all_yeast_orfs.txt", stringsAsFactors = FALSE, col.names = "ORF")$ORF
DE_ORFs = read.table("./hw3data/q4/differentially_expressed_orfs.txt", stringsAsFactors = FALSE,
    col.names = "ORF")$ORF
GO_Annotation = read.table("./hw3data/q4/go_bp_matrix.txt", stringsAsFactors = FALSE,
    header = TRUE, check.names = FALSE, row.names = 1)
GO_Terms = read.table("./hw3data/q4/go_bp_to_annotation.txt", stringsAsFactors = FALSE,
    sep = "\t", check.names = FALSE, quote = "", col.names = c("GOterm", "Description"))
```

4.1 Data Validation

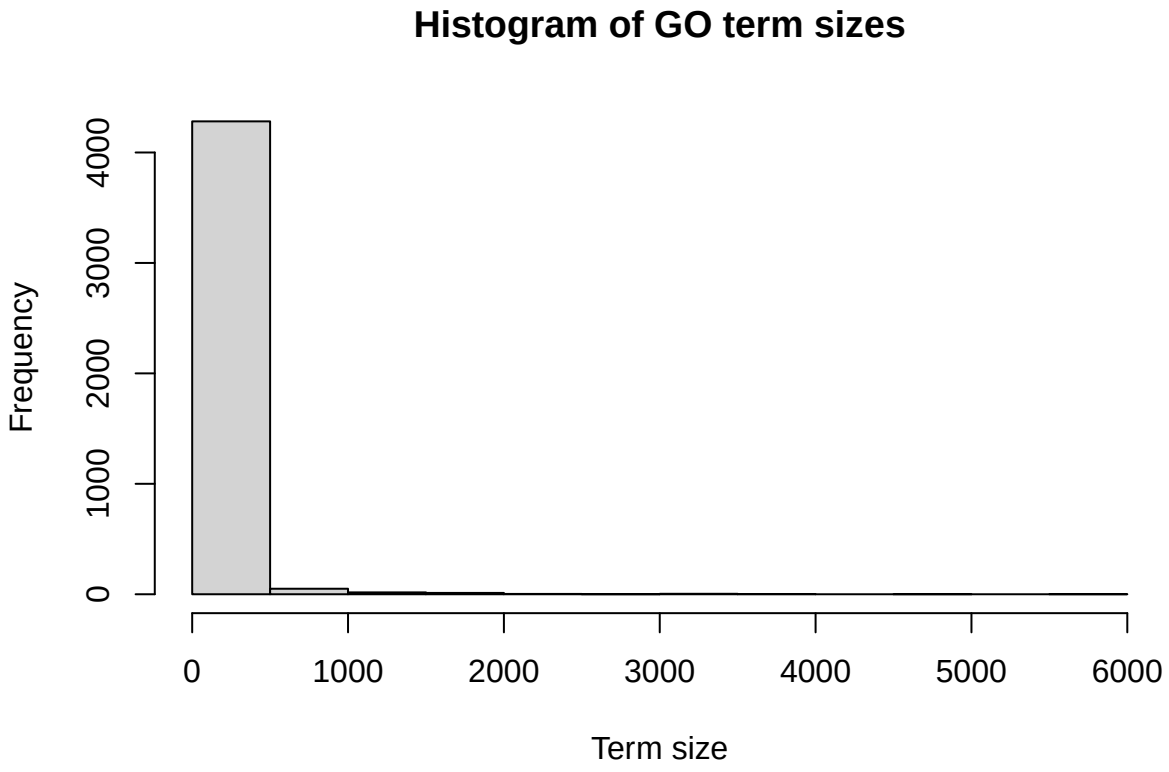
(a) Are all genes in your differentially expressed list included in the list of tested genes? If not, which genes are missing from the tested genes? Why do you think they are missing? If possible, fix the differentially expressed list. Otherwise, eliminate these genes from the differentially expressed list.

```
# Check if differentially experessed list is subset of tested genes list
is_subset <- all(DE_ORFs %in% ORFs)

# Find which genes are missing from the tested genes
if (!is_subset) {
    # NOTE: The missing genes are actually in ORFs but are mispelled (listed
    # with an 'O' instead of a '0'), meaning that ORFs would have duplicate
    # genes if the mispellings were corrected and reinserted into the list
    missing_genes <- setdiff(DE_ORFs, ORFs)

    # Eliminate duplicates and missing genes from the differentially expressed
    # list
    Corrected_DE_ORFs <- unique(DE_ORFs[!DE_ORFs %in% missing_genes])
}
```

- **There are missing genes, specifically the genes YNRO20C, YILO47C, YLR2O1C. The missing genes appear to be mispellings of existing genes in ORFs, so there might have been an error in preparing or cleaning the data after the experiment.**

(b) Calculate the size of each GO term, i.e. the number of genes annotated to it. Plot the distribution of GO term sizes. What is the largest GO term? Report the GO term id, name and size.

```
# Find number of genes annotated to each GO term
GO_Terms$Size <- apply(GO_Annotation, 2, count <- function(x) {
    sum(x)
})

# Plot distribution of GO term sizes
hist(GO_Terms$Size, main = "Histogram of GO term sizes", xlab = "Term size")
```




```
# Find the largest GO term
index <- which.max(GO_Terms$Size)
max_GO_term <- GO_Terms[index, ]
knitr::kable(cbind(index, max_GO_term$GOterm, max_GO_term$Size) %>%
    `colnames<-`(c("id", "name", "size")), caption = "Max GO term")
```

Table 5: Max GO term

id	name	size
949	GO:0008150	5911

(c) Calculate the number of GO terms for gene YDR026C. Is GO:0006725 "cellular aromatic compound metabolic process" among the terms?

```
# Subset corresponding row by nonzero elements
gene_GO_terms <- GO_Annotation[which(rownames(GO_Annotation) == "YDR026C"), ] %>%
    select_if(colSums(.) != 0)

gene_GO_terms_size <- length(gene_GO_terms)
is_among <- "GO:0006725" %in% colnames(gene_GO_terms)
```

- **There are 86 GO terms for gene YDR026C. GO:0006725 is among the terms.**

4.2 Enrichment analysis

Use the hypergeometric test to calculate the enrichment of each GO term among the list of differentially expressed genes from your experiment.

```
# Optimization
rnames <- rownames(GO_Annotation)
```

```
# Number of genes in background
N <- length(ORFs)

# Number of differentially-expressed genes
n <- length(Corrected_DE_ORFs)

GO_Terms$K <- GO_Terms$Size
for (i in 1:nrow(GO_Terms)) {
    # Number of differentially-expressed genes with GO term
    GO_Terms$k[i] <- length(intersect(Corrected_DE_ORFs, rnames[which(GO_Annotation[,
        i] > 0)]))
    # Calculate p-values
    GO_Terms$PValue[i] <- phyper(GO_Terms$k[[i]] - 1, GO_Terms$K[i], N - GO_Terms$K[i],
        n, lower.tail = FALSE)
    # Calculate fold enrichments
    GO_Terms$FoldEnrichment[i] <- (GO_Terms$k[[i]]/n)/(GO_Terms$K[i]/N)
}

# Correct p-values
GO_Terms$Corrected_PValue <- p.adjust(GO_Terms$PValue, method = "bonferroni")

# Print out ranked list of all significant GO terms
GO_Terms$id <- rownames(GO_Terms)
significant_terms <- GO_Terms %>%
    filter(Corrected_PValue < 0.05) %>%
    arrange(Corrected_PValue)
knitr::kable(cbind(significant_terms$id, significant_terms$GOterm, significant_terms$K,
    significant_terms$k, format(significant_terms$FoldEnrichment, digits = 4), format(significant_terms$
        digits = 4), format(significant_terms$Corrected_PValue, digits = 4)) %>%
    `colnames<-`(c("id", "name", "# background", "# DE", "enrichment", "p-value",
        "corrected p-value")), caption = "Significant GO terms by corrected p-value")
```

Table 6: Significant GO terms by corrected p-value

id	name	# background	# DE	enrichment	p-value	corrected p-value
845	GO:0007005	315	56	8.734	1.305e-40	5.707e-37
3963	GO:0097034	26	21	39.680	2.877e-32	1.258e-28
2173	GO:0033108	33	22	32.751	1.292e-30	5.649e-27
972	GO:0008535	20	18	44.214	1.733e-29	7.578e-26
2218	GO:0033617	18	17	46.398	9.553e-29	4.176e-25
3586	GO:0070271	262	41	7.688	1.531e-26	6.692e-23
2708	GO:0043623	167	33	9.708	1.465e-24	6.406e-21
566	GO:0006461	251	36	7.046	8.186e-22	3.579e-18
1774	GO:0022904	24	12	24.564	6.324e-15	2.765e-11
2581	GO:0042773	24	12	24.564	6.324e-15	2.765e-11
2582	GO:0042775	24	12	24.564	6.324e-15	2.765e-11
3810	GO:0071822	405	36	4.367	7.322e-15	3.201e-11
388	GO:0006119	25	12	23.581	1.196e-14	5.227e-11
1773	GO:0022900	25	12	23.581	1.196e-14	5.227e-11
3539	GO:0065003	456	37	3.986	5.175e-14	2.262e-10
2111	GO:0032543	101	19	9.242	6.878e-14	3.007e-10
2857	GO:0045333	91	18	9.717	1.326e-13	5.797e-10

id	name	# background	# DE	enrichment	p-value	corrected p-value
2345	GO:0034622	400	34	4.176	1.917e-13	8.381e-10
391	GO:0006122	10	8	39.302	1.011e-12	4.419e-09
839	GO:0006996	1283	58	2.221	6.224e-11	2.721e-07
1766	GO:0022607	671	40	2.929	8.809e-11	3.851e-07
1502	GO:0015980	134	18	6.599	1.255e-10	5.488e-07
2726	GO:0043933	611	37	2.975	3.783e-10	1.654e-06
366	GO:0006091	161	18	5.492	2.676e-09	1.170e-05
2745	GO:0044085	991	47	2.330	2.853e-09	1.247e-05
991	GO:0009060	77	13	8.294	3.247e-09	1.420e-05
1335	GO:0010821	22	8	17.864	5.855e-09	2.560e-05
3558	GO:0070129	16	7	21.493	1.191e-08	5.207e-05
1513	GO:0016043	1800	66	1.801	1.373e-08	6.004e-05
3814	GO:0071840	2077	72	1.703	1.672e-08	7.309e-05
727	GO:0006743	13	6	22.674	9.568e-08	4.183e-04
728	GO:0006744	13	6	22.674	9.568e-08	4.183e-04
2517	GO:0042181	13	6	22.674	9.568e-08	4.183e-04
4140	GO:1901661	13	6	22.674	9.568e-08	4.183e-04
4141	GO:1901663	13	6	22.674	9.568e-08	4.183e-04
1336	GO:0010822	14	6	21.054	1.647e-07	7.200e-04
392	GO:0006123	9	5	27.293	3.791e-07	1.657e-03
2217	GO:0033615	10	5	24.564	7.459e-07	3.261e-03
2680	GO:0043461	10	5	24.564	7.459e-07	3.261e-03
532	GO:0006412	363	23	3.113	7.526e-07	3.290e-03
3587	GO:0070272	11	5	22.331	1.345e-06	5.882e-03
2516	GO:0042180	22	6	13.398	3.581e-06	1.565e-02
3560	GO:0070131	13	5	18.895	3.627e-06	1.586e-02
2678	GO:0043457	7	4	28.073	5.448e-06	2.382e-02
3547	GO:0070071	14	5	17.545	5.551e-06	2.427e-02
1299	GO:0010608	144	13	4.435	5.912e-06	2.585e-02
98	GO:0000372	8	4	24.564	1.073e-05	4.689e-02
101	GO:0000376	8	4	24.564	1.073e-05	4.689e-02

4.3 Theoretical question

Imagine that at some point, you realized that the method used to quantify gene expression was restricted to only non-essential genes in the yeast genome. Explain how this could affect your previous analysis and the obtained p-values.

- **If the method used to quantify gene expression was limited to non-essential genes, it would mean that the list of differentially expressed genes was likewise limited. Since non-essential genes were incorrectly over-represented in the analysis, the obtained p-values are far lower than they should be. In other words, the ranked list could contain genes that in reality are far less significant.**

```
sessionInfo(package = NULL)
```

```
## R version 4.1.1 (2021-08-10)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur 10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats4    parallel  stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] Biostrings_2.60.2   GenomeInfoDb_1.28.4 XVector_0.32.0
##  [4] IRanges_2.26.0      S4Vectors_0.30.2    BiocGenerics_0.38.0
##  [7] BiocManager_1.30.16 formatR_1.11        knitr_1.36
## [10] dplyr_1.0.7
##
## loaded via a namespace (and not attached):
##  [1] highr_0.9              pillar_1.6.4           compiler_4.1.1
##  [4] bitops_1.0-7           tools_4.1.1            zlibbioc_1.38.0
##  [7] digest_0.6.28          evaluate_0.14          lifecycle_1.0.1
## [10] tibble_3.1.5           pkgconfig_2.0.3        rlang_0.4.12
## [13] yaml_2.2.1             xfun_0.27              fastmap_1.1.0
## [16] GenomeInfoDbData_1.2.6 stringr_1.4.0          generics_0.1.0
## [19] vctrs_0.3.8            tidyselect_1.1.1       glue_1.4.2
## [22] R6_2.5.1               fansi_0.5.0            rmarkdown_2.11
## [25] purrr_0.3.4            magrittr_2.0.1         ellipsis_0.3.2
## [28] htmltools_0.5.2        utf8_1.2.2             stringi_1.7.5
## [31] RCurl_1.98-1.5         crayon_1.4.1
```