Gene Ontology and Gene Set Enrichment Analysis

Group 7 - Alexander Fastner, Franziska Koller

Introduction

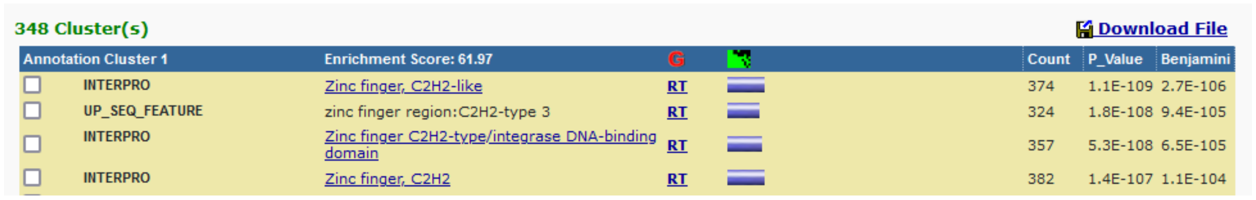
Genes interact in complex networks. Gene Ontology (GO) is an approach to annotate genes based on their functionalities and relationships. GO is a directed acyclic graph containing GO terms (annotations) that describe functions, localizations and processes of genes and proteins.
Gene sets can be classified and pathways between gene sets can be identified (also in combination with other tools and databases such as KEGG or Reactome).
In order to identify genes that are overrepresented in certain pathways or gene sets, Gene Set Enrichment Analysis (GSEA) is an effective method. The genes that appear with a higher number than expected are called **enriched**. Enrichment scores and p-values can be defined and calculated accordingly. The enrichment score quantifies the amount to which the genes of this set are overrepresented (enriched).
Enriched sets and pathways can indicate a functional connection for certain features. The goal of GSEA is to find out whether the gene set under investigation shows a difference in expression (over- or underexpression) for certain genes or even the majority of the gene set. The consequence of such abnormal expression can be the cause of phenotypic differences or diseases.

For this assignment we implement a basic analysis tool for GSEA and interpret the results. The tool evaluates GO terms given in an obo file and provides several scores such as the number of associated genes, the shortest path to an enriched GO term and several p-values.

Analysis

DAVID (**D**atabase for **A**nnotation, **V**isualization and **I**ntegrated **D**iscovery) is a web based tool for gene set enrichment analysis and can help with the functional interpretation of gene sets. It works with multiple heterogeneous and widely distributed public databases and provides several tools to summarize relevant biological patterns.
The input for DAVID are gene lists consisting of <=3000 genes.
It provides functional annotation clustering, a functional annotation chart and a functional annotation table.

For this task 3000 genes that are associated with the provided GO terms (in go.obo) were extracted and analyzed with DAVID. The genes that are connected to the most GO terms were selected. DAVID provides 348 clusters with a maximum enrichment score of 67.97 (cf. Figure 1a). The majority of the genes in Cluster 1 can be associated with the zinc finger protein.

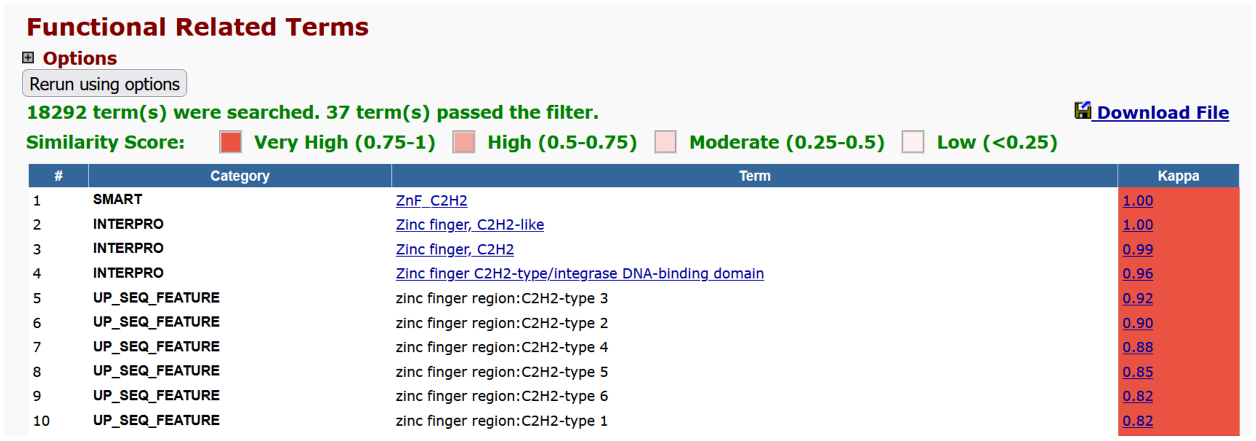


348 Cluster(s)

Download File

Annotation Cluster 1		Enrichment Score: 61.97	G		Count	P_Value	Benjamini
☐	INTERPRO	Zinc finger, C2H2-like	RT	▬	374	1.1E-109	2.7E-106
☐	UP_SEQ_FEATURE	zinc finger region:C2H2-type 3	RT	▬	324	1.8E-108	9.4E-105
☐	INTERPRO	Zinc finger C2H2-type/integrase DNA-binding domain	RT	▬	357	5.3E-108	6.5E-105
☐	INTERPRO	Zinc finger, C2H2	RT	▬	382	1.4E-107	1.1E-104

Figure 1a: Output (head) of functional annotation clustering with DAVID for 3000 genes



Functional Related Terms

⊞ Options

Rerun using options

18292 term(s) were searched. 37 term(s) passed the filter.

Download File

Similarity Score: 🟥 Very High (0.75-1) 🟥 High (0.5-0.75) ▢ Moderate (0.25-0.5) ▢ Low (<0.25)

#	Category	Term	Kappa
1	SMART	ZnF_C2H2	1.00
2	INTERPRO	Zinc finger, C2H2-like	1.00
3	INTERPRO	Zinc finger, C2H2	0.99
4	INTERPRO	Zinc finger C2H2-type/integrase DNA-binding domain	0.96
5	UP_SEQ_FEATURE	zinc finger region:C2H2-type 3	0.92
6	UP_SEQ_FEATURE	zinc finger region:C2H2-type 2	0.90
7	UP_SEQ_FEATURE	zinc finger region:C2H2-type 4	0.88
8	UP_SEQ_FEATURE	zinc finger region:C2H2-type 5	0.85
9	UP_SEQ_FEATURE	zinc finger region:C2H2-type 6	0.82
10	UP_SEQ_FEATURE	zinc finger region:C2H2-type 1	0.82

Figure 1b: Output (top 10) of functional related term from DAVID

GSEA

- mappingtype = ensembl - minsize = 50, maxsize = 500 - obo file = go.obo - root = biological_process - mapping = goa_human_ensembl.tsv - enrich = simul_exp_go_bp_ensembl.tsv	minsize and maxisize are not ignored (consider only GO terms with correct size)	minsize and maxisize are ignored (consider all GO terms)
Total number of genes (only genes that are associated with one of the GO terms)	12 741	15 497
Total number of genes (all genes that appear in the mapping file goa_human_ensembl.tsv)	18 445	
Number of gene sets (number of DAGNodes)	1 567	29 385
Number of leaves in DAG tree	586	14 991
Shortest path to Root	2	1
Longest path to Root	11	13

Table 1: Summary stats for genesets

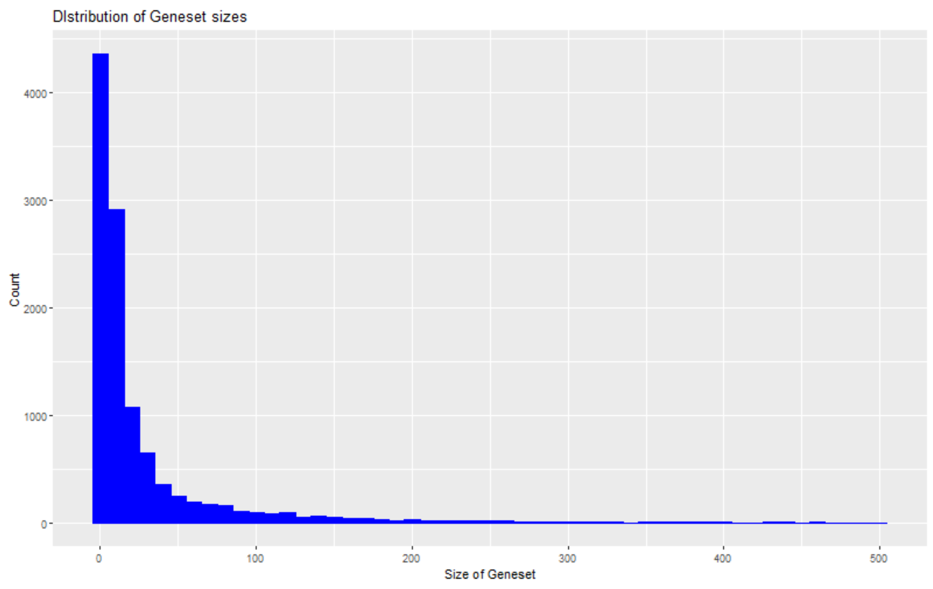


Figure 2: Distribution of Geneset sizes

Figure 2 shows that the majority of the gene sets (over 4000) have a size of only 0-10. Gene sets with a bigger size are much less common.

```
ggplot(as(adjusted, "data.frame"), aes(x = size)) + geom_histogram(binwidth = 10, color = "blue", fill=I("blue"))+
    labs(x = "Size of Geneset", y = "Count", title = "Distribution of Geneset sizes for all Genesets")
```

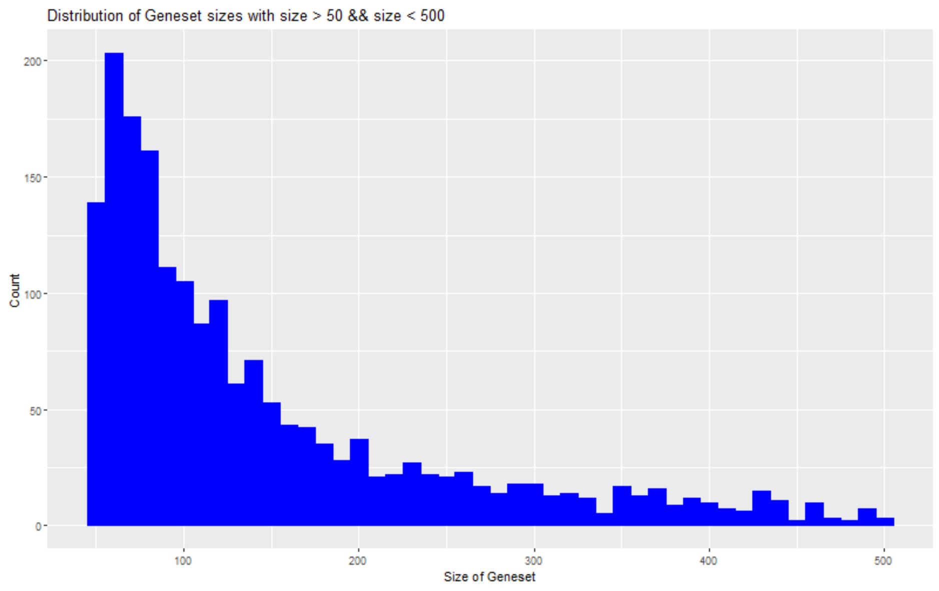


Figure 3: Distribution of Geneset sizes with 50<size<500

If the size of the gene sets is selected to be between 50 and 500, a similar trend can be observed: a gene set with a lower number of genes occurs more often than gene sets with a high number of genes.

```
ggplot(as(filtered, "data.frame"), aes(x = size)) + geom_histogram(binwidth = 10, color = "blue", fill=I("blue"))+
  labs(x = "Size of Geneset", y = "Count", title = "Distribution of Geneset sizes with size > 50 && size < 500")
```

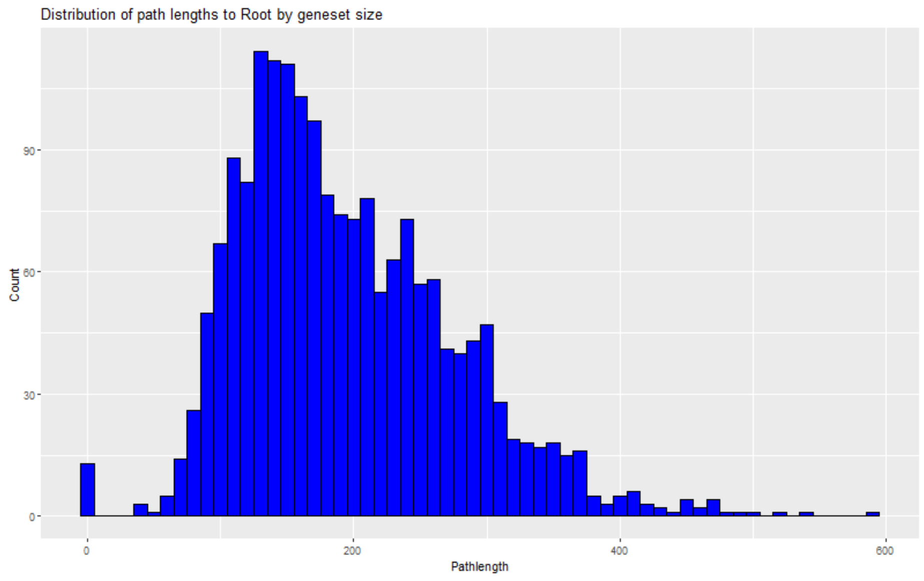


Figure 4: Distribution of path lengths by geneset size.

Depicted here are the number of path lengths (shortest_path_to_a_true) at various lengths. It forms a slightly right-skewed normal distribution.

```
ggplot(as(df, "data.frame"), aes(x = pathlength)) + geom_histogram(binwidth = 10, color = "black", fill=I("blue")) +
  labs(x = "Pathlength", y = "Count", title = "Distribution of path lengths to Root by geneset size")
```

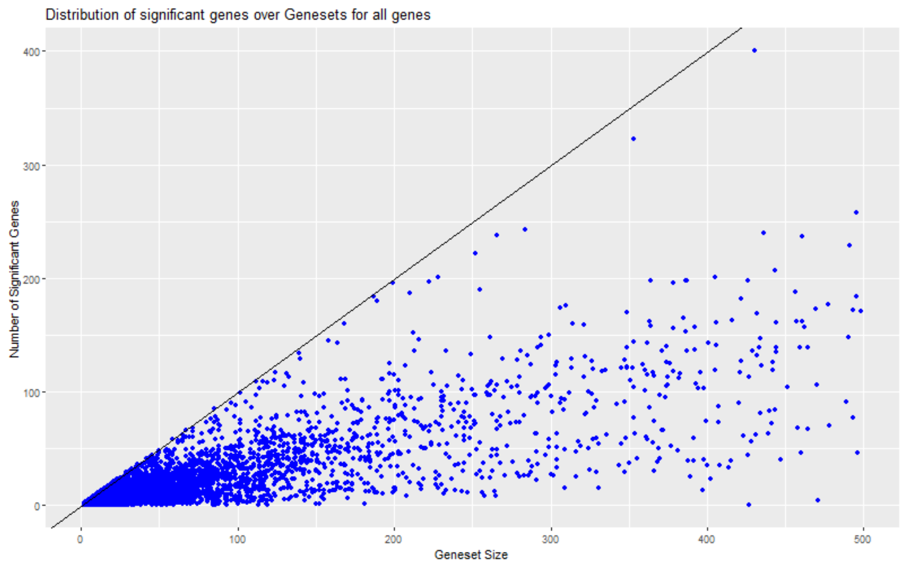


Figure 5: Distribution of Significant genes vs Geneset size.

Seen here is a plot of the number of Significant genes against the respective Geneset size. The black line has a slope of 1 and points on that line reflect a geneset completely comprised of significant genes.

```
ggplot(as(adjusted, "data.frame"), aes(x = size, y = noverlap)) + geom_point(color = "blue", fill=I("black")) +
  geom_abline(intercept=0, slope=1) +
  labs(x = "Geneset Size", y = "Number of Significant Genes", title = "Distribution of significant genes over Genesets for all genes")
```

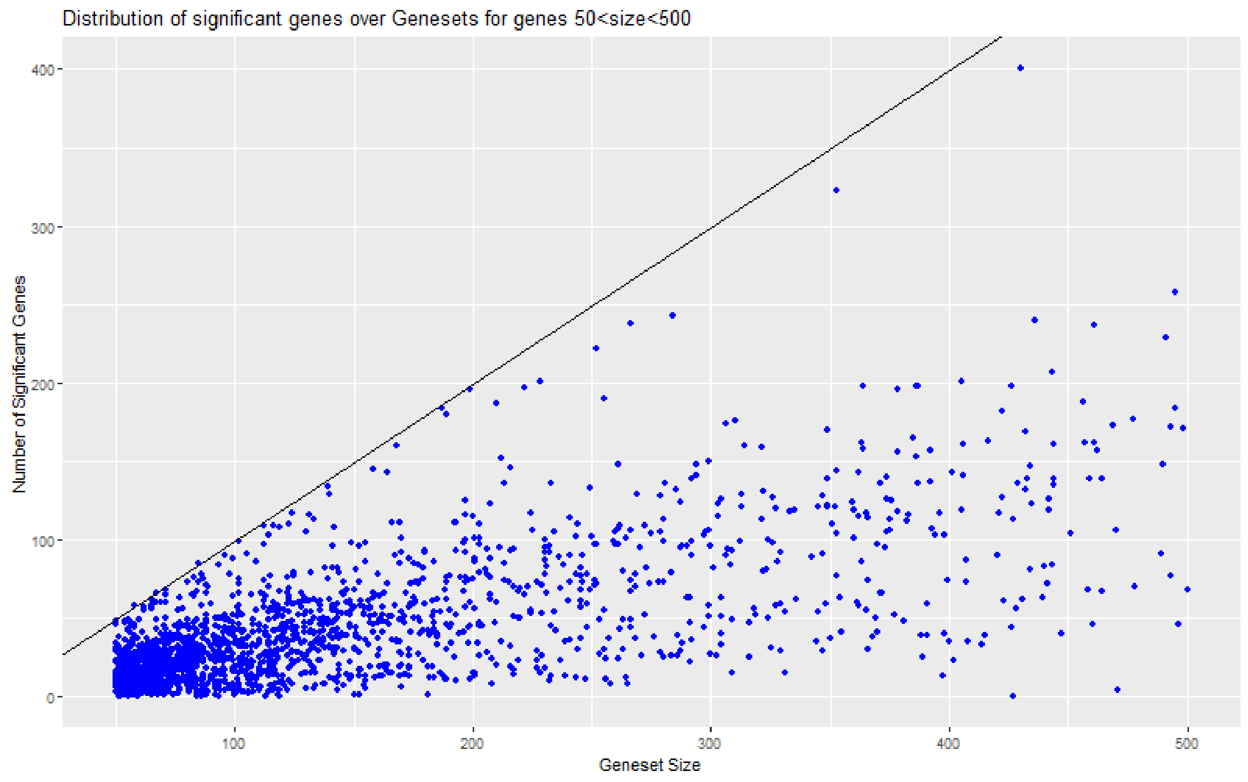
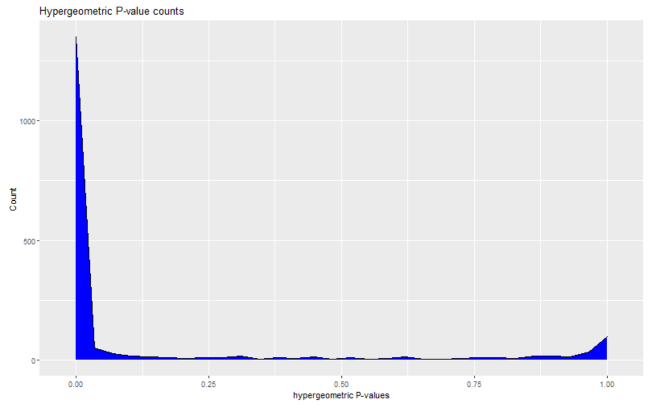


Figure 6: Distribution of Significant genes vs Geneset size for genes with 50<size<500.

This plot shows the number of significant genes for each genesize. The data for this includes only those genesets with more than 50 and less than 500 genes. Aside from a handful of outliers most genesets contain ~50% or less significant genes.

```
ggplot(as(filtered, "data.frame"), aes(x = size, y = noverlap)) + geom_col(binwidth = 10, color = "blue", fill=I("blue")) +
   labs(x = "Geneset Size", y = "Number of Significant Genes", title = "Distribution of significant genes over Genesets for genes 50<size<500")
```



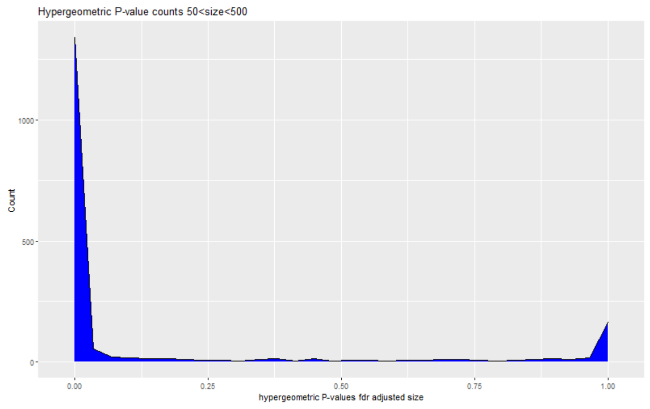
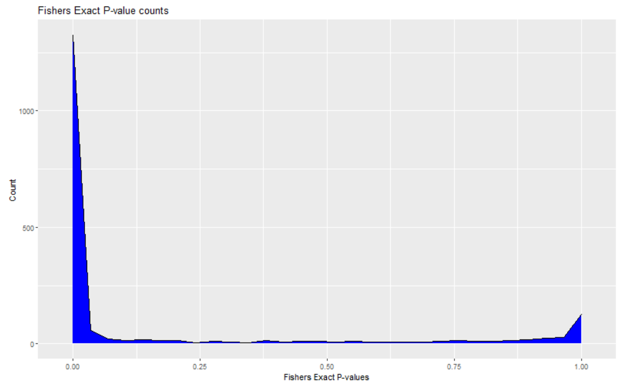


Figure 7: Hypergeometric P-values all genes

Figure 8: Hypergeometric P-values 50<size<500

The analysis of the p-values from the hypergeometric distribution and the Fisher's Exact test shows that most DAGNodes show a very small and therefore significant p-value. There is no big difference between all genes and only the selected ones with a size between 50 and 500.
This indicates that the assignment of the genes to certain GO terms provides useful information about the underlying interactions and processes between the analyzed genes.

```
ggplot(as(filtered, "data.frame"), aes(x = hg_pval)) + geom_area(stat = "bin", color = "black", fill=I("blue")) +
  labs(x = "hypergeometric P-values", y = "Count", title = "Hypergeometric P-value counts")
```



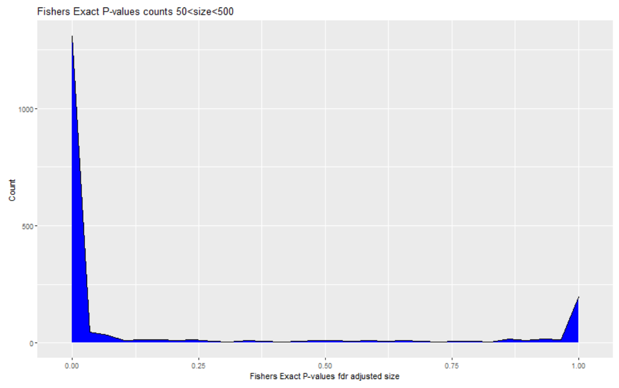


Figure 9: Fisher's Exact P-values all genes

Figure 10: Fisher's Exact P-values 50<size<500

```
ggplot(as(filtered, "data.frame"), aes(x = fej_pval)) + geom_area(stat = "bin", color = "black", fill=I("blue")) +
  labs(x = "Fishers Exact P-values", y = "Count", title = "Fishers Exact P-value counts")
```

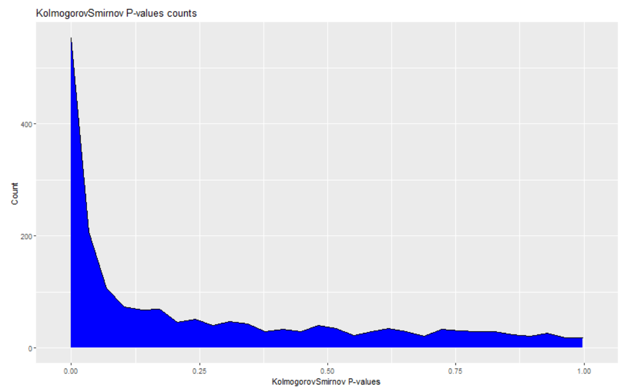


Figure 11: Kolmogorov Smirnov P-values all genes

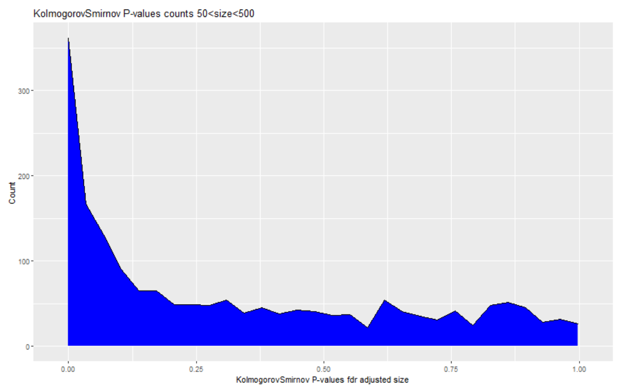


Figure 12: Kolmogorov Smirnov P-values 50<size<500

The Kolmogorov Smirnov test shows a similar pattern to the hypergeometric distribution and the Fisher's Exact test, though there are less low p-values and slightly more spread evenly throughout.

```
ggplot(as(filtered, "data.frame"), aes(x = ks_pval)) + geom_area(stat = "bin", color = "black", fill=I("blue")) +
    labs(x = "KolmogorovSmirnov P-values", y = "Count", title = "KolmogorovSmirnov P-values counts")
```

Code Analysis

obo file

The first input file is the obo file (go.obo). We go through all GO terms of the file but only consider the entries with the correct GO namespace that is given as input parameter (can be biological_process, cellular_component or molecular_function). Every valid GO term is saved as a **DAGNode** Object with several attributes such as the GO id and the corresponding name. The GO ids of the direct parents of the GO term, as provided by the keyword "is_a" by the obo file, are saved in a HashSet<String> parents. All DAGNodes are saved in a HashMap<String, DAGNode> all_DAGNodes. The keys in this HashMap are the GO ids, the values are the DAGNode Objects.
In order to associate each DAGNode not only with its direct parents, but also its non-direct parents (i.e. all the GO terms that can be reached via the is_a relationships), we go through the DAGNode tree saved in the hashmap and add the non-direct parents to each DAGNode in a second HashSet<String> allParents recursively. All possible paths to an end point (i.e. a GO term that has no parents) are saved for each DAGNode as Strings in a HashSet<String> correct_paths.

mapping file

The mapping file reader distinguishes between mappingtype “ensembl” (the mapping file provides all associated GO ids per gene in *tsv* format) or mappingtype “go” (the mapping file provides all associated GO ids per gene in *gaf* format).
For mappingtype “go” we collect all genes and their associated GO ids in a HashMap<String, **Gene**> and go through the HashMap afterwards to add each gene to all associated GO ids (and also to all of their parents) in the DAGNode tree. All GO ids that are connected to the gene are saved in a HashSet<String> all_GO_ids_of_gene in the Gene object.
For mappingtype “ensembl” we directly save the genes to each DAGNode and add the associated GO ids and their parents to the HashSet<String> all_GO_ids_of_gene.
Genes that are not connected to any GO term in the tree are disregarded.

enrich file

The enrich file provides a log2 fold change estimation of the differential expression of each gene (later used for the Kolmogorov Smirnov distribution) and whether the gene is significant or not. This information is simply added to each gene object. It also specifies which GO terms are enriched. If a GO term is enriched the boolean isEnriched of this DAGNode is set to true.

calculations for output

size

The next step was to go through all DAGNodes and count the number of significant genes (as defined by the enrich file) per DAGNode. Only GO terms with a size between minsize and maxsize as specified by the input parameters are considered.

shortest_path_to_a_true

This path specifies the shortest path to the next enriched DAGNode.
For each DAGNode we compute all possible paths to each enriched DAGNode.
First all common ancestors of the DAGNode and the enriched node are identified. Common ancestors are those GO terms that are in the HashSet allParents of *both* nodes. The path to each common ancestor is computed (1.) from the DAGNode and (2.) from the enriched node (the total path length is the sum of these two paths minus one). For this we go through all correct_paths and find, if present, the lengths of all paths to the common ancestor and return the shortest one. This approach is correct since all possible paths are checked and only the shortest ones are saved.

p-values

Now the p-values for all GO terms can be computed. The hypergeometric distribution and the Fisher’s Exact Test were computed with all significant genes occurring in the tree and the number of genes associated with each DAGNode. The Kolmogorov Smirnov distribution was calculated with background and in-set distribution. After saving all p-values a multiple testing correction (Benjamini Hochberg) was performed.

second output file

If the second output file is required by the input parameters we go through all genes and all DAGNode pairs where both DAGNodes are associated with the gene are identified and analyzed. If the DAGNodes are descendants/ascendants of each other they are classified as is_relative. The path length between the two nodes is computed with the same method that was used for shortest_path_to_a_true, only that the path does not lead to a true but to the second DAGNode. To get the number of gene ids associated with both DAGNodes we check their HashSets containing all genes connected to them and find the size of the intersection. We then compute the ratio of the intersection to the number of all associated genes for both DAGNodes and save the maximum.

The actual time usage of our program on the Abgabe Server is approximately four minutes (246s).