Single Peptide Results Analysis

Caleb Easterly

August 7, 2018

Introduction

The single peptide analysis methods and results are contained in this document.

Five peptide analysis

The five peptides are as follows:

```
fivepep <- c('AFLPGSLVDTRPVR',
             'DIAMQIAAVNPTYLNREEVPTEVIEHEK',
             'DLFKNPIHPYTK',
             'DVTIEAVNSLYEK',
             'EVPDWAAQLNENTNVKGLRIAVPK')
```

Unipept

Process

- Paste the tabular list of peptides into the Unipept 'Metaproteomics Analysis' web application (https://morty.ugent.be/mpa)

- Parameters:
  - Equate I and L: FALSE
  - Filter duplicate peptides: FALSE
  - Advanced missed cleavage handling: TRUE
- Download results
- Annotate each peptide with only the GO terms that are present in 5% or more of the proteins (percentages are returned in GO term column)

```
library(dplyr)
library(stringr)
cov_pat <- "\ \\(.{3,4}\\)"
uni <- read.csv('unipept_20_peptides_result.csv',
                stringsAsFactors = FALSE) %>%
    select(peptide,
           uni_go_bp = GO..biological.process.,
           uni_go_mf = GO..molecular.function.,
           uni_go_cc = GO..cellular.component.) %>%
    filter(peptide %in% fivepep) %>%
    mutate(uni_go_bp = str_replace_all(string = uni_go_bp, pattern = cov_pat, replacement = ""),
           uni_go_mf = str_replace_all(string = uni_go_mf, pattern = cov_pat, replacement = ""),
           uni_go_cc = str_replace_all(string = uni_go_cc, pattern = cov_pat, replacement = ""))
```

eggNOG mapper

- Use the Galaxy version of eggNOG mapper, on Galaxy-P
- Parameters:
  - Annotation type: DIAMOND
  - Scoring matrix and gap costs: PAM30, 9 and 1
  - Taxonomic Scope: Bacteria
  - Orthologs: use all orthologs
  - Gene Ontology evidence: use non-electronic terms
  - Seed Orthology Search Options
    - Min e-value: 200000
    - Min bit score: 20
- Download and compare GO terms

```
em <- read.delim("eggnog_mapper_20_sequences_results.tabular",
                 stringsAsFactors = FALSE,
                 header=FALSE) %>%
    select(peptide = V1, em_prot = V2, em_go = V6, em_gene = V5, em_descript = V13) %>%
    filter(peptide %in% fivepep) %>%
    mutate(em_go = str_replace_all(em_go, pattern = ",", replacement = "; "))
```

BLASTP against UniProt

- Use the UniProtKB BLAST web search on each peptide, one-by-one
- Parameters
  - Target database: UniProtKB
  - E-Threshold: 10
  - Matrix: Auto
  - Filtering: None
  - Gapped: Yes
  - Hits: 50
- For each peptide, download the result list and get all GO terms and TaxID associated with that peptide
- To match Unipept, annotate each peptide with only the GO terms that are present in 5% or more of the proteins
- Get the most frequent protein name
- For taxonomy, we can also calculate the lowest common ancestor of each peptide (TODO)

```
peptide <- rep(0, 5)
blast_go <- rep(0, 5)
files <- list.files('uniprot_blastp_outputs')
for (i in 1:5){
    peptide[i] <- fivepep[i]
    result <- read.delim(paste('uniprot_blastp_outputs', paste(fivepep[i], '.tab', sep=""), sep="/"),
                    stringsAsFactors = FALSE,
                    na.strings = c("", "NA", "NaN"))
    gos <- table(unlist(str_split(result$Gene.ontology.IDs, "; ")))/50
    blast_go[i] <- paste(names(gos)[which(gos > 0.05)], collapse = "; ")
}
blast <- data.frame(peptide, blast_go, stringsAsFactors = FALSE)
```

MetaGOmics

- Upload HOMD to metaGOmics

- Parameters:
    - Uniprot database: Uniprot sprot
    - Blast e-value cutoff: 1e-10
    - Use only top hit?: TRUE
- Result URL: https://www.yeastrc.org/metagomics/viewUploadedFasta.do?uid=42jgJAcLHHZBoRQk
- One-by-one, upload peptides and run
- Download results individually, combine into table

```
peptide <- rep(0, 5)
mg_go <- rep(0, 5)
dir <- 'metaGOmics_single_peptides_outputs/'
files <- list.files(dir)
for (i in 1:5){
    peptide[i] <- fivepep[i]
    result <- read.delim(paste(dir, paste(fivepep[i], '.txt', sep=""), sep=""),
                  stringsAsFactors = FALSE,
                  na.strings = c("", "NA", "NaN"),
                  comment.char = "#")
    gos <- result$GO.acc
    mg_go[i] <- paste(gos, collapse = "; ")
}
mg <- data.frame(peptide, mg_go, stringsAsFactors = FALSE)
```

Combine all of the results:

```
all_results <- plyr::join_all(list(em, blast, mg, uni), by = "peptide")
```

All of the results are below:

```
library(pander)
# knitr::kable(all_results)
pander::pander(all_results, split.cell = 80, split.table = Inf)
```

peptide	em_prot	em_go	em_gene	em_descript	blast_go	mg_go	uni_go_bp	uni_go_mf	uni_go_cc
AFLPG[illegible]	[illegible]	GO:0003674; GO:0003676; GO:0003723; GO:0003729; GO:0003735; GO:0005198; GO:0005488; GO:0005575; GO:0005622; GO:0005623; GO:0005737; GO:0005829; GO:0005840; GO:0006412; GO:0006417; GO:0008150; GO:0008152; GO:0009058; GO:0009059; GO:0009889; GO:0009890; GO:0009892; GO:0009987; GO:0010467; GO:0010468; GO:0010556; GO:0010558; GO:0010605; GO:0010608; GO:0015935; GO:0016020; GO:0017148; GO:0019222; GO:0019538; GO:0022626; GO:0022627; GO:0030529; GO:0031323; GO:0031324; GO:0031326; GO:0031327; GO:0032268; GO:0032269; GO:0032991; GO:0034645; GO:0043170; GO:0043226; GO:0043228; GO:0043229; GO:0043232; GO:0044237; GO:0044238; GO:0044249; GO:0044260; GO:0044267; GO:0044391; GO:0044422; GO:0044424; GO:0044444;	RPSA	thus facilitating recognition of the initiation point. It is needed to translate mRNA with a short Shine-Dalgarno (SD) purine-rich sequence (By similarity)	GO:0003723; GO:0003735; GO:0005840; GO:0006412	GO:0005840; GO:0005198; GO:1901363; GO:0006412; GO:0006417; GO:0017148; GO:0008150; GO:0010556; GO:0015935; GO:0010558; GO:0034248; GO:0034249; GO:0005575; GO:0032991; GO:0008152; GO:0044391; GO:0000028; GO:0043603; GO:0043604; GO:0003674; GO:0005737; GO:0003676; GO:0010605; GO:0043043; GO:0010608; GO:0050789; GO:0050794; GO:0044422; GO:0044424; GO:2000112; GO:2000113; GO:0065003; GO:0065007; GO:0010629; GO:0071826; GO:0044444; GO:0044445; GO:0044446; GO:0009058; GO:0009059; GO:0006518; GO:0071840; GO:0016043; GO:0044464; GO:0003723; GO:0043933; GO:0003729; GO:0006807; GO:0030529; GO:0003735; GO:0034622; GO:0032268; GO:0032269; GO:0060255; GO:0022607; GO:0009889; GO:0009890; GO:0009892; GO:0051171;	GO: 0006412	GO: 0003723; GO: 0003735	GO: 0005840

peptide	em_prot	em_go	em_gene	em_descript	blast_go	mg_go	uni_go_bp	uni_go_mf	uni_go_cc
DIAMQ[illegible]VPTISVIEHEK	479436.Vpar_0867	GO:0003674; GO:0003676; GO:0003723; GO:0003746; GO:0005488; GO:0005575; GO:0005618; GO:0005622; GO:0005623; GO:0005737; GO:0005829; GO:0005886; GO:0006412; GO:0006414; GO:0008135; GO:0008150; GO:0008152; GO:0008270; GO:0009058; GO:0009059; GO:0009987; GO:0010467; GO:0016020; GO:0019538; GO:0030312; GO:0034645; GO:0040007; GO:0043167; GO:0043169; GO:0043170; GO:0044237; GO:0044238; GO:0044249; GO:0044260; GO:0044267; GO:0044424; GO:0044444; GO:0044464; GO:0046872; GO:0046914; GO:0071704; GO:0071944; GO:0097159; GO:1901363; GO:1901576	tsf	Associates with the EF-Tu.GDP complex and induces the exchange of GDP to GTP. It remains bound to the aminoacyl-tRNA.EF-Tu.GTP complex up to the GTP hydrolysis stage on the ribosome (By similarity)	GO:0003746; GO:0005737	GO:0008152; GO:0043170; GO:0044464; GO:0003723; GO:0044267; GO:0043603; GO:0043604; GO:0006807; GO:0044271; GO:0003674; GO:0008135; GO:0005737; GO:0003676; GO:1901363; GO:0019538; GO:0003746; GO:0043043; GO:0006412; GO:1901564; GO:1901566; GO:0006414; GO:0044424; GO:0008150; GO:1901576; GO:0034641; GO:0034645; GO:0044237; GO:0044238; GO:0005622; GO:0009987; GO:0097159; GO:0044249; GO:0009058; GO:0005575; GO:0009059; GO:0006518; GO:0071704; GO:0005488; GO:0044260		GO: 0003746	GO: 0005737

peptide	em_prot	em_go	em_gene	em_descript	blast_go	mg_go	uni_go_bp	uni_go_mf	uni_go_cc
DLFKNPTPYTK	457405.FSDG_00110			ATP-binding protein	GO:0005524; GO:0015833; GO:0016887	GO:0016817; GO:0016818; GO:0044464; GO:0048869; GO:0051716; GO:0043934; GO:0009605; GO:1901363; GO:0006810; GO:0008104; GO:0048646; GO:0032549; GO:0032550; GO:0008150; GO:0032553; GO:0050896; GO:0032555; GO:0051179; GO:0032559; GO:0044763; GO:0071496; GO:0044765; GO:1901265; GO:0044767; GO:0036094; GO:1902578; GO:0005575; GO:0030554; GO:0005488; GO:0043167; GO:0045184; GO:0043168; GO:0007154; GO:0015833; GO:0042886; GO:0000166; GO:0097367; GO:0016020; GO:0017111; GO:0016462; GO:0051234; GO:0003674; GO:0005886; GO:0031668; GO:0030420; GO:0030154; GO:0005524; GO:0017076; GO:0016887; GO:0030435; GO:0016787; GO:0033036; GO:0044699; GO:0032502; GO:0015031; GO:0001882; GO:0003824; GO:0009987; GO:0035639;	GO: 0015833	GO: 0005524; GO: 0016887	

peptide	em_prot	em_go	em_gene	em_descript	blast_go	mg_go	uni_go_bp	uni_go_mf	uni_go_cc
DVTIEAVNSKYEK	479436.Vpar_0475		MOAB	molybdenum cofactor	GO:0001732; GO:0003743; GO:0005829; GO:0006777; GO:0016282; GO:0032436; GO:0033290; GO:0034613; GO:0070196; GO:0071540; GO:1900182	GO:0008152; GO:0043170; unknownfun; GO:0018130; unknowncmp; GO:0006732; GO:0051189; GO:1901360; GO:0006807; GO:0003674; GO:1901362; GO:0019538; GO:0043545; GO:1901564; GO:0090407; GO:1901566; GO:0046483; GO:0008150; GO:1901576; GO:0051186; GO:0051188; GO:0009108; GO:0044237; GO:0044238; GO:0006777; GO:0019637; GO:0009987; GO:0044249; GO:0009058; GO:0005575; GO:0071704; GO:0006793; GO:0019720; GO:0006796; GO:0032324	GO: 0006777		

peptide	em_prot	em_go	em_gene	em_descript	blast_go	mg_go	uni_go_bp	uni_go_mf	uni_go_cc
EVPDW[illegible]	[illegible]	GO:0006412; GO:0040007	[illegible]	Allows the formation of correctly charged Gln-tRNA(Gln) through the transamidation of misacylated Glu-tRNA(Gln) in organisms which lack glutaminyl-tRNA synthetase. The reaction takes place in the presence of glutamine and ATP through an activated gamma-phospho-Glu-tRNA(Gln) (By similarity)	GO:0004040; GO:0005524; GO:0006412; GO:0016740; GO:0030956; GO:0050567	GO:0044464; GO:0006807; GO:1901363; GO:0006412; GO:0032549; GO:0032550; GO:0008150; GO:0032553; GO:0032555; GO:0034641; GO:0032559; GO:0034645; GO:0044237; GO:0044238; GO:1901265; GO:0036094; GO:0030956; GO:0044249; GO:0005575; GO:0030554; GO:0016874; GO:0004040; GO:0032991; GO:0005488; GO:0043167; GO:0016879; GO:0044260; GO:0043168; GO:0000166; GO:0008152; GO:0043170; GO:0097367; GO:0016884; GO:0044267; GO:0043603; GO:0043604; GO:0044271; GO:0003674; GO:0019538; GO:0043043; GO:1901564; GO:1901566; GO:0005524; GO:0017076; GO:0044424; GO:1901576; GO:0016787; GO:0009987; GO:0001882; GO:0003824; GO:0035639; GO:0001883; GO:0097159; GO:0009058; GO:0009059; GO:0006518; GO:0071704; GO:0050567; GO:0016810;	GO: 0006412	GO: 0004040; GO: 0005524; GO: 0050567	GO: 0030956

peptide	em_prot	em_go	em_gene	em_descript	blast_go	mg_go	uni_go_bp	uni_go_mf	uni_go_cc

Let’s go through the peptides one-by-one

AFLPGSLVDTRPVR

Here, BLAST and Unipept give the same four GO terms:

```r
buni <- all_results[1, 'blast_go']
buni
```

```
## [1] "GO:0003723; GO:0003735; GO:0005840; GO:0006412"
```

Let’s go to the QuickGO API to get the names of these 4.

```r
library(httr)
library(jsonlite)
buni_split <- str_split(buni, "; ", simplify = TRUE)
get_go_names <- function(id_vector){
    base_url <- 'https://www.ebi.ac.uk/QuickGO/services/ontology/go/terms/'
    terms <- str_replace(id_vector, ":", "%3A")
    joined_terms <- paste(terms, collapse="%2C")
    term_url <- paste(base_url, joined_terms, sep="")
    term_info <- GET(term_url, accept("application/json"))
    json <- toJSON(content(term_info))
    names <- unlist(fromJSON(json)$results$name)
    names
}
get_go_names(buni_split)
```

```
## [1] "translation"
## [2] "ribosome"
## [3] "RNA binding"
## [4] "structural constituent of ribosome"
```

On the other hand, both eggNOG mapper and metaGOmics give huge lists of GO terms. Are the 4 BLAST+Unipept terms contained in these lists?

Does eggnog mapper contain all 4?

```r
all(str_detect(all_results[1, ]$em_go, pattern = buni_split))
```

```
## [1] TRUE
```

Does metagomics contain all 4?

```r
all(str_detect(all_results[1, ]$mg_gos, pattern = buni_split))
```

```
## [1] TRUE
```

The question is, then, what the other terms are.

We can look at this in 3 ways:

1) Are the extra terms more general (ancestors) of the 4 we found?
2) Are there any terms which are more specific (children) of the 4 we found?
3) Are there terms that are not ancestors or children? These may be false hits.

Let’s look at question 1):

```r
# get all ancestors of the 4 terms
get_go_ancestors <- function(id_vector){
    base_url <- 'https://www.ebi.ac.uk/QuickGO/services/ontology/go/terms/'
    terms <- str_replace(id_vector, ":", "%3A")
    joined_terms <- paste(terms, collapse="%2C")
    term_url <- paste(base_url, joined_terms, '/ancestors?relations=is_a', sep="")
    term_info <- GET(term_url, accept("application/json"))
    json <- toJSON(content(term_info))
    names <- unlist(fromJSON(json)$results$ancestors)
    names
}
ancestors <- get_go_ancestors(buni_split)
get_go_names(ancestors)
```

```
##  [1] "molecular_function"
##  [2] "amide biosynthetic process"
##  [3] "cellular amide metabolic process"
##  [4] "biological_process"
##  [5] "metabolic process"
##  [6] "organonitrogen compound metabolic process"
##  [7] "organonitrogen compound biosynthetic process"
##  [8] "organic substance biosynthetic process"
##  [9] "heterocyclic compound binding"
## [10] "translation"
## [11] "nitrogen compound metabolic process"
## [12] "peptide metabolic process"
## [13] "ribosome"
## [14] "cellular_component"
## [15] "macromolecule metabolic process"
## [16] "intracellular organelle"
## [17] "organelle"
## [18] "non-membrane-bounded organelle"
## [19] "intracellular non-membrane-bounded organelle"
## [20] "nucleic acid binding"
## [21] "structural molecule activity"
## [22] "peptide biosynthetic process"
## [23] "macromolecule biosynthetic process"
## [24] "biosynthetic process"
## [25] "cellular nitrogen compound metabolic process"
## [26] "cellular macromolecule biosynthetic process"
## [27] "RNA binding"
## [28] "structural constituent of ribosome"
## [29] "organic cyclic compound binding"
## [30] "ribonucleoprotein complex"
## [31] "protein-containing complex"
## [32] "binding"
## [33] "organic substance metabolic process"
## [34] "protein metabolic process"
## [35] "cytoplasmic part"
## [36] "cell part"
## [37] "intracellular part"
## [38] "cellular process"
## [39] "primary metabolic process"
## [40] "cellular metabolic process"
## [41] "cellular biosynthetic process"
## [42] "cellular macromolecule metabolic process"
## [43] "cellular protein metabolic process"
## [44] "cellular nitrogen compound biosynthetic process"
```

Now, let's look at the overlap between the ancestors (including the original 4 terms) and eggnog mapper, and between the ancestors and metaGOmics:

```
four_and_ancestors <- c(ancestors, buni_split)
```

```
metaGOmics_list <- c(str_split(all_results[1, 'mg_go'], "; ", simplify=TRUE))
overlap <- length(intersect(four_and_ancestors, metaGOmics_list))
overlap
```

```
## [1] 44
```

```
eggnog_list <- c(str_split(all_results[1, 'em_go'], "; ", simplify =TRUE))
overlap <- length(intersect(four_and_ancestors, eggnog_list))
overlap
```

```
## [1] 34
```

Let's do the same for the children:

```
# get all childen of the 4 terms
get_go_descendants <- function(id_vector){
    base_url <- 'https://www.ebi.ac.uk/QuickGO/services/ontology/go/terms/'
    terms <- str_replace(id_vector, ":", "%3A")
    joined_terms <- paste(terms, collapse="%2C")
    term_url <- paste(base_url, joined_terms, '/descendants?relations=is_a', sep="")
    term_info <- GET(term_url, accept("application/json"))
    json <- toJSON(content(term_info))
    names <- unlist(fromJSON(json)$results$descendants)
    names
}
descendants <- get_go_descendants(buni_split)
get_go_names(descendants)
```

```
##   [1] "7S RNA binding"
##   [2] "poly(U) RNA binding"
##   [3] "ribonuclease P RNA binding"
##   [4] "CUA codon-amino acid adaptor activity"
##   [5] "CUC codon-amino acid adaptor activity"
##   [6] "CUU codon-amino acid adaptor activity"
##   [7] "UAG codon-amino acid adaptor activity"
##   [8] "UAA codon-amino acid adaptor activity"
##   [9] "UAC codon-amino acid adaptor activity"
##  [10] "UGG codon-amino acid adaptor activity"
##  [11] "UGA codon-amino acid adaptor activity"
##  [12] "UGC codon-amino acid adaptor activity"
##  [13] "UGU codon-amino acid adaptor activity"
##  [14] "UAU codon-amino acid adaptor activity"
##  [15] "UCG codon-amino acid adaptor activity"
##  [16] "UCA codon-amino acid adaptor activity"
##  [17] "UCC codon-amino acid adaptor activity"
##  [18] "UUU codon-amino acid adaptor activity"
##  [19] "UCU codon-amino acid adaptor activity"
## [20] "UUG codon-amino acid adaptor activity"
## [21] "UUA codon-amino acid adaptor activity"
## [22] "UUC codon-amino acid adaptor activity"
## [23] "GGA codon-amino acid adaptor activity"
## [24] "GGC codon-amino acid adaptor activity"
## [25] "GGU codon-amino acid adaptor activity"
## [26] "GAG codon-amino acid adaptor activity"
## [27] "GCG codon-amino acid adaptor activity"
## [28] "GCA codon-amino acid adaptor activity"
## [29] "GCC codon-amino acid adaptor activity"
## [30] "GCU codon-amino acid adaptor activity"
## [31] "GAA codon-amino acid adaptor activity"
## [32] "GAC codon-amino acid adaptor activity"
## [33] "GAU codon-amino acid adaptor activity"
## [34] "GUG codon-amino acid adaptor activity"
## [35] "GUA codon-amino acid adaptor activity"
## [36] "GUC codon-amino acid adaptor activity"
## [37] "AGU codon-amino acid adaptor activity"
## [38] "AAG codon-amino acid adaptor activity"
## [39] "AAA codon-amino acid adaptor activity"
## [40] "AAC codon-amino acid adaptor activity"
## [41] "GUU codon-amino acid adaptor activity"
## [42] "AGG codon-amino acid adaptor activity"
## [43] "AGA codon-amino acid adaptor activity"
## [44] "AGC codon-amino acid adaptor activity"
## [45] "AAU codon-amino acid adaptor activity"
## [46] "ACG codon-amino acid adaptor activity"
## [47] "ACA codon-amino acid adaptor activity"
## [48] "AUC codon-amino acid adaptor activity"
## [49] "AUU codon-amino acid adaptor activity"
## [50] "CGG codon-amino acid adaptor activity"
## [51] "CGA codon-amino acid adaptor activity"
## [52] "ACC codon-amino acid adaptor activity"
## [53] "ACU codon-amino acid adaptor activity"
## [54] "AUG codon-amino acid adaptor activity"
## [55] "AUA codon-amino acid adaptor activity"
## [56] "CGC codon-amino acid adaptor activity"
## [57] "CGU codon-amino acid adaptor activity"
## [58] "CAG codon-amino acid adaptor activity"
## [59] "CCA codon-amino acid adaptor activity"
## [60] "CCC codon-amino acid adaptor activity"
## [61] "CCU codon-amino acid adaptor activity"
## [62] "CUG codon-amino acid adaptor activity"
## [63] "CAA codon-amino acid adaptor activity"
## [64] "CAC codon-amino acid adaptor activity"
## [65] "CAU codon-amino acid adaptor activity"
## [66] "CCG codon-amino acid adaptor activity"
## [67] "GGG codon-amino acid adaptor activity"
## [68] "large ribosomal subunit rRNA binding"
## [69] "small ribosomal subunit rRNA binding"
## [70] "poly(A) binding"
## [71] "translation factor activity, RNA binding"
## [72] "poly-pyrimidine tract binding"
## [73] "pre-mRNA branch point binding"
##  [74] "G-quadruplex RNA binding"
##  [75] "steroid receptor RNA activator RNA binding"
##  [76] "cytoplasmic translation"
##  [77] "poly-purine tract binding"
##  [78] "mRNA 3'-UTR AU-rich region binding"
##  [79] "telomerase RNA binding"
##  [80] "RNA stem-loop binding"
##  [81] "snRNA stem-loop binding"
##  [82] "translation termination factor activity"
##  [83] "5S rRNA primary transcript binding"
##  [84] "5S rRNA binding"
##  [85] "selenocysteine insertion sequence binding"
##  [86] "translation release factor activity, codon nonspecific"
##  [87] "translation release factor activity, codon specific"
##  [88] "translation"
##  [89] "mRNA binding involved in posttranscriptional gene silencing"
##  [90] "pre-mRNA binding"
##  [91] "mRNA 5'-UTR binding"
##  [92] "tRNA binding"
##  [93] "miRNA binding"
##  [94] "siRNA binding"
##  [95] "double-stranded miRNA binding"
##  [96] "mRNA cap binding"
##  [97] "mitochondrial ribosome"
##  [98] "polysomal ribosome"
##  [99] "BRE binding"
## [100] "ribosome"
## [101] "triplet codon-amino acid adaptor activity"
## [102] "snoRNA binding"
## [103] "RNA 2'-O-ribose methylation guide activity"
## [104] "tRNA pseudouridylation guide activity"
## [105] "snRNA 2'-O-ribose methylation guide activity"
## [106] "rRNA 2'-O-ribose methylation guide activity"
## [107] "snRNA pseudouridylation guide activity"
## [108] "tRNA 2'-O-ribose methylation guide activity"
## [109] "snRNA modification guide activity"
## [110] "RNA pseudouridylation guide activity"
## [111] "tRNA modification guide activity"
## [112] "rRNA pseudouridylation guide activity"
## [113] "rRNA modification guide activity"
## [114] "RNA modification guide activity"
## [115] "U6atac snRNA binding"
## [116] "U5 snRNA binding"
## [117] "U12 snRNA binding"
## [118] "U11 snRNA binding"
## [119] "pre-mRNA 3'-splice site binding"
## [120] "pre-mRNA 5'-splice site binding"
## [121] "U6 snRNA 3'-end binding"
## [122] "U2 snRNA binding"
## [123] "U4atac snRNA binding"
## [124] "U4 snRNA binding"
## [125] "U1 snRNA binding"
## [126] "chloroplast ribosome"
## [127] "N6-methyladenosine-containing RNA binding"
## [128] "regulatory region RNA binding"
## [129] "transcription regulatory region RNA binding"
## [130] "pre-miRNA binding"
## [131] "primary miRNA binding"
## [132] "RNA strand annealing activity"
## [133] "GU repeat RNA binding"
## [134] "alpha-aminoacyl-tRNA binding"
## [135] "snRNA binding"
## [136] "AU-rich element binding"
## [137] "U6 snRNA binding"
## [138] "rRNA binding"
## [139] "rRNA primary transcript binding"
## [140] "base pairing with mRNA"
## [141] "base pairing with RNA"
## [142] "cytosolic ribosome"
## [143] "7SK snRNA binding"
## [144] "RNA cap binding"
## [145] "RNA cap 4 binding"
## [146] "RNA trimethylguanosine cap binding"
## [147] "RNA 7-methylguanosine cap binding"
## [148] "organellar ribosome"
## [149] "mRNA CDS binding"
## [150] "histone pre-mRNA DCP binding"
## [151] "U7 snRNA binding"
## [152] "histone pre-mRNA stem-loop binding"
## [153] "misfolded RNA binding"
## [154] "regulatory RNA binding"
## [155] "RNA strand-exchange activity"
## [156] "poly(G) binding"
## [157] "mitochondrial ribosomal large subunit rRNA binding"
## [158] "mRNA 3'-UTR binding"
## [159] "RNA binding"
## [160] "double-stranded RNA binding"
## [161] "single-stranded RNA binding"
## [162] "mRNA binding"
## [163] "structural constituent of ribosome"
## [164] "translation initiation factor activity"
## [165] "translation elongation factor activity"
## [166] "translation release factor activity"
## [167] "pre-mRNA intronic binding"
## [168] "pre-mRNA intronic pyrimidine-rich binding"
## [169] "sequence-specific mRNA binding"
## [170] "methionyl-initiator methionine tRNA binding"
## [171] "5.8S rRNA binding"
## [172] "trans-activation response element binding"
## [173] "iron-responsive element binding"
## [174] "poly(C) RNA binding"
## [175] "uridine-rich cytoplasmic polyadenylylation element binding"
## [176] "base pairing with tRNA"
## [177] "base pairing with snRNA"
## [178] "base pairing with rRNA"
## [179] "telomeric repeat-containing RNA binding"
## [180] "plastid ribosome"
## [181] "box C/D snoRNA binding"
## [182] "U3 snoRNA binding"
## [183] "box H/ACA snoRNA binding"
## [184] "piRNA binding"
## [185] "21U-RNA binding"
## [186] "translation at presynapse"
## [187] "translation at presynapse, modulating chemical synaptic transmission"
## [188] "translation at synapse"
## [189] "translation at postsynapse"
## [190] "mitochondrial translation"
## [191] "plastid translation"
```

```
four_and_descendants <- c(descendants, buni_split)
```

MetaGOmics overlap:

```
length(intersect(four_and_descendants, metaGOmics_list))
```

```
## [1] 5
```

eggNOG overlap with descendants:

```
eggnog_list <- c(str_split(all_results[1, 'em_go'], "; ", simplify =TRUE))
length(intersect(four_and_descendants, eggnog_list))
```

```
## [1] 6
```

Ok, so still not a lot of overlap. Let's examine the terms that are neither the 4 terms or their descendants or ancestors:

```
full_tree <- c(buni_split, ancestors, descendants)
```

metaGOmics

```
diff <- setdiff(metaGOmics_list, full_tree)
get_go_names(diff)
```

```
##  [1] "cellular component organization"
##  [2] "membrane"
##  [3] "biological regulation"
##  [4] "protein-containing complex assembly"
##  [5] "regulation of translation"
##  [6] "negative regulation of cellular biosynthetic process"
##  [7] "regulation of cellular biosynthetic process"
##  [8] "regulation of cellular metabolic process"
##  [9] "negative regulation of cellular metabolic process"
## [10] "regulation of cellular process"
## [11] "negative regulation of cellular macromolecule biosynthetic process"
## [12] "regulation of cellular macromolecule biosynthetic process"
## [13] "protein-containing complex subunit organization"
## [14] "regulation of biological process"
## [15] "ribosomal small subunit assembly"
## [16] "regulation of protein metabolic process"
## [17] "negative regulation of protein metabolic process"
## [18] "negative regulation of nitrogen compound metabolic process"
## [19] "regulation of nitrogen compound metabolic process"
## [20] "regulation of macromolecule metabolic process"
## [21] "cytoplasm"
## [22] "regulation of primary metabolic process"
## [23] "small ribosomal subunit"
## [24] "regulation of cytoplasmic translation"
## [25] "negative regulation of cytoplasmic translation"
## [26] "cellular protein-containing complex assembly"
## [27] "cytosolic small ribosomal subunit"
## [28] "ribonucleoprotein complex assembly"
## [29] "RNA binding"
## [30] "negative regulation of biological process"
## [31] "negative regulation of cellular process"
## [32] "negative regulation of translation"
## [33] "regulation of cellular amide metabolic process"
## [34] "negative regulation of cellular amide metabolic process"
## [35] "regulation of biosynthetic process"
## [36] "regulation of gene expression"
## [37] "cellular component assembly"
## [38] "negative regulation of gene expression"
## [39] "posttranscriptional regulation of gene expression"
## [40] "negative regulation of macromolecule metabolic process"
## [41] "regulation of macromolecule biosynthetic process"
## [42] "negative regulation of macromolecule biosynthetic process"
## [43] "cytosolic part"
## [44] "intracellular organelle part"
## [45] "organelle part"
## [46] "negative regulation of metabolic process"
## [47] "negative regulation of biosynthetic process"
## [48] "regulation of metabolic process"
## [49] "ribonucleoprotein complex subunit organization"
## [50] "regulation of cellular protein metabolic process"
## [51] "negative regulation of cellular protein metabolic process"
## [52] "cellular component organization or biogenesis"
## [53] "ribosomal subunit"
## [54] "ribonucleoprotein complex"
```

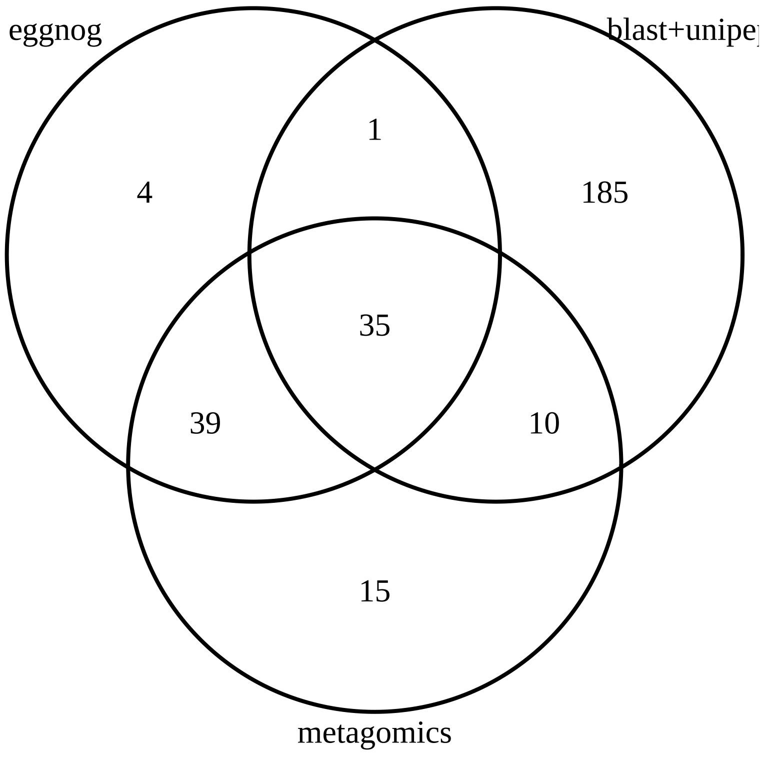
eggNOG mapper

```r
em_diff <- setdiff(eggnog_list, full_tree)
get_go_names(em_diff)
```

```
##  [1] "membrane"
##  [2] "biological regulation"
##  [3] "regulation of translation"
##  [4] "negative regulation of cellular biosynthetic process"
##  [5] "regulation of cellular biosynthetic process"
##  [6] "regulation of cellular metabolic process"
##  [7] "negative regulation of cellular metabolic process"
##  [8] "regulation of cellular process"
##  [9] "negative regulation of cellular macromolecule biosynthetic process"
## [10] "regulation of cellular macromolecule biosynthetic process"
## [11] "regulation of biological process"
## [12] "regulation of protein metabolic process"
## [13] "negative regulation of protein metabolic process"
## [14] "regulation of macromolecule metabolic process"
## [15] "cytoplasm"
## [16] "cytosol"
## [17] "cell"
## [18] "intracellular"
## [19] "regulation of primary metabolic process"
## [20] "small ribosomal subunit"
## [21] "regulation of cytoplasmic translation"
## [22] "negative regulation of cytoplasmic translation"
## [23] "cytosolic small ribosomal subunit"
## [24] "negative regulation of biological process"
## [25] "negative regulation of cellular process"
## [26] "negative regulation of translation"
## [27] "regulation of biosynthetic process"
## [28] "regulation of gene expression"
## [29] "gene expression"
## [30] "posttranscriptional regulation of gene expression"
## [31] "negative regulation of macromolecule metabolic process"
## [32] "regulation of macromolecule biosynthetic process"
## [33] "negative regulation of macromolecule biosynthetic process"
## [34] "cytosolic part"
## [35] "intracellular organelle part"
## [36] "organelle part"
## [37] "negative regulation of metabolic process"
## [38] "negative regulation of biosynthetic process"
## [39] "regulation of metabolic process"
## [40] "regulation of cellular protein metabolic process"
## [41] "negative regulation of cellular protein metabolic process"
## [42] "ribosomal subunit"
## [43] "ribonucleoprotein complex"
```

Visualize the overlap between the full Blast+Unipept tree (descendants, ancestors) and the eggNOG and metaGOmics term lists.

```r
library(VennDiagram)
grid.newpage()
grid.draw(venn.diagram(
    list("eggnog" = eggnog_list, "blast+unipept" = full_tree, "metagomics" = metaGOmics_list),
    NULL))
```



```r
file.remove(list.files(pattern = "VennDiagram.*log")) # venn diagram log files
```

```
## [1] TRUE
```

Future directions

1) repeat this for the other 4 peptides
2) how do we handle terms that are not descendants or ancestors? We could define some distance cutoff, and say that everything beyond that is a false hit. For example, we could say that if the shortest path between a metaGOmics or eggNOG term and any term in the full B+U tree has length greater than or equal to 2 than it is a false hit.

Distance

Idea: if a term is not a child of any term in the full expanded graph of the Uniprot-assigned terms, it is a false hit.

```r
get_children <- function(goids){
    base_url <- 'https://www.ebi.ac.uk/QuickGO/services/ontology/go/terms/'
    terms <- str_replace(goids, ":", "%3A")
    joined_terms <- paste(terms, collapse="%2C")
    term_url <- paste(base_url, joined_terms, sep="")
    term_info <- GET(term_url, accept("application/json"))
    json <- toJSON(content(term_info))
    children <- fromJSON(json)$results$children
    children_is_a <- lapply(children, function(x) unlist(x[x$relation == "is_a", ]$id))
    return(children_is_a)
}
```

Let's do this for the full tree above.

```
full_tree_with_children <- get_children(full_tree)

# combine term-specific sets of children
all_children <- c(full_tree, unlist(full_tree_with_children))

# how many terms are not in the full tree with children?
em_diff_with_kids <- setdiff(eggnog_list, all_children)
mg_diff_with_kids <- setdiff(metaGOmics_list, all_children)
```

Calculate proportions to answer three questions: 1) How many of Uniprot's terms does the tool pick up? 2) What is the proportion of total terms from the tool that are extraneous?

Eggnog

```
# answer to 1
length(intersect(eggnog_list, buni_split)) / length(buni_split)
```

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

```
# answer to 2
length(em_diff_with_kids)/length(eggnog_list)
```

```
## [1] 0.4050633
```

MetaGOmics

```
# answer to 1
length(intersect(metaGOmics_list, buni_split)) / length(buni_split)
```

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

```
# answer to 2
length(mg_diff_with_kids)/length(metaGOmics_list)
```

```
## [1] 0.4545455
```

GO glossary

Here, I get the names of all the above GO terms.

```
library(httr)
library(jsonlite)
base_url <- 'https://www.ebi.ac.uk/QuickGO/services/ontology/go/terms/'
term_url <- paste(base_url, 'GO%3A0008150%2CGO%3A0008152', sep="")
term_info <- GET(term_url, verbose(), accept("application/json"))
json <- toJSON(content(term_info))
df <- fromJSON(json)$results
```