# Expanding GO annotations to include ancestors

Gene ontology (GO) terms are heirachical: http://geneontology.org/docs/ontology-relations/

For this reason, when analysing the over-represented GO terms, it's neccessary to consider all the GO terms for a feature, not just those which are directly annotated to the feature.

To date, all GO over-representation tools I've used fail to consider the ancestor terms unless you provide them. As a motivating example, tRNA binding proteins are not always annotated separately as both "tRNA-binding" and "RNA-binding". For Uniprot, they will only have both annotations if they are annotated as RNA binding via a difference source than their tRNA binding annotation.

In this notebook, we take a dataframe with GO annotations and expand all GO terms to include the ancestors too using functions from our `./GO.R` script. For some species this may take a long time - It takes ~ 20 minutes for the ~20,000 *H.sapiens* proteins. Hence output is saved in the `../results` directory.

```
suppressMessages(library(dplyr))
suppressMessages(library(tidyverse))
suppressMessages(library(biobroom))
suppressMessages(library(UniProt.ws))
source("./GO.R")
```

4.1 Question: What does the source function do? Why would we want to keep this code separate from the notebook.

This file lists all the human Swiss-Prot proteins. We could also parse this information from the fasta database we searched against.

```
human_protein_ids <- read.delim("../raw/human_protein_ids_plus_gene_names.tsv")
print(head(human_protein_ids))
```

```
##    Entry  Entry.name
## 1 P04217  A1BG_HUMAN
## 2 Q9NQ94  A1CF_HUMAN
## 3 P01023  A2MG_HUMAN
## 4 A8K2U0 A2ML1_HUMAN
## 5 U3KPV4 A3LT2_HUMAN
## 6 Q9NPC4 A4GAT_HUMAN
##
## 1
## 2
## 3
## 4
## 5
## 6 Lactosylceramide 4-alpha-galactosyltransferase (EC 2.4.1.228) (Alpha-1,4-N-acetylglucosaminyltransf
##                 Gene.names
## 1                     A1BG
## 2            A1CF ACF ASP
## 3       A2M CPAMD5 FWP007
## 4            A2ML1 CPAMD9
## 5 A3GALT2 A3GALT2P IGBS3S
## 6  A4GALT A14GALT A4GALT1
```

Get the Uniprot interface object and save

```
humanup <- UniProt.ws(taxId=9606) # H.sapiens
saveRDS(humanup, '../results/h_sapiens_uniprot_annotations.rds')
```

Load the interface object

```
humanup <- readRDS('../results/h_sapiens_uniprot_annotations.rds')
print(humanup)
```

```
## "UniProt.ws" object:
## An interface object for UniProt web services
## Current Taxonomy ID:
## 9606
## Current Species name:
## Homo sapiens
## To change Species see: help('availableUniprotSpecies')
```

Get all GO terms for the proteins of interest

```
sapiens.annot <- AnnotationDbi::select(
  humanup,
  keys = human_protein_ids$Entry,
  columns = c("GO-ID", "INTERPRO", "PROTEIN-NAMES"),
  keystyle = "UNIPROTKB")

saveRDS(sapiens.annot, '../results/h_sapiens_annotations.rds')
```

Reformat the dataframe so that each row contains a single GO ID for a single protein

```
sapiens.annot <- readRDS('../results/h_sapiens_annotations.rds')

sapiens.go <-  sapiens.annot %>%
  data.frame() %>%
  separate_rows(GO.ID, sep="; ") %>%
  dplyr::select(UNIPROTKB, PROTEIN.NAMES, GO.ID)
```

For each "tRNA binding" protein, indicate whether it is also directly annotated as "RNA binding". Note they they are all indirectly annotated since tRNA binding is a child term of RNA binding: https://www.ebi.ac.uk/QuickGO/term/GO:0000049

```
RBP_GO_TERM <- "GO:0003723"
TRNA_BINDING_GO_TERM <- "GO:0000049"

RBPs <- sapiens.go %>% filter (GO.ID==RBP_GO_TERM) %>% pull(UNIPROTKB)
TRNA_BPs <- sapiens.go %>% filter (GO.ID==TRNA_BINDING_GO_TERM) %>% pull(UNIPROTKB)

print(sapply(TRNA_BPs, FUN=function(x) x %in% RBPs))
```

```
## P49588 Q5JTZ9 Q12904 Q13686 Q96BT7 P49589 Q7Z7A3 Q2VPK5 Q6PI48 Q8TEA8 Q96FN9
##  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE
## Q5JPH6 P68104 P57772 Q9BY44 Q9P2K8 O95363 Q9Y285 Q99714 P41252 Q9NSE4 Q13325
##  FALSE   TRUE  FALSE  FALSE  FALSE  FALSE   TRUE   TRUE  FALSE  FALSE   TRUE
## P38935 Q15046 Q9P272 P56192 Q9UBP6 Q92552 O60524 Q08J23 Q9H649 Q8TEA1 O75127
##   TRUE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE   TRUE  FALSE  FALSE   TRUE
## Q9Y606 P54136 P18077 Q9HD40 Q7Z7L1 Q68D06 P05455 P26639 O14746 Q9NWX6 Q9NXH9
##   TRUE  FALSE   TRUE  FALSE  FALSE  FALSE   TRUE  FALSE   TRUE  FALSE   TRUE
## Q8TBZ6 Q6PF06 Q7L0Y3 Q7Z4G4 Q7Z2T5 O75648 Q96Q11 A2RUC4 Q9HAV4 O43592 P54577
##   TRUE  FALSE   TRUE  FALSE   TRUE  FALSE  FALSE  FALSE   TRUE  FALSE   TRUE
```

```
## Q9Y2Z4 Q86U90
##   TRUE  FALSE
```

4.2 Question: Why wouldn't every protein/gene be directly annotated with all terms up the hierarchy?

Below we demonstrate how the `getAllGOTerms()` function expands the directly annotated terms (only 2) to all the ancester terms (176 in total!) for a single protein

```
go.single <- sapiens.go %>% filter(UNIPROTKB=='Q86V81')
sapiens.go.single <- getAllGOTerms(go.single, verbose=FALSE)
```

```
##   UNIPROTKB
## 1    Q86V81
## 2    Q86V81
##
## 1 THO complex subunit 4 (Tho4) (Ally of AML-1 and LEF-1) (Aly/REF export factor) (Transcriptional co
## 2 THO complex subunit 4 (Tho4) (Ally of AML-1 and LEF-1) (Aly/REF export factor) (Transcriptional co
##        GO.ID
## 1 GO:0000018
## 2 GO:0000346
## [1] "GO:0000018" "GO:0000346"
```

```
## 'select()' returned 1:1 mapping between keys and columns
```

```
## [1] "Expanding GO terms to include all ancestors for all entries"
```

```
## Warning: filter_() is deprecated.
## Please use filter() instead
##
## The 'programming' vignette or the tidyeval book can help you
## to program with filter() : https://tidyeval.tidyverse.org
## This warning is displayed once per session.
```

```
## Warning: group_by_() is deprecated.
## Please use group_by() instead
##
## The 'programming' vignette or the tidyeval book can help you
## to program with group_by() : https://tidyeval.tidyverse.org
## This warning is displayed once per session.
```

```
## 'select()' returned 1:1 mapping between keys and columns
```

```
print(dim(sapiens.go.single))
```

```
## [1] 176   4
```

Then we apply this function to our full set of GO terms across all proteins of interest and save out for use in later notebooks

```
sapiens.go.full <- getAllGOTerms(sapiens.go, verbose=FALSE)
saveRDS(sapiens.go.full, "../results/h_sapiens_go_full.rds")
```

Note all tRNA binding proteins are now also directly annotated as RNA binding proteins too

```
sapiens.go.full <- readRDS("../results/h_sapiens_go_full.rds")

RBPs <- sapiens.go.full %>% filter (GO.ID==RBP_GO_TERM) %>% pull(UNIPROTKB)
TRNA_BPs <- sapiens.go.full %>% filter (GO.ID==TRNA_BINDING_GO_TERM) %>% pull(UNIPROTKB)
```

```r
print(table(sapply(TRNA_BPs, FUN=function(x) x %in% RBPs)))
```

```
##
## TRUE
##   57
```