COG Analysis

Tania Kurbessoian

6/9/2021

```
library(ape)
library(patchwork)
library(ggplot2)
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v tibble  3.1.0     v stringr 1.4.0
## v tidyr   1.1.2     v forcats 0.5.0
## v purrr   0.3.4

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(RColorBrewer)
library(magrittr)
```

```
##
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':
##
##     set_names
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
library(reshape)
```

```
##
## Attaching package: 'reshape'

## The following objects are masked from 'package:tidyr':
##
##     expand, smiths

## The following object is masked from 'package:dplyr':
##
##     rename
```

```
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following object is masked from 'package:reshape':
##
##     melt

## The following object is masked from 'package:purrr':
##
##     transpose

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
library(stringr)
library(viridis)
```

```
## Loading required package: viridisLite
```

```
library(cowplot)
```

```
##
## Attaching package: 'cowplot'

## The following object is masked from 'package:reshape':
##
##     stamp

## The following object is masked from 'package:patchwork':
##
##     align_plots
```

```
pdf("All_COG_F_endo.pdf", width=12, height=7)
#data_endo <- read_tsv("endolithicus_CCFEE_5311.annotations.txt")
#cog.only_all <- data_endo[c(1,16)]
#write.table(cog.only_all, file='cog.only_all.tsv', quote=FALSE, sep='\t')
cog.only_all_cleaned <- read_tsv("cog.only_all_cleaned.txt")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   GeneID = col_character(),
##   COG = col_character()
## )
```

```
#split.cog_all <- cbind(cog.only, fread(text = cog.only$COG, sep = ":", header = FALSE))
#split.cog_more <- cbind(split.cog, fread(text = split.cog$COG, sep = ";", header = FALSE))
#colnames(split.cog_more) = c("GeneID", "COG", "Split1", "Split2", "Split3")
#cog_cleaned.only <-split.cog[c(1,3)]
cog.counts.all <- dplyr::count(cog.only_all_cleaned, COG)
colnames(cog.counts.all) = c("COG", "Counts")
removed_na <- na.omit(cog.counts.all)
#plot1 <- ggplot(data = cog.counts.all) +
#  geom_point(mapping = aes(x = COG, y = Counts))

#Thanks to this website - https://www.r-bloggers.com/2013/09/how-to-expand-color-palette-with-ggplot-an
xticks <- c( "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "S",

colourCount = length(unique(removed_na$COG))
getPalette = colorRampPalette(brewer.pal(9, "Dark2"))
```

```
## Warning in brewer.pal(9, "Dark2"): n too large, allowed maximum for palette Dark2 is 8
## Returning the palette you asked for with that many colors
```

```
p1 <- ggplot(data=removed_na, aes(x=COG, y=Counts, fill=COG)) + geom_bar(stat="identity") + geom_text(a
scale_fill_manual(values = getPalette(colourCount)) +  theme_bw()

#geom_bar(stat="identity") + guides(fill = guide_legend(ncol = 1)) + xlab("COG Class")  +ggtitle("COG R
#scale_fill_manual(values = getPalette(colourCount)) +  theme_bw()
p1
dev.off()
```

```
## pdf
##   2
```

```
pdf("All_COG_F_simplex.pdf", width=12, height=7)
#data_simp <- read_tsv("simplex_CCFEE_5184.annotations.txt")
#simp.cog.only_all <- data_simp[c(1,16)]
#write.table(simp.cog.only_all, file='simp.cog.only_all.tsv', quote=FALSE, sep='\t')
simp.cog.only_all_cleaned <- read_tsv("simp.cog.only_all_cleaned.txt")
```

```
##
## -- Column specification --------------------------------------------------------
```

```
## cols(
##   GeneID = col_character(),
##   COG = col_character()
## )
```

```
#split.cog_all <- cbind(cog.only, fread(text = cog.only$COG, sep = ":", header = FALSE))
#split.cog_more <- cbind(split.cog, fread(text = split.cog$COG, sep = ";", header = FALSE))
#colnames(split.cog_more) = c("GeneID", "COG", "Split1", "Split2", "Split3")
#cog_cleaned.only <-split.cog[c(1,3)]
simp.cog.counts.all <- dplyr::count(simp.cog.only_all_cleaned, COG)
colnames(simp.cog.counts.all) = c("COG", "Counts")
simp.removed_na <- na.omit(simp.cog.counts.all)
#plot1 <- ggplot(data = cog.counts.all) +
#  geom_point(mapping = aes(x = COG, y = Counts))

#Thanks to this website - https://www.r-bloggers.com/2013/09/how-to-expand-color-palette-with-ggplot-an

xticks <- c( "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "S",
colourCount = length(unique(simp.removed_na$COG))
getPalette = colorRampPalette(brewer.pal(9, "Dark2"))
```

```
## Warning in brewer.pal(9, "Dark2"): n too large, allowed maximum for palette Dark2 is 8
## Returning the palette you asked for with that many colors
```

```
p2 <- ggplot(data=simp.removed_na, aes(x=COG, y=Counts, fill=COG)) + geom_bar(stat="identity") + geom_t
scale_fill_manual(values = getPalette(colourCount)) +  theme_bw()


p2
dev.off()
```

```
## pdf
##   2
```

```
pdf("All_COG_H_wer.pdf", width=12, height=7)
#data_wer <- read_tsv("werneckii_EXF-2000.annotations.txt")
#cog.only_all_wer <- data_wer[c(1,16)]
#write.table(cog.only_all_wer, file='Wer_cog.only_all.tsv', quote=FALSE, sep='\t')
Wer_cog.only_all_cleaned <- read_tsv("Wer_cog.only_all_cleaned.txt")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   GeneID = col_character(),
##   COG = col_character()
## )
```

```
#split.cog_all <- cbind(cog.only, fread(text = cog.only$COG, sep = ":", header = FALSE))
#split.cog_more <- cbind(split.cog, fread(text = split.cog$COG, sep = ";", header = FALSE))
#colnames(split.cog_more) = c("GeneID", "COG", "Split1", "Split2", "Split3")
#cog_cleaned.only <-split.cog[c(1,3)]
```

```
Wer_cog.counts.all <- dplyr::count(Wer_cog.only_all_cleaned, COG)
colnames(Wer_cog.counts.all) = c("COG", "Counts")
Wer_removed_na <- na.omit(Wer_cog.counts.all)
#plot1 <- ggplot(data = cog.counts.all) +
#  geom_point(mapping = aes(x = COG, y = Counts))
#Thanks to this website - https://www.r-bloggers.com/2013/09/how-to-expand-color-palette-with-ggplot-an

xticks <- c( "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "S",

colourCount = length(unique(Wer_removed_na$COG))
getPalette = colorRampPalette(brewer.pal(9, "Dark2"))
```

```
## Warning in brewer.pal(9, "Dark2"): n too large, allowed maximum for palette Dark2 is 8
## Returning the palette you asked for with that many colors
```

```
p3 <- ggplot(data=Wer_removed_na, aes(x=COG, y=Counts, fill=COG)) + geom_bar(stat="identity") + geom_te
scale_fill_manual(values = getPalette(colourCount)) +  theme_bw()
p3
dev.off()
```

```
## pdf
##   2
```

```
pdf("Filtered_COG_F_endo.pdf", width=12, height=7)
#Now to take the duplicated subset of above data.
#load in the textfile of only genes that are duplicated.
#data_dup_endo <- read_tsv("Friedmanniomyces_endolithicus_CCFEE_5311.v3.KaKs.tsv")
#dups <- data_dup_endo[c(1,2)]

#write.table(dups, file='dups.tsv', quote=FALSE, sep='\t')

renamed_dups <- read_tsv("renamed_dups.txt")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   GeneID = col_character()
## )
```

```
#filtered_dups <- inner_join(split.cog_more, renamed_dups, by="GeneID")
#filtered_dups <- inner_join(renamed_dups, split.cog_more, by="GeneID")

#filtered_dups <- setkey(setDT(renamed_dups), GeneID)[data_endo]
#filtered_dups <- setkey(setDT(data_endo), GeneID)[renamed_dups]
#write.table(filtered_dups, file='filtered_dups.txt', quote=FALSE, sep='\t')
filtered_dups <- read_tsv("filtered_dups.txt")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_character(),
```

```
##   Start = col_double(),
##   Stop = col_double(),
##   `Alias/Synonyms` = col_logical(),
##   InterPro = col_logical(),
##   `GO Terms` = col_logical()
## )
## i Use `spec()` for the full column specifications.
```

```
cog.only <- filtered_dups[c(1,16)]
#write.table(cog.only, file='cog.only.filtered_dups.txt', quote=FALSE, sep='\t')
cog.only_cleaned <- read_tsv("cog.only.cleaned.txt")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   GeneID = col_character(),
##   COG = col_character()
## )
```

```
#split.cog <- cbind(cog.only, fread(text = cog.only$COG, sep = ";", header = FALSE))
#split.cog_more <- cbind(split.cog, fread(text = split.cog$V1, sep = ";", header = FALSE))

#colnames(split.cog_more) = c("GeneID", "COG", "Split1", "Split2")
#split.cog_again <- cbind(split.cog, fread(text = split.cog_more$Split2, sep = ";", header = FALSE))
#colnames(split.cog_again) = c("GeneID", "COG", "Split1", "Split2")

cog.counts.filtered_new <- dplyr::count(cog.only_cleaned,COG)
colnames(cog.counts.filtered_new) = c("COG", "Counts")
removed_na <- na.omit(cog.counts.filtered_new)


xticks <- c( "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "S",
#p <- ggplot(data=removed_na, aes(x=COG, y=Counts, fill=COG))+ geom_bar(stat="identity", position=posit
#Thanks to this website - https://www.r-bloggers.com/2013/09/how-to-expand-color-palette-with-ggplot-an
colourCount = length(unique(removed_na$COG))
getPalette = colorRampPalette(brewer.pal(9, "Dark2"))
```

```
## Warning in brewer.pal(9, "Dark2"): n too large, allowed maximum for palette Dark2 is 8
## Returning the palette you asked for with that many colors
```

```
q1 <- ggplot(data=removed_na, aes(x=COG, y=Counts, fill=COG))+ geom_bar(stat="identity") + geom_text(aes
scale_fill_manual(values = getPalette(colourCount)) +  theme_bw()
q1
dev.off()
```

```
## pdf
##   2
```

```
pdf("Filtered_COG_F_simp.pdf", width=12, height=7)
#Now to take the duplicated subset of above data.
#load in the textfile of only genes that are duplicated.
```

```
#data_dup_endo <- read_tsv("Friedmanniomyces_simplex_CCFEE_5184.v3.KaKs.tsv")
#dups <- data_dup_endo[c(1,2)]

#write.table(dups, file='F.simp.dups.tsv', quote=FALSE, sep='\t')

renamed_dups <- read_tsv("renamed_simp.dups.txt")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   NAQP01_03629 = col_character()
## )
```

```
#filtered_dups <- inner_join(split.cog_more, renamed_dups, by="GeneID")
#filtered_dups <- inner_join(renamed_dups, split.cog_more, by="GeneID")

#filtered_dups <- setkey(setDT(renamed_dups), GeneID)[data_endo]
#filtered_dups <- setkey(setDT(data_endo), GeneID)[renamed_dups]
#write.table(filtered_dups, file='filtered_dups.txt', quote=FALSE, sep='\t')
filtered_dups <- read_tsv("filtered_dups_3.txt")
```

```
## Warning: Missing column names filled in: 'X8' [8], 'X10' [10], 'X11' [11],
## 'X12' [12], 'X13' [13], 'X17' [17], 'X18' [18], 'X19' [19], 'X20' [20],
## 'X21' [21], 'X22' [22], 'X23' [23]
```

```
## Warning: Duplicated column names deduplicated:
## 'ATGCGGTCCAACCGCCTGCTGTCGTTGATTGGCATTGAGAAGATACCGACCTTGGAGCGTGTCGACTTTCGTGATAACAAGCTGTACGACCCTACCGAG
## =>
## 'ATGCGGTCCAACCGCCTGCTGTCGTTGATTGGCATTGAGAAGATACCGACCTTGGAGCGTGTCGACTTTCGTGATAACAAGCTGTACGACCCTACCGAG
## 'ATGCGGTCCAACCGCCTGCTGTCGTTGATTGGCATTGAGAAGATACCGACCTTGGAGCGTGTCGACTTTCGTGATAACAAGCTGTACGACCCTACCGAG
## =>
## 'ATGCGGTCCAACCGCCTGCTGTCGTTGATTGGCATTGAGAAGATACCGACCTTGGAGCGTGTCGACTTTCGTGATAACAAGCTGTACGACCCTACCGAG
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_character(),
##   `973` = col_double(),
##   `1968` = col_double(),
##   X10 = col_logical(),
##   X23 = col_logical()
## )
## i Use `spec()` for the full column specifications.
```

```
## Warning: 1 parsing failure.
##  row col           expected                          actual                  file
## 3571 X23 1/0/T/F/TRUE/FALSE SMCOG1087:hypothetical protein 'filtered_dups_3.txt'
```

```
simp.cog.only <- filtered_dups[c(1,16)]
write.table(simp.cog.only, file='simp.cog.only.filtered_dups.txt', quote=FALSE, sep='\t')
simp.cog.only_cleaned <- read_tsv("simp.cog.only.cleaned.txt")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   GeneID = col_character(),
##   COG = col_character()
## )
```

```
#colnames(simp.cog.only_cleaned) = c("GeneID", "COG")
#split.cog <- cbind(cog.only, fread(text = cog.only$COG, sep = ";", header = FALSE))
#split.cog_more <- cbind(split.cog, fread(text = split.cog$V1, sep = ";", header = FALSE))

#colnames(split.cog_more) = c("GeneID", "COG", "Split1", "Split2")
#split.cog_again <- cbind(split.cog, fread(text = split.cog_more$Split2, sep = ";", header = FALSE))
#colnames(split.cog_again) = c("GeneID", "COG", "Split1", "Split2")

simp.cog.counts.filtered_new <- dplyr::count(simp.cog.only_cleaned,COG)
colnames(simp.cog.counts.filtered_new) = c("COG", "Counts")
simp.removed_na <- na.omit(simp.cog.counts.filtered_new)


xticks <- c( "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "S",
#p <- ggplot(data=removed_na, aes(x=COG, y=Counts, fill=COG))+ geom_bar(stat="identity", position=posit
#Thanks to this website - https://www.r-bloggers.com/2013/09/how-to-expand-color-palette-with-ggplot-an
colourCount = length(unique(removed_na$COG))
getPalette = colorRampPalette(brewer.pal(9, "Dark2"))
```

```
## Warning in brewer.pal(9, "Dark2"): n too large, allowed maximum for palette Dark2 is 8
## Returning the palette you asked for with that many colors
```

```
q2 <- ggplot(data=simp.removed_na, aes(x=COG, y=Counts, fill=COG))+ geom_bar(stat="identity") + geom_te
scale_fill_manual(values = getPalette(colourCount)) +  theme_bw()
q2
dev.off()
```

```
## pdf
##   2
```

```
pdf("Filtered_COG_H_wer.pdf", width=12, height=7)
#Now to take the duplicated subset of above data.
#load in the textfile of only genes that are duplicated.
#data_dup_wer <- read_tsv("Hortaea_werneckii_EXF-2000.v3.KaKs.tsv")
#dups <- data_dup_wer[c(1,2)]

#write.table(dups, file='H.wer.dups.tsv', quote=FALSE, sep='\t')

#H_wer_renamed_dups <- read_tsv("renamed_H_wer.dups.txt")
#filtered_dups <- inner_join(split.cog_more, renamed_dups, by="GeneID")
#filtered_dups <- inner_join(renamed_dups, split.cog_more, by="GeneID")

#filtered_dups <- setkey(setDT(renamed_dups), GeneID)[data_endo]
#filtered_dups <- setkey(setDT(data_endo), GeneID)[renamed_dups]
#write.table(filtered_dups, file='filtered_dups.txt', quote=FALSE, sep='\t')
H_wer_filtered_dups <- read_tsv("filtered_dups_Hwer.txt")
```

```
## Warning: Missing column names filled in: 'X10' [10], 'X12' [12], 'X13' [13],
## 'X18' [18], 'X19' [19], 'X20' [20], 'X21' [21], 'X22' [22]
```

```
## Warning: Duplicated column names deduplicated:
## 'TTGACATGCGCATCCAGGGGCTTTCCTGAACAGGCCTCTGCACTCGTCAAGAGAGGCCGCCGGGTTTTGCATCGCGCTACAGAAAGCCACAGAATCATC
## =>
## 'TTGACATGCGCATCCAGGGGCTTTCCTGAACAGGCCTCTGCACTCGTCAAGAGAGGCCGCCGGGTTTTGCATCGCGCTACAGAAAGCCACAGAATCATC
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_character(),
##   `58` = col_double(),
##   `1609` = col_double(),
##   X10 = col_logical()
## )
## i Use `spec()` for the full column specifications.
```

```
wer.cog.only <- H_wer_filtered_dups[c(1,16)]
write.table(wer.cog.only, file='wer.cog.only.filtered_dups.txt', quote=FALSE, sep='\t')
wer.cog.only_cleaned <- read_tsv("wer.cog.only.cleaned.txt")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   MUNK01_000001 = col_character(),
##   `I:(I) Lipid transport and metabolism` = col_character()
## )
```

```
colnames(wer.cog.only_cleaned) = c("GeneID", "COG")
#split.cog <- cbind(cog.only, fread(text = cog.only$COG, sep = ";", header = FALSE))
#split.cog_more <- cbind(split.cog, fread(text = split.cog$V1, sep = ";", header = FALSE))

#colnames(split.cog_more) = c("GeneID", "COG", "Split1", "Split2")
#split.cog_again <- cbind(split.cog, fread(text = split.cog_more$Split2, sep = ";", header = FALSE))
#colnames(split.cog_again) = c("GeneID", "COG", "Split1", "Split2")

wer.cog.counts.filtered_new <- dplyr::count(wer.cog.only_cleaned,COG)
colnames(wer.cog.counts.filtered_new) = c("COG", "Counts")
wer.removed_na <- na.omit(wer.cog.counts.filtered_new)


xticks <- c( "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "S",
#p <- ggplot(data=removed_na, aes(x=COG, y=Counts, fill=COG))+ geom_bar(stat="identity", position=posit
#Thanks to this website - https://www.r-bloggers.com/2013/09/how-to-expand-color-palette-with-ggplot-an
colourCount = length(unique(wer.removed_na$COG))
getPalette = colorRampPalette(brewer.pal(9, "Dark2"))
```

```
## Warning in brewer.pal(9, "Dark2"): n too large, allowed maximum for palette Dark2 is 8
## Returning the palette you asked for with that many colors
```

```
q3 <- ggplot(data=wer.removed_na, aes(x=COG, y=Counts, fill=COG))+ geom_bar(stat="identity") + geom_tex
scale_fill_manual(values = getPalette(colourCount)) +  theme_bw()
q3
dev.off()
```

```
## pdf
##   2
```

```
#p1
#p2
#p3
#cowplot - grid them together, remove legends, and have only one at the edge
#https://github.com/wilkelab/cowplot/blob/master/vignettes/shared_legends.Rmd
pdf("All_grid.pdf", width=35, height=10)
#All_grid <-plot_grid(p1, p2, p3, labels = c('A', 'B', 'C'), label_size = 12)
#All_grid
prow <- plot_grid(
  p1 + theme(legend.position="none"),
  p2 + theme(legend.position="none"),
  p3 + theme(legend.position="none"),
  align = 'vh',
  labels = c("A", "B", "C"),
  hjust = -1,
  nrow = 1
)
#prow

legend <- get_legend(
  # create some space to the left of the legend
  p1 + theme(legend.box.margin = margin(0, 0, 0, 12))
)
Legend_added <- plot_grid(prow, legend, rel_widths = c(3, .4))

Legend_added
ggsave("All_grid.png",Legend_added, width=35, height=10)
dev.off()
```

```
## pdf
##   2
```

```
#cowplot - grid them together, remove legends, and have only one at the edge
#https://github.com/wilkelab/cowplot/blob/master/vignettes/shared_legends.Rmd
pdf("Filtered_grid.pdf", width=35, height=10)
#Filtered_grid <-plot_grid(q1, q2, q3, labels = c('A', 'B', 'C'), label_size = 12)
filtered_prow <- plot_grid(
  q1 + theme(legend.position="none"),
  q2 + theme(legend.position="none"),
  q3 + theme(legend.position="none"),
  align = 'vh',
  labels = c("A", "B", "C"),
  hjust = -1,
  nrow = 1
```

```
)
#prow

legend <- get_legend(
  # create some space to the left of the legend
  p1 + theme(legend.box.margin = margin(0, 0, 0, 12))
)
Filtered_Legend_added <- plot_grid(filtered_prow, legend, rel_widths = c(3, .4))

Filtered_Legend_added
ggsave("Filtered_grid.png",Filtered_Legend_added, width=35, height=10)
dev.off()
```

```
## pdf
##   2
```

#+ theme(axis.text.x = element_text(vjust = 0.5, hjust=1)) #(breaks=c("(A:(A) RNA processing and modification", #"B:(B) Chromatin structure and dynamics", #"C:(C) Energy production and conversion", #"D:(D) Cell cycle control, cell division, chromosome partitioning", #"E:(E) Amino acid transport and metabolism", #"F:(F) Nucleotide transport and metabolism", #"G:(G) Carbohydrate transport and metabolism", #"H:(G) Coenzyme transport and metabolism", #"I:(I) Lipid transport and metabolism", #"J:(J) Translation, ribosomal structure and biogenesis", #"K:(K) Transcription", #"L:(L) Replication, recombination and repair", #"M:(M) Cell wall/membrane/envelope biogenesis", #"N:(N) Cell motility", #"O:(O) Posttranslational modification, protein turnover, chaperones", #"P:(P) Inorganic ion transport and metabolism", #"Q:(Q) Secondary metabolites biosynthesis, transport and catabolism", #"S:(S) Function unknown", #"T:(T) Signal transduction mechanisms", #"U:(U) Intracellular trafficking, secretion, and vesicular transport", #"V:(V) Defense mechanisms", #"W:(W) Extracellular structures", #"Y:(Y) Nuclear Structure", #"Z:(Z) Cytoskeleton)"), # labels=c("A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "S", "T", "U","V", "W", "Y", "Z"))