Fraction of total counts for categories

Zach Reitz

2025-08-15

```
library(ggplot2)
library(DESeq2)
library(Rmisc)
library(cowplot)
library(scales)
library(grid)
library(gridExtra)
library(ggrepel)
library(khroma)
library(data.table)
```

Load data

Using raw counts

```
counts <- readRDS( "../1_counts_processing/output/raw_counts.Rda")
metadata <- readRDS( "../1_counts_processing/output/metadata.Rda")
kegg_ontology <- readRDS( "../0_databases/output/kegg_ontology.Rda")
kegg_lookup <- readRDS( "../0_databases/output/kegg_lookup.Rda")

timeseries <- c("Starved", "1d", "2d", "4d", "7d", "Starved ")
```

Break into subcategories

```
proportions_list <- lapply(counts, function(this) {
  library_size <- colSums(this)

  # Get genes in each subcategory
  in_cat <- split(kegg_ontology$D, kegg_ontology$B)
  in_cat <- sapply(in_cat, function(x) {
    unique(x[x %in% rownames(this)])
  })

  sum_by_cat <- sapply(in_cat, function(cat) {
    colSums(this[cat, ])
  })

  sum_by_cat <- as.data.frame(t(sum_by_cat))

  ## Convert to fraction
  proportions <- data.frame(sapply(seq_len(ncol(this)), function(i) {
    sum_by_cat[,i] / library_size[i] * 100
  }))

  rownames(proportions) <- rownames(sum_by_cat)
  proportions$desc <- kegg_lookup[rownames(proportions), "Description"]
  proportions
})
```

```
trash_cats <- c(
    "Metabolism of terpenoids and polyketides",
    "Biosynthesis of other secondary metabolites",
    "Xenobiotics biodegradation and metabolism",
    "Not included in regular maps",
    "Information processing in viruses",
    "Cellular community - eukaryotes",
    "Cellular community - prokaryotes"
  )

trash_treats <- c("S.Starved", "NH4", "WellFed.10", "WellFed")

df_list <- lapply(proportions_list, function(prop) {
  # Longify
  df2 <- reshape2::melt(as.data.frame(prop), id.vars = "desc")

  df2 <- within(df2, {
    value <- as.numeric(value)
    replicate <- metadata[variable, "replicate"]
    treatment <- metadata[variable, "treatment"]
  })

  # remove NH4, superstarved, wellfed.10
  df2 <- df2[!df2$treatment %in% trash_treats, ]

  # Remove some useless categories
  df2 <- df2[!df2$desc %in% trash_cats, ]
  df2
})

names(df_list) <- c("MC", "KN")
df_combo <- data.table::rbindlist(df_list, idcol = "org")
starved2 <- df_combo[df_combo$treatment == "Starved", ]
starved2$treatment <- "Starved "
df_combo <- rbind(df_combo, starved2)

df_combo$treatment <- factor(df_combo$treatment, levels = timeseries)
df_combo$org <- factor(df_combo$org, levels = c("MC", "KN"))

times <- c("Starved" = 0, "1d" = 1, "2d" = 2, "4d" = 4, "7d" = 7, "Starved " = 10)
df_combo$times <- times[as.character(df_combo$treatment)]
```

```
# Export for other notebooks
saveRDS(df_combo, file = "output/category_proportions.Rda")

export <- subset(df_combo, times != 10)
```

```
write.csv(export, "output/category_proportions.csv", row.names = F)
```

Correlations between MC and KN

```
cats <- proportions_list[[1]]$desc
cats <- cats[!cats %in% trash_cats]
# Filter useless categories
dfs <- lapply(proportions_list, function(x) subset(x, !x$desc %in% trash_cats, select = -desc))
# Filter other samples
keep <- ! metadata[substr(colnames(dfs[[1]]),2,3) ,"treatment"] %in% trash_treats
dfs <- lapply(dfs, subset, select = keep)

# Correlate MC and KN for each category
cors <- lapply(seq_along(cats), function(i) {
  cor <- cor.test(
    t(dfs[[1]][i, ]),
    t(dfs[[2]][i, ])
  )
  c(cor$estimate, "p.value" = cor$p.value)
})
cors <- do.call(rbind, cors)
rownames(cors) <- cats

cors <- as.data.frame(cors)

cors <- cors[order(cors$cor),]

cors
```

```
##                                                cor      p.value
## Lipid metabolism                      -0.681145256 7.316584e-03
## Translation                           -0.632715968 1.516535e-02
## Transcription                         -0.495315300 7.171166e-02
## Carbohydrate metabolism               -0.367152960 1.965814e-01
## Signal transduction                   -0.234198362 4.203048e-01
## Energy metabolism                     -0.222289194 4.449756e-01
## Membrane transport                    -0.095537251 7.452691e-01
## Folding, sorting and degradation      -0.067365195 8.190121e-01
## Signaling molecules and interaction   -0.007748215 9.790274e-01
## Amino acid metabolism                 -0.001481194 9.959904e-01
## Metabolism of other amino acids        0.060667942 8.367730e-01
## Metabolism of cofactors and vitamins   0.134687235 6.461841e-01
## Cell growth and death                  0.216762062 4.566635e-01
## Chromosome                             0.357800904 2.090992e-01
## Nucleotide metabolism                  0.373168649 1.887869e-01
## Glycan biosynthesis and metabolism     0.593423997 2.527950e-02
## Cell motility                          0.838556032 1.788274e-04
## Transport and catabolism               0.869730227 5.298158e-05
## Replication and repair                 0.884910190 2.606881e-05
```

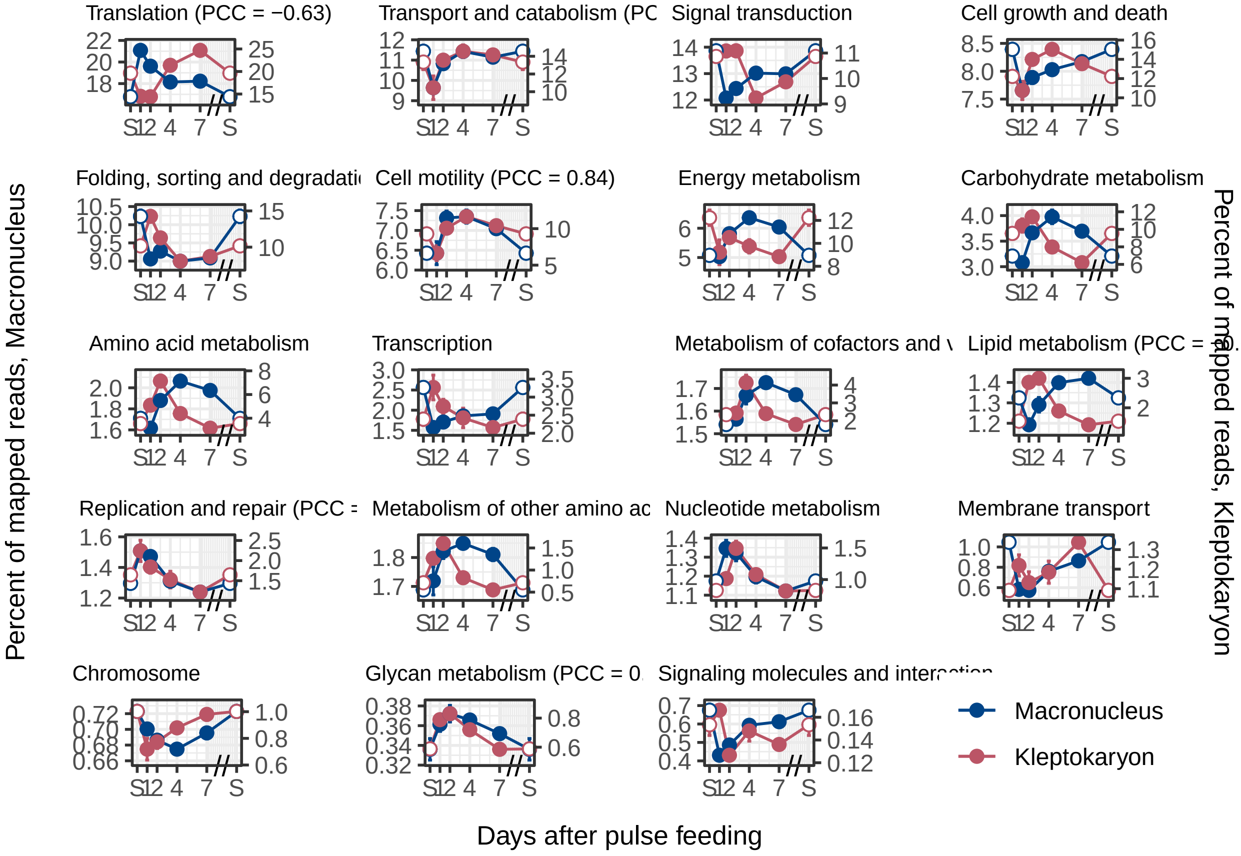
```
fraction_summary <- summarySE(data = df_combo, "value", groupvars = c("desc", "times", "org"))
fraction_summary <- within(fraction_summary, {
  min <- value - se
  max <- value + se
})

# Split by category
cat_dfs <- split(fraction_summary, fraction_summary$desc)
means <- sapply(cat_dfs, function(x) mean(x$value))
cat_dfs <- cat_dfs[order(means, decreasing = T)]
```

Plotting

```
## Shared layers
x_axis_ticks <- list(scale_x_continuous(breaks = c(0, 1, 2, 4, 7, 10), labels = c("S", 1, 2, 4, 7, "S")
                     minor_breaks = c(3,5,6, seq(7, 10, by = 3/14))))
```

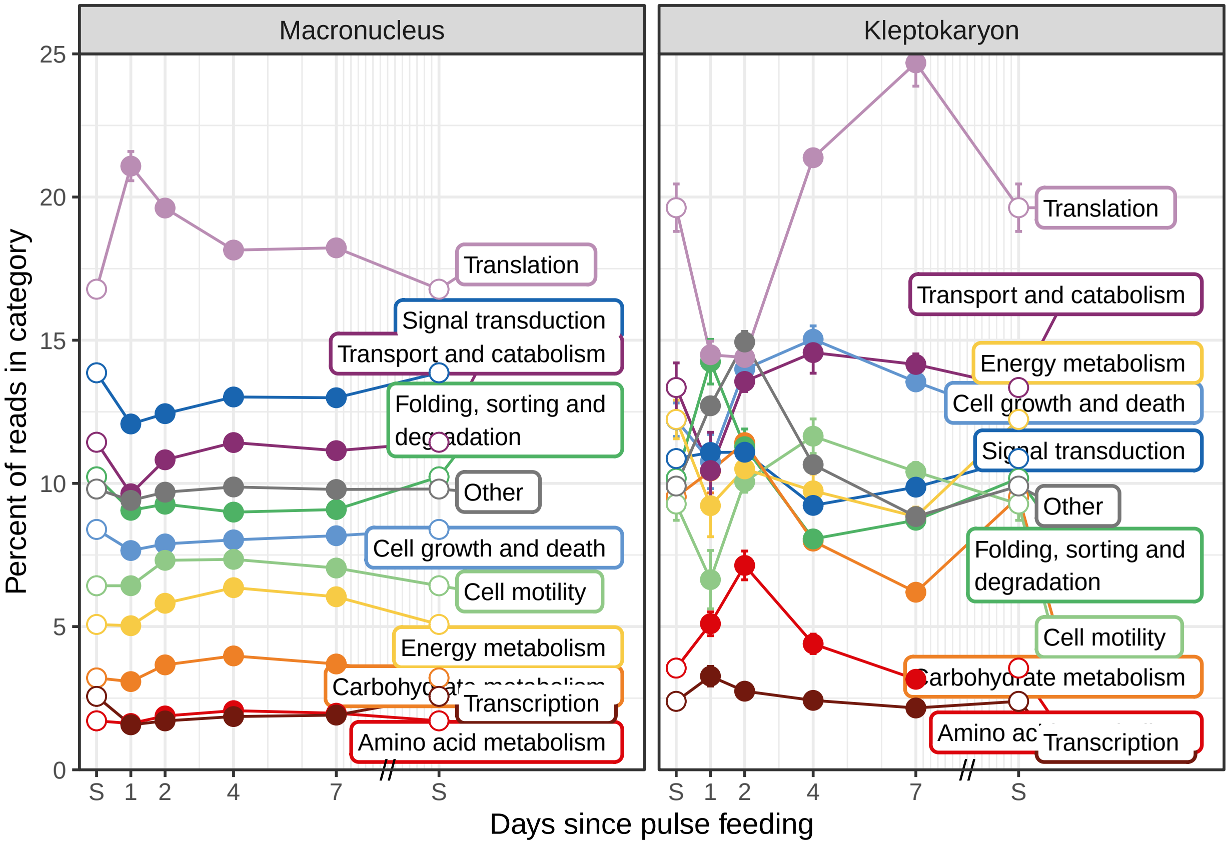
Left and right axes with different scales

```
plts <- lapply(cat_dfs, function(d) {
  cat <- d$desc[[1]]

  # Add correlation to title if significant
  cor <- cors[cat, ]
  if (startsWith(cat, "Glycan")) { # Glycan too long
    cat <- "Glycan metabolism"
  }
  if (cor[2] < 0.05) {
    cat <- paste0(cat, " (PCC = ", round(cor[1], 2), ")")
  }

  # Calculate transformation
  means <- aggregate(d$value, list(Organism = d$org), mean)$x
  mins <- aggregate(d$value, list(Organism = d$org), min)$x
  maxes <- aggregate(d$value, list(Organism = d$org), max)$x
  ranges <- maxes - mins

  # Transform
  d[d$org == "KN", c("value", "min", "max")] <- (d[d$org == "KN", c("value", "min", "max")] - mins[2])

  # Calculations for y limits
  ymin <- min(d$min)
  ymax <- max(d$max)
  yrange <- ymax - ymin

  # Plot MC on top
  d$org <- factor(d$org, levels = c("KN", "MC"))

  # Horizontally offset MC and KN (?)
  dodge <- 0

  ggplot(d, aes(x = times, y = value, group = org, color = org)) +
    geom_errorbar(aes(ymin = min, ymax = max), width = 0.8, position = position_dodge(width = dodge)) +
    geom_line(position = position_dodge(width = dodge)) +
    geom_point(position = position_dodge(width = dodge), size = 2) +
    # White point for starved
    geom_point(position = position_dodge(width = dodge), size = 2,
      data = subset(d, d$times %in% c(0,10)), fill = "white", shape = 21, show.legend = F) +

    scale_y_continuous(
      breaks = breaks_pretty(3),
      #labels = function(x) format(x, digits = 2),
      sec.axis = sec_axis(
        ~. * ranges[2] / ranges[1] - mins[1] * ranges[2] / ranges[1] + mins[2],
        breaks = breaks_pretty(3)
      )) +

    scale_color_manual(values = c("#BB5566", "#004488"),
      labels = c("Kleptokaryon", "Macronucleus"),
      guide = guide_legend(reverse = T, title = NULL)) +
    theme_bw() +
    theme(
      #axis.text.x = element_text(angle = 0, vjust = 1, hjust=1),
      plot.title = element_text(size = 8),
      plot.title.position = "plot",
      legend.position = "none",
      aspect.ratio = 0.6
      ) +
    #labs(y = "Log-ratio transformed fraction of reads", title = cat)
    labs(x = NULL, y = NULL, title = cat) +
    coord_cartesian(xlim = c(-0.5, 10.5),
                    ylim = c(ymin - 0.1 * yrange, ymax + 0.1 * yrange),
                    clip = "off", expand = F) +
    annotate("text", x = 8.5, y = ymin - 0.1 * yrange, label = "//") +
    x_axis_ticks

})

#plts[1:15] <- lapply(plts[1:15], function(x) x + theme(axis.title.x = element_blank(), axis.text.x = e

legend <- get_plot_component(plts[[1]] + theme(legend.position = "right"), "guide-box-right")
plts <- append(plts, list(legend))

plt <- plot_grid(plotlist = plts, nrow = 5, ncol = 4)

  #add labels to plot
grid.arrange(arrangeGrob(plt,
  left = textGrob("Percent of mapped reads, Macronucleus", gp=gpar(fontsize=10), rot=90),
  right = textGrob("Percent of mapped reads, Kleptokaryon", gp=gpar(fontsize=10), rot=-90),
  bottom = textGrob("Days after pulse feeding", gp=gpar(fontsize=10))
))
```



Plotting everything with the same scale

```
# Everything not in the top 10 categories becomes "Other"
means <- sapply(cat_dfs, function(x) mean(x$value))
top10 <- names(sort(means, decreasing = T))[1:10]

not_top_data <- df_combo[!df_combo$desc %in% top10, ]
not_top_sums <- aggregate(not_top_data, value ~ org + variable + treatment + times, sum)

not_top_summary <- summarySE(data = not_top_sums, "value", groupvars = c("org", "times"))
not_top_summary <- within(not_top_summary, {
  desc <- "Other"
  min <- value - se
  max <- value + se
})

top10_summary <- fraction_summary[fraction_summary$desc %in% top10, ]

combo_summary <- rbind(top10_summary, not_top_summary)
# Make other gray
combo_summary$desc <- factor(combo_summary$desc, levels = c(names(means), "Other"))

tol = c(khroma::color("discreterainbow")(23)[c(4,7, 8, 10, 12, 13, 16, 18, 20, 22)], "#777777")

plt <- ggplot(combo_summary, aes(x = times, y = value, group = desc, color = desc)) +
  geom_line(show.legend = F) +
  geom_errorbar(aes(ymin = min, ymax = max), width = 0.2, show.legend = F) +
  # First one does a colored box and the second does a black label
  geom_label_repel(data = subset(combo_summary, times == 10),
                   aes(label = stringr::str_wrap(desc, 25)), seed = 123, size = 3,
                   nudge_x = 1, label.size = 1.3,
                   max.overlaps = Inf, show.legend = F, hjust = "left", direction = "y") +
  geom_label_repel(data = subset(combo_summary, times == 10),
                   aes(label = stringr::str_wrap(desc, 25)), seed = 123, size = 3,
                   nudge_x = 1, color = "black", label.size = NA, segment.colour = NA,
                   max.overlaps = Inf, show.legend = F, hjust = "left", direction = "y") +
  geom_point(show.legend = F, size = 3) +
  # White point
  geom_point(data = subset(combo_summary, times == 10 | times == 0),
             fill = "white", shape = 21,
             show.legend = F, size = 3) +
  facet_wrap(vars(org), labeller = labeller(org = c("MC" = "Macronucleus", "KN" = "Kleptokaryon"))) +
  labs(x = "Days since pulse feeding", y = "Percent of reads in category") +
  scale_x_continuous(breaks = c(0, 1, 2, 4, 7, 10), labels = c("S", 1, 2, 4, 7, "S"),
                     minor_breaks = c(3,5,6, seq(7, 10, by = 3/14))) +
  coord_cartesian(xlim = c(-0.5, 16), ylim = c(0, 25), clip = "off", expand = F) +
  theme_bw() +
  scale_color_manual(values = tol) +
  theme(strip.text = element_text(size = 10)) +
  annotate("text", x = 8.5, y = 0, label = "//")

plt
```


Macronucleus
Kleptokaryon
Percent of reads in category
Translation
Signal transduction
Transport and catabolism
Folding, sorting and degradation
Other
Cell growth and death
Cell motility
Energy metabolism
Carbohydrate metabolism
Transcription
Amino acid metabolism
Translation
Transport and catabolism
Energy metabolism
Cell growth and death
Signal transduction
Other
Folding, sorting and degradation
Cell motility
Carbohydrate metabolism
Amino ac
Transcription
S
//
S
S
//
S
Days since pulse feeding