Challenges + job category

Overview

Secondary analysis of survey Q9: "How frequently have you encountered the following challenges while working on open-source projects?"

In this script, I am considering challenges in light of job category.

Import packages and utilities

```
project_root <- here::here() # requires that you be somewhere in the
# project directory (not above it)
# packages
suppressMessages(source(file.path(project_root, "scripts/packages.R")))
# functions and objects used across scripts
suppressMessages(source(file.path(project_root, "scripts/utils.R")))
```

Load data

```
challenges <- load_qualtrics_data("clean_data/challenges_Q9.tsv")
other_quant <- load_qualtrics_data("clean_data/other_quant.tsv")
```

Wrangle data

```
challenges_and_job <- challenges
challenges_and_job$job_category <- other_quant$job_category

head(challenges_and_job)
```

```
  Coding time Documentation time Managing issues Attracting users  Recognition
1      Always             Always          Always            Always       Always
2  Frequently       Occasionally    Occasionally      Occasionally Occasionally
3  Frequently             Always    Occasionally            Always Occasionally
4      Always             Always      Frequently      Occasionally   Frequently
5      Always             Always          Rarely      Occasionally   Frequently
6
      Hiring     Security Finding peers Finding mentors Education time
1     Always       Always        Always          Always         Always
2     Rarely   Frequently  Occasionally      Frequently     Frequently
3 Frequently   Frequently  Occasionally    Occasionally         Rarely
4     Always Occasionally        Rarely          Rarely     Frequently
5      Never        Never         Never           Never         Always
6
  Educational resources        Legal Finding funding Securing funding
1                Always       Always          Always           Always
2            Frequently   Frequently      Frequently     Occasionally
3                Rarely       Always          Always           Always
4                Rarely Occasionally      Frequently       Frequently
5          Occasionally Occasionally          Rarely           Always
6
          job_category
1              Faculty
2             Post-Doc
3 Other research staff
4              Faculty
5              Faculty
6 Other research staff
```

Remove rows that contain any empty entries.

```
nrow(challenges_and_job)
```

```
[1] 332
```

```
challenges_and_job <- exclude_empty_rows(challenges_and_job, strict = TRUE) # from scripts/u
nrow(challenges_and_job)
```

```
[1] 233
```

For visual clarity in our plots, let's combine postdocs and other staff researchers, as well as undergrads and grad students.

```r
challenges_and_job <- challenges_and_job %>%
  mutate(
    job_category = recode(
      job_category,
      "Post-Doc" = "Postdocs and Staff Researchers",
      "Other research staff" = "Postdocs and Staff Researchers"
    )
  )

challenges_and_job <- challenges_and_job %>%
  mutate(
    job_category = recode(
      job_category,
      "Grad Student" = "Students",
      "Undergraduate" = "Students"
    )
  )

challenges_and_job$participantID <- row.names(challenges_and_job)

head(challenges_and_job)
```

```
  Coding time Documentation time Managing issues Attracting users  Recognition
1      Always             Always          Always           Always       Always
2  Frequently       Occasionally    Occasionally     Occasionally Occasionally
3  Frequently             Always    Occasionally           Always Occasionally
4      Always             Always      Frequently     Occasionally   Frequently
5      Always             Always          Rarely     Occasionally   Frequently
7  Frequently         Frequently      Frequently       Frequently   Frequently
      Hiring     Security Finding peers Finding mentors Education time
1     Always       Always        Always          Always         Always
2     Rarely   Frequently  Occasionally      Frequently     Frequently
3 Frequently   Frequently  Occasionally    Occasionally         Rarely
4     Always Occasionally        Rarely          Rarely     Frequently
5      Never        Never         Never           Never         Always
7     Always        Never         Never           Never     Frequently
  Educational resources        Legal Finding funding Securing funding
1                Always       Always          Always           Always
2            Frequently   Frequently      Frequently     Occasionally
```

```
3                 Rarely       Always          Always            Always
4                 Rarely Occasionally     Frequently        Frequently
5           Occasionally Occasionally          Rarely            Always
7                  Never       Always          Always            Always
                    job_category participantID
1                        Faculty             1
2 Postdocs and Staff Researchers             2
3 Postdocs and Staff Researchers             3
4                        Faculty             4
5                        Faculty             5
7                        Faculty             7
```

Let's reshape the data from wide to long format for easier counting and plotting.

```r
long_data <- challenges_and_job %>%
  pivot_longer(
    cols = -c(participantID, job_category),
    names_to = "challenge",
    values_to = "challenge_level"
  )

long_data
```

```
# A tibble: 3,262 x 4
   job_category participantID challenge          challenge_level
   <chr>        <chr>         <chr>              <chr>
 1 Faculty      1             Coding time        Always
 2 Faculty      1             Documentation time Always
 3 Faculty      1             Managing issues    Always
 4 Faculty      1             Attracting users   Always
 5 Faculty      1             Recognition        Always
 6 Faculty      1             Hiring             Always
 7 Faculty      1             Security           Always
 8 Faculty      1             Finding peers      Always
 9 Faculty      1             Finding mentors    Always
10 Faculty      1             Education time     Always
# i 3,252 more rows
```

Top 5 challenges per group, by "points"

```
long_data <- long_data %>%
  mutate(
    challenge_score = recode(
      challenge_level,
      "Never"            = 0L,
      "Non-applicable"   = 0L,
      "Rarely"           = 1L,
      "Occasionally"     = 2L,
      "Frequently"       = 3L,
      "Always"           = 4L
    )
  )
# Using interger literals 0L, 1L, etc., ensures that
# the new column will be integers, not doubles.

long_data
```

```
# A tibble: 3,262 x 5
   job_category participantID challenge          challenge_level challenge_score
   <chr>        <chr>         <chr>              <chr>                     <int>
 1 Faculty      1             Coding time        Always                        4
 2 Faculty      1             Documentation time Always                        4
 3 Faculty      1             Managing issues    Always                        4
 4 Faculty      1             Attracting users   Always                        4
 5 Faculty      1             Recognition        Always                        4
 6 Faculty      1             Hiring             Always                        4
 7 Faculty      1             Security           Always                        4
 8 Faculty      1             Finding peers      Always                        4
 9 Faculty      1             Finding mentors    Always                        4
10 Faculty      1             Education time     Always                        4
# i 3,252 more rows
```

Let's just inspect all the basic stats for all the challenges for each group.

```
# Helper to compute the (numeric) mode
get_mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
```

```r
}

get_summary_df <- function(job_str, df) {
  res <- df %>%
    filter(job_category == job_str) %>%
    group_by(challenge) %>%
    summarise(
      total = sum(challenge_score),
      mean = mean(challenge_score, na.rm = TRUE),
      median = median(challenge_score),
      mode = get_mode(challenge_score),
      st_dev = sd(challenge_score, na.rm = TRUE)
    ) %>%
    ungroup() %>%
    arrange(desc(total))
  return(res)
}
```

```r
jobs_ordered <- c(
  "Faculty",
  "Postdocs and Staff Researchers",
  "Students",
  "Non-research Staff"
)

summary_tables <- lapply(jobs_ordered, function(j) get_summary_df(j, long_data))
names(summary_tables) <- jobs_ordered
summary_tables
```

```
$Faculty
# A tibble: 14 x 6
   challenge              total  mean median  mode st_dev
   <chr>                  <int> <dbl>  <int> <int>  <dbl>
 1 Documentation time       176  2.98      3     4   1.22
 2 Coding time              170  2.88      3     4   1.31
 3 Securing funding         158  2.68      3     4   1.58
 4 Finding funding          149  2.53      3     4   1.57
 5 Education time           142  2.41      2     4   1.35
 6 Managing issues          115  1.95      2     3   1.41
 7 Hiring                   114  1.93      2     0   1.65
 8 Attracting users         113  1.92      2     2   1.34
 9 Recognition              108  1.83      2     0   1.52
```

```
10 Educational resources    84  1.42      1     0   1.29
11 Legal                    84  1.42      1     0   1.32
12 Finding mentors          76  1.29      1     0   1.40
13 Finding peers            76  1.29      1     0   1.19
14 Security                 76  1.29      1     0   1.35

$`Postdocs and Staff Researchers`
# A tibble: 14 x 6
   challenge             total  mean median  mode st_dev
   <chr>                 <int> <dbl>  <int> <int>  <dbl>
 1 Documentation time      174  3.16      3     3  0.788
 2 Coding time             145  2.64      3     3  1.16
 3 Education time          133  2.42      3     3  1.30
 4 Securing funding        120  2.18      3     4  1.72
 5 Finding funding         118  2.15      2     4  1.60
 6 Attracting users        116  2.11      2     3  1.49
 7 Managing issues         113  2.05      2     2  1.30
 8 Recognition              88  1.6       2     2  1.24
 9 Educational resources    85  1.55      1     1  1.15
10 Legal                    84  1.53      1     1  1.17
11 Finding mentors          73  1.33      1     0  1.28
12 Hiring                   68  1.24      0     0  1.59
13 Security                 62  1.13      1     0  1.19
14 Finding peers            58  1.05      1     0  0.970

$Students
# A tibble: 14 x 6
   challenge             total  mean median  mode st_dev
   <chr>                 <int> <dbl>  <int> <int>  <dbl>
 1 Coding time              85 2.58       3     3   1.12
 2 Documentation time       84 2.55       3     3   1.37
 3 Education time           75 2.27       2     3   1.21
 4 Attracting users         59 1.79       2     0   1.54
 5 Educational resources    59 1.79       2     1   1.27
 6 Managing issues          57 1.73       2     2   1.35
 7 Finding mentors          53 1.61       2     0   1.25
 8 Finding funding          52 1.58       1     0   1.64
 9 Securing funding         49 1.48       1     0   1.64
10 Recognition              46 1.39       1     0   1.32
11 Legal                    42 1.27       1     0   1.35
12 Finding peers            41 1.24       1     0   1.35
13 Security                 26 0.788      0     0   1.34
14 Hiring                   18 0.545      0     0   1.23
```

```
$`Non-research Staff`
# A tibble: 14 x 6
   challenge              total  mean median  mode st_dev
   <chr>                  <int> <dbl>  <dbl> <int>  <dbl>
 1 Documentation time       252  2.93      3     3  0.968
 2 Coding time              206  2.40      3     3  1.26
 3 Education time           189  2.20      2     3  1.21
 4 Managing issues          166  1.93      2     2  1.18
 5 Attracting users         154  1.79      2     0  1.48
 6 Security                 143  1.66      2     2  1.28
 7 Educational resources    141  1.64      2     1  1.11
 8 Legal                    123  1.43      1     2  1.20
 9 Finding mentors          121  1.41      1     0  1.30
10 Finding funding          113  1.31      0     0  1.63
11 Securing funding         111  1.29      0     0  1.65
12 Finding peers             92  1.07      1     0  1.09
13 Recognition               92  1.07      1     0  1.23
14 Hiring                    91  1.06      0     0  1.35
```

Let's plot the top 5. First, a little data wrangling.

```r
all_tbl <- bind_rows(summary_tables, .id = "job_category")

top5_by_job <- all_tbl %>%
  group_by(job_category) %>%
  slice_max(mean, n = 5, with_ties = FALSE) %>%
  ungroup() %>%
  select(job_category, challenge, mean) %>%
  mutate(job_category = factor(job_category, levels = jobs_ordered))

# Reorder factor levels for visual clarity
ordered_challenges_top5_by_points <- c(
  "Documentation time",
  "Coding time",
  "Education time",
  "Educational resources",
  "Securing funding",
  "Finding funding",
  "Managing issues",
  "Attracting users"
)
```

```
top5_by_job$challenge <- factor(
  top5_by_job$challenge,
  levels = ordered_challenges_top5_by_points
)
```

Let's add a whitespace in this long job category name

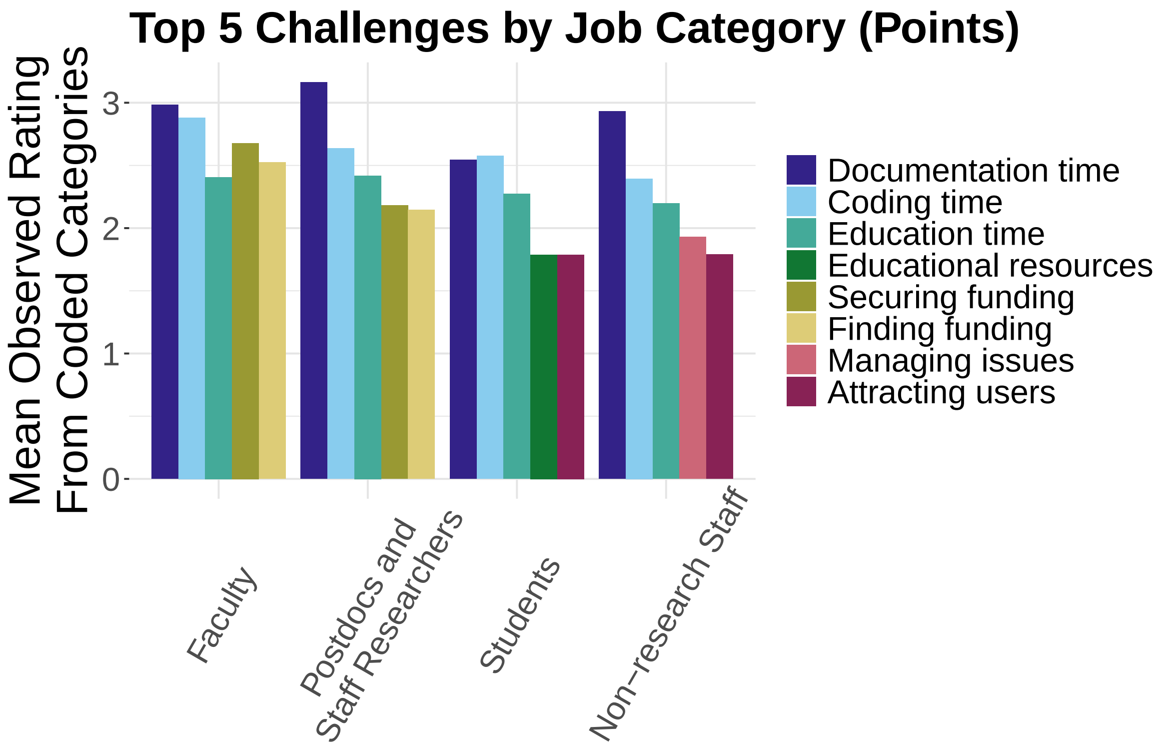
```
top5_by_job <- top5_by_job %>%
  mutate(
    job_category = recode(
      job_category,
      "Postdocs and Staff Researchers" = "Postdocs and\nStaff Researchers",
    )
  )
```

Let's hard-code a color palette that is tailored to these data. This will be useful in the next section, when we plot almost the same set of challenges, and we'll want the challenges to correspond to the same colors in the legend.

```
# I'm just including the names here for my own reference,
# but they're not actually used in the code.
chall_colors <- list(
  # modified from https://sronpersonalpages.nl/~pault/
  "Documentation time" = "#332288",
  "Coding time" = "#88CCEE",
  "Education time" = "#44AA99",
  "Educational resources" = "#117733",
  "Securing funding" = "#999933",
  "Finding funding" = "#DDCC77",
  "Managing issues" = "#CC6677",
  "Attracting users" = "#882255"
)
```

```
top5_by_points_plot <- ggplot(
  top5_by_job,
  aes(
    x = job_category,
    y = mean,
    fill = challenge
  )
) +
```

```
  geom_col(position = position_dodge()) +
  scale_fill_manual(values = chall_colors) +
  labs(
    x = "Job Category",
    y = "Mean Observed Rating\nFrom Coded Categories",
    fill = "Challenge",
    title = "Top 5 Challenges by Job Category (Points)"
  ) +
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_text(size = 24),
    axis.text.x = element_text(angle = 60, vjust = 0.6, size = 18),
    axis.text.y = element_text(size = 18),
    axis.ticks.x = element_blank(),
    legend.title = element_blank(),
    legend.text = element_text(size = 18),
    panel.background = element_blank(),
    panel.grid = element_line(linetype = "solid", color = "gray90"),
    plot.title = element_text(hjust = 0, size = 24, face = "bold"),
    plot.margin = unit(c(0.3, 0.3, 0.3, 0.3), "cm")
  )
top5_by_points_plot
```




```
save_plot("top5_challenges_by_job_points.tiff", 12, 10, p=top5_by_points_plot)
```

Top 5 challenges per group, by proportion frequently or always

As another way of confirming/exploring these trends, let's look at the proportion of each group who said "frequently" or "always".

```
# Calculate proportion of TRUEs by taking the mean of a logical vector,
# created by %in%.
proportions <- long_data %>%
  group_by(job_category, challenge) %>%
  summarize(proportion = mean(challenge_level %in% c("Frequently", "Always"))) %>%
  ungroup()
```

```
`summarise()` has grouped output by 'job_category'. You can override using the
`.groups` argument.
```

```r
proportions
```

```
# A tibble: 56 x 3
   job_category challenge               proportion
   <chr>        <chr>                        <dbl>
 1 Faculty      Attracting users             0.356
 2 Faculty      Coding time                  0.712
 3 Faculty      Documentation time           0.763
 4 Faculty      Education time               0.492
 5 Faculty      Educational resources        0.186
 6 Faculty      Finding funding              0.627
 7 Faculty      Finding mentors              0.220
 8 Faculty      Finding peers                0.169
 9 Faculty      Hiring                       0.475
10 Faculty      Legal                        0.169
# i 46 more rows
```

```r
top5_by_prop <- proportions %>%
  group_by(job_category) %>%
  slice_max(order_by = proportion, n = 5)
```

```r
# Filter to include only challenges present in the top5 dataframe
filtered_props <- proportions %>%
  semi_join(top5_by_prop, by = c("job_category", "challenge"))
```

Let's inspect the challenges that made the cut. Are they the same as the challenges from calculating the top 5 the other way (by points)?

```r
ordered_challenges_top5_by_points
```

```
[1] "Documentation time"    "Coding time"           "Education time"
[4] "Educational resources" "Securing funding"      "Finding funding"
[7] "Managing issues"       "Attracting users"
```

```r
unique(filtered_props$challenge)
```

```
[1] "Coding time"        "Documentation time" "Education time"
[4] "Finding funding"    "Securing funding"   "Attracting users"
[7] "Managing issues"
```

They are not the same. In this case, educational resources didn't make the cut. We can still use the same order of factor levels.

```
# Reorder factor levels
filtered_props$challenge <- factor(
  filtered_props$challenge,
  levels = ordered_challenges_top5_by_points
)

filtered_props$job_category <- factor(
  filtered_props$job_category,
  levels = jobs_ordered
)
```

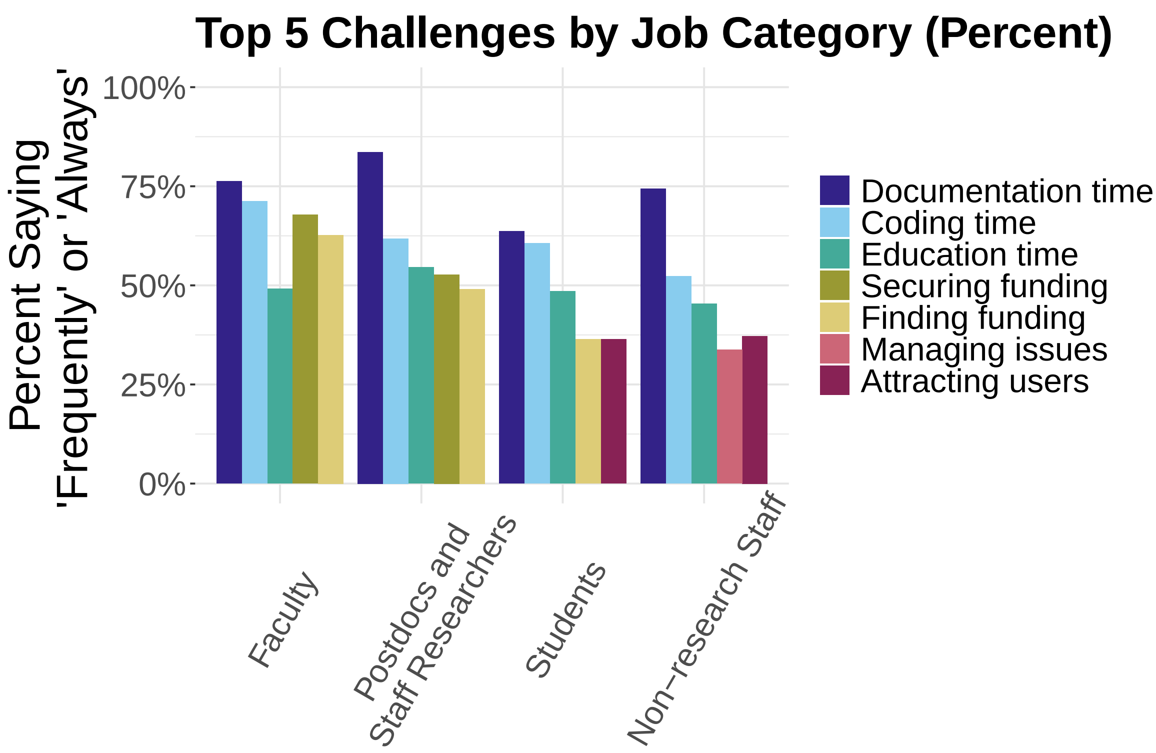
Let's add a whitespace in this long job category name

```
filtered_props <- filtered_props %>%
  mutate(
    job_category = recode(
      job_category,
      "Postdocs and Staff Researchers" = "Postdocs and\nStaff Researchers",
    )
  )
```

Custom color palette, which doesn't contain "Educational resources"

```
# I'm just including the names here for my own reference,
# but they're not actually used in the code.
chall_colors2 <- list(
  # modified from https://sronpersonalpages.nl/~pault/
  "Documentation time" = "#332288",
  "Coding time" = "#88CCEE",
  "Education time" = "#44AA99",
  #"Educational resources" = "#117733",
  "Securing funding" = "#999933",
  "Finding funding" = "#DDCC77",
  "Managing issues" = "#CC6677",
  "Attracting users" = "#882255"
)
```

```
top5_by_perc_plot <- ggplot(
  filtered_props,
  aes(
    x = job_category,
    y = proportion,
    fill = challenge
  )
) +
  geom_col(position = position_dodge()) +
  scale_y_continuous(labels = scales::percent, limits = c(0, 1)) +
  scale_fill_manual(values = chall_colors2) +
  labs(
    x = "Job Category",
    y = "Percent Saying\n'Frequently' or 'Always'",
    fill = "Challenge",
    title = "Top 5 Challenges by Job Category (Percent)"
  ) +
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_text(size = 24),
    axis.text.x = element_text(angle = 60, vjust = 0.6, size = 18),
    axis.text.y = element_text(size = 18),
    axis.ticks.x = element_blank(),
    legend.title = element_blank(),
    legend.text = element_text(size = 18),
    panel.background = element_blank(),
    panel.grid = element_line(linetype = "solid", color = "gray90"),
    plot.title = element_text(hjust = 0, size = 24, face = "bold"),
    plot.margin = unit(c(0.3, 0.3, 0.3, 0.3), "cm")
  )
top5_by_perc_plot
```




```
save_plot("top5_challenges_by_job_percent.tiff", 12, 10, p=top5_by_perc_plot)
```

```
p_combined <- patchwork::wrap_plots(
  top5_by_points_plot,
  plot_spacer(),
  top5_by_perc_plot
) +
  plot_layout(widths = c(1, 0.05, 1)) +
  theme(plot.margin = margin(t = 1, r = 1, b = 1, l = 1, unit = "cm"))
p_combined <- p_combined +
  plot_annotation(tag_levels = "A") &
  theme(plot.tag = element_text(size = 26))

# SVG is higher quality
svglite::svglite(
  file.path(FIGURE_PATH, "figureS7.svg"),
  width = 26,
  height = 10
)
```

```
print(p_combined)
dev.off()
```

```
pdf
  2
```

Consider clusters

Exploratory plot

In a previous notebook, we found that the distributions of responses to the various challenges could be clustered like so:

Cluster 1:
Education time
Documentation time
Coding time

Cluster 2:
Securing funding
Hiring
Finding funding

Cluster 3:
Everthing else

This makes me curious: does the distribution of job categories also vary by cluster? Before we try any statistics, let's just make a plot. This will be a variation of the detailed plot above.

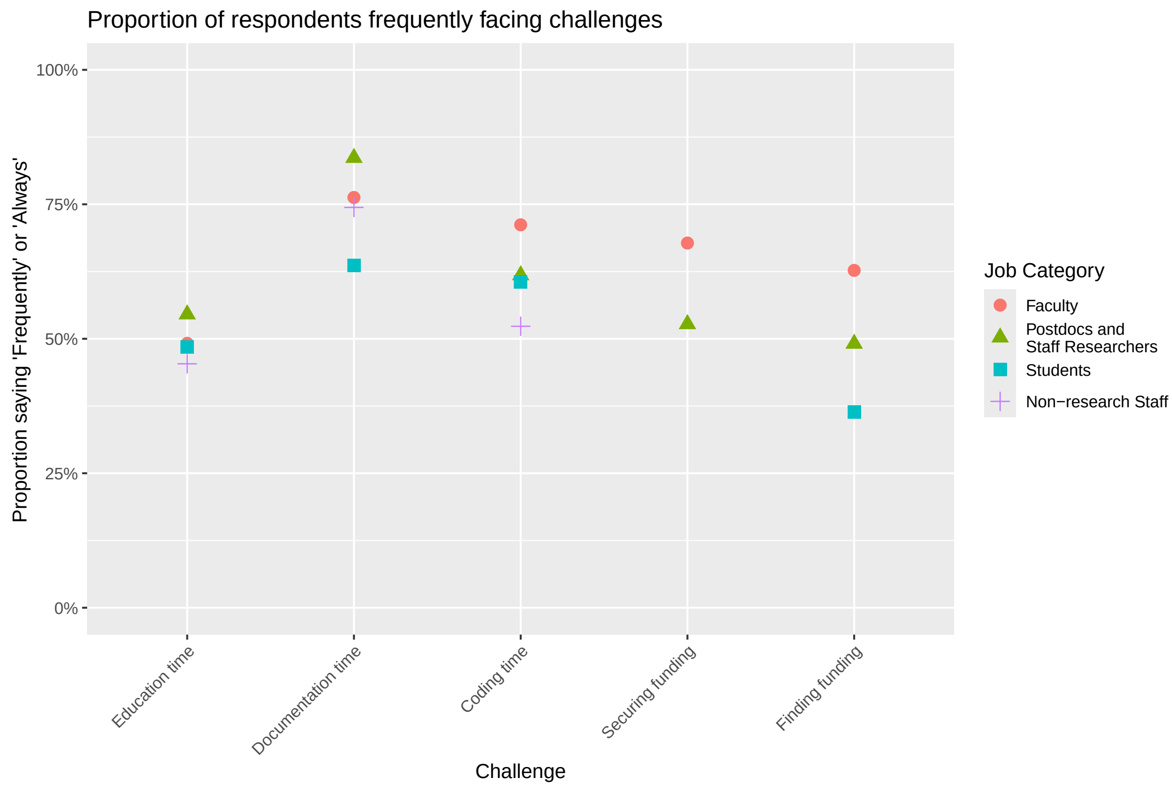
We're just going to subset the "frequently" or "always" data to include only clusters 1 and 2, and we'll reorder the factor levels accordingly.

```
clusters1and2 <- c(
  "Education time",
  "Documentation time",
  "Coding time",
  "Securing funding",
  "Finding funding",
  "Hiring"
)
```

```r
to_plot_clusters <- subset(filtered_props, challenge %in% clusters1and2)

to_plot_clusters$challenge <- factor(
  to_plot_clusters$challenge,
  levels = clusters1and2
)
```

```r
challenges_plot_clusters1and2 <- ggplot(
  to_plot_clusters,
  aes(
    x = challenge,
    y = proportion,
    group = job_category,
    color = job_category,
    shape = job_category
  )
) +
  geom_point(size = 3) +
  scale_y_continuous(labels = scales::percent, limits = c(0, 1)) +
  labs(
    x = "Challenge",
    y = "Proportion saying 'Frequently' or 'Always'",
    color = "Job Category",
    shape = "Job Category",
    title = "Proportion of respondents frequently facing challenges"
  ) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
challenges_plot_clusters1and2
```



Hm. Well, the results for cluster 1 are a little messy, which kind of makes sense, since you'd expect these to be challenges everyone struggles with. The only obvious trend to me is that undergrads struggle less than everyone else with education time and documentation time. But this group is too small to conclude anything with confidence.

Cluster 2 is a bit more interesting. It seems, at a glance, like it's pretty safe to say that faculty struggle with these challenges more than everyone else, with postdocs and staff researchers close behind.

I guess we should do a regression to test it.

Regression for cluster 2

Let's once again combine the smaller groups to get more statistical power.

```
cluster2data <- subset(
  long_data,
  challenge %in% c("Securing funding", "Finding funding", "Hiring")
)
```

```r
cluster2data <- cluster2data %>%
  mutate(
    job_category = recode(
      job_category,
      "Post-Doc" = "Postdocs and Staff Researchers",
      "Other research staff" = "Postdocs and Staff Researchers"
    )
  )

cluster2data <- cluster2data %>%
  mutate(
    job_category = recode(
      job_category,
      "Grad Student" = "Students",
      "Undergraduate" = "Students"
    )
  )

cluster2data$challenge_level <- factor(cluster2data$challenge_level)

cluster2data
```

```
# A tibble: 699 x 5
   job_category          participantID challenge challenge_level challenge_score
   <chr>                 <chr>         <chr>     <fct>                     <int>
 1 Faculty               1             Hiring    Always                        4
 2 Faculty               1             Finding ~ Always                        4
 3 Faculty               1             Securing~ Always                        4
 4 Postdocs and Staff R~ 2             Hiring    Rarely                        1
 5 Postdocs and Staff R~ 2             Finding ~ Frequently                    3
 6 Postdocs and Staff R~ 2             Securing~ Occasionally                  2
 7 Postdocs and Staff R~ 3             Hiring    Frequently                    3
 8 Postdocs and Staff R~ 3             Finding ~ Always                        4
 9 Postdocs and Staff R~ 3             Securing~ Always                        4
10 Faculty               4             Hiring    Always                        4
# i 689 more rows
```

This code is really similar to code in solutions_stats.qmd. See that notebook for commentary on these models.

Model 1: job_category * challenge interaction

```r
fit1 <- ordinal::clmm(challenge_level ~ job_category * challenge +
              (1 | participantID),
           data = cluster2data, link = "logit", Hess = TRUE)
```

Model 2: challenge as a random effect, no correlation between participant intercept and job effect

```r
fit2 <- ordinal::clmm(challenge_level ~ job_category +
              (1 | challenge) +
              (1 | participantID) +
              (0 + job_category | challenge),
           data = cluster2data, link = "logit", Hess = TRUE)
```

Model 3: No job category

```r
fit3 <- ordinal::clmm(challenge_level ~ challenge +
              (1 | participantID),
           data = cluster2data, link = "logit", Hess = TRUE)
```

Model 4: No challenge category

```r
fit4 <- ordinal::clmm(challenge_level ~ job_category +
              (1 | participantID),
           data = cluster2data, link = "logit", Hess = TRUE)
```

Model 5: job_category + solution

```r
fit5 <- ordinal::clmm(challenge_level ~ job_category + challenge +
              (1 | participantID),
           data = cluster2data, link = "logit", Hess = TRUE)
```

Model 6: no random effects

```
# note clm function bc clmm is for mixed models
fit6 <- ordinal::clm(challenge_level ~ job_category * challenge,
             data = cluster2data, link = "logit", Hess = TRUE)
```

Compare models

```
models <- list(
  "fit1"=fit1, # job_category * challenge
  "fit2"=fit2, # challenge as random effect
  "fit3"=fit3, # Null model: no job
  "fit4"=fit4, # Null model: no challenge
  "fit5"=fit5, # Null model: no interaction
  "fit6"=fit6 # Null model: no participants
)
```

```
sapply(models, function(x) round(stats::AIC(x)))
```

```
fit1 fit2 fit3 fit4 fit5 fit6
2096 2112 2102 2107 2087 2188
```

Models 1 and 5 look best in terms of AIC.

Let's check the condition number of the Hessian. I don't really understand what this is, but the clmm2 tutorial says that high numbers, say larger than say 10^4 or 10^6, indicate poor fit.

```
sapply(models, function(x)
summary(x)$info["cond.H"]
)
```

```
$fit1.cond.H
[1] "9.0e+02"

$fit2.cond.H
[1] "5.2e+02"
```

```
$fit3.cond.H
[1] "1.0e+02"

$fit4.cond.H
[1] "3.6e+02"

$fit5.cond.H
[1] "3.7e+02"

$fit6.cond.H
[1] "1.5e+03"
```

All look ok.

Complex models vs null models

Let's use an anova to compare nested models.

```
stats::anova(fit1, fit5)
```

```
Likelihood ratio tests of cumulative link models:

     formula:                                                        link:
fit5 challenge_level ~ job_category + challenge + (1 | participantID) logit
fit1 challenge_level ~ job_category * challenge + (1 | participantID) logit
     threshold:
fit5 flexible
fit1 flexible

     no.par    AIC  logLik LR.stat df Pr(>Chisq)
fit5     11 2086.8 -1032.4
fit1     17 2095.9 -1031.0  2.8688  6     0.8251
```

Interesting, that p-value is not significant. So it appears the interaction term is not needed.

Let's also double-check that participants are worth including.

```
stats::anova(fit1, fit6)
```

```
Likelihood ratio tests of cumulative link models:

     formula:                                                         link:
fit6 challenge_level ~ job_category * challenge                       logit
fit1 challenge_level ~ job_category * challenge + (1 | participantID) logit
     threshold:
fit6 flexible
fit1 flexible

     no.par    AIC  logLik LR.stat df Pr(>Chisq)
fit6     16 2188.2 -1078.1
fit1     17 2095.9 -1031.0  94.252  1  < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Yes, it appears they are.

Does it matter whether we include job as a variable? Let's compare it to the model without an interaction term.

```r
stats::anova(fit3, fit5)
```

```
Likelihood ratio tests of cumulative link models:

     formula:                                                         link:
fit3 challenge_level ~ challenge + (1 | participantID)                logit
fit5 challenge_level ~ job_category + challenge + (1 | participantID) logit
     threshold:
fit3 flexible
fit5 flexible

     no.par    AIC  logLik LR.stat df Pr(>Chisq)
fit3      8 2102.0 -1043.0
fit5     11 2086.8 -1032.4  21.189  3  9.616e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Yes, including job improves model fit.

So far, fit5 is the one to beat.

More goodness-of-fit tests

SEs of the coefficients

```
summary(fit5$coefficients)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-0.7551  0.5609  0.8817  1.3455  1.4904  4.8889
```

```
summary(fit2$coefficients)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-0.9148  0.6869  1.1423  1.5189  1.9698  4.7287
```

These look pretty similar.

Let's do one more diagnostic. fit6 is the equivalent model to fit1b, but with fixed effects only. Since we can do the nominal_test and scale_test on this model, let's try it and see if it sets off any red flags.

```
nominal_test(fit6)
```

```
Tests of nominal effects

formula: challenge_level ~ job_category * challenge
                      Df  logLik    AIC    LRT  Pr(>Chi)
<none>                   -1078.1 2188.2
job_category          12 -1058.8 2173.5 38.635 0.0001208 ***
challenge              8 -1070.4 2188.8 15.345 0.0527751 .
job_category:challenge
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
scale_test(fit6)
```

```
Tests of scale effects

formula: challenge_level ~ job_category * challenge
                      Df  logLik    AIC    LRT  Pr(>Chi)
<none>                   -1078.1 2188.2
```

```
job_category            3 -1069.8 2177.6 16.552 0.0008736 ***
challenge               2 -1072.3 2180.7 11.463 0.0032428 **
job_category:challenge 11 -1059.4 2172.8 37.383 9.939e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Boo. The model with no random effects has violations to both assumptions.

Ouch. That's not ideal. Maybe we can proceed with caution, and follow up with a non-parametric test on whatever trends we find? https://www.fharrell.com/post/po/

EMMs

Again, see the solutions_stats notebook for more detail on this.

```r
emm <- summary(emmeans(fit5, ~ challenge | job_category, mode = "mean.class"))
emm
```

```
job_category = Faculty:
 challenge        mean.class    SE  df asymp.LCL asymp.UCL
 Finding funding        2.36 0.194 Inf      1.98      2.74
 Hiring                 2.84 0.203 Inf      2.44      3.23
 Securing funding       2.24 0.188 Inf      1.87      2.61

job_category = Non-research Staff:
 challenge        mean.class    SE  df asymp.LCL asymp.UCL
 Finding funding        3.39 0.152 Inf      3.09      3.69
 Hiring                 3.80 0.137 Inf      3.54      4.07
 Securing funding       3.27 0.157 Inf      2.96      3.58

job_category = Postdocs and Staff Researchers:
 challenge        mean.class    SE  df asymp.LCL asymp.UCL
 Finding funding        2.93 0.204 Inf      2.53      3.33
 Hiring                 3.39 0.186 Inf      3.02      3.75
 Securing funding       2.80 0.205 Inf      2.40      3.20

job_category = Students:
 challenge        mean.class    SE  df asymp.LCL asymp.UCL
 Finding funding        3.40 0.230 Inf      2.95      3.85
 Hiring                 3.81 0.205 Inf      3.41      4.21
 Securing funding       3.28 0.237 Inf      2.82      3.75
```

```
Confidence level used: 0.95
```

Pairwise comparisons and p-values

Here we look at pairwise contrasts by challenge.

```r
emm2 <- emmeans(fit5, ~ job_category | challenge, mode = "mean.class")
by_chall <- summary(
  pairs(emm2, by = "challenge"),
  infer = TRUE # infer CIs
)

by_chall
```

```
challenge = Finding funding:
 contrast                                            estimate    SE  df
 Faculty - (Non-research Staff)                      -1.03047 0.227 Inf
 Faculty - Postdocs and Staff Researchers            -0.56513 0.261 Inf
 Faculty - Students                                  -1.03984 0.284 Inf
 (Non-research Staff) - Postdocs and Staff Researchers  0.46535 0.233 Inf
 (Non-research Staff) - Students                     -0.00936 0.258 Inf
 Postdocs and Staff Researchers - Students           -0.47471 0.289 Inf
 asymp.LCL asymp.UCL z.ratio p.value
    -1.613    -0.448  -4.542  <.0001
    -1.237     0.107  -2.162  0.1341
    -1.770    -0.309  -3.657  0.0015
    -0.134     1.065   1.994  0.1900
    -0.673     0.654  -0.036  1.0000
    -1.218     0.269  -1.640  0.3560

challenge = Hiring:
 contrast                                            estimate    SE  df
 Faculty - (Non-research Staff)                      -0.96846 0.225 Inf
 Faculty - Postdocs and Staff Researchers            -0.55074 0.256 Inf
 Faculty - Students                                  -0.97680 0.272 Inf
 (Non-research Staff) - Postdocs and Staff Researchers  0.41772 0.212 Inf
 (Non-research Staff) - Students                     -0.00834 0.230 Inf
 Postdocs and Staff Researchers - Students           -0.42606 0.261 Inf
 asymp.LCL asymp.UCL z.ratio p.value
    -1.546    -0.391  -4.310  0.0001
```

```
    -1.208     0.107  -2.151  0.1371
    -1.675    -0.279  -3.596  0.0018
    -0.128     0.963   1.968  0.2002
    -0.599     0.582  -0.036  1.0000
    -1.098     0.246  -1.630  0.3617

challenge = Securing funding:
 contrast                                              estimate    SE  df
 Faculty - (Non-research Staff)                        -1.03296 0.225 Inf
 Faculty - Postdocs and Staff Researchers              -0.55827 0.259 Inf
 Faculty - Students                                    -1.04261 0.286 Inf
 (Non-research Staff) - Postdocs and Staff Researchers  0.47469 0.237 Inf
 (Non-research Staff) - Students                       -0.00965 0.266 Inf
 Postdocs and Staff Researchers - Students             -0.48434 0.295 Inf
 asymp.LCL asymp.UCL z.ratio p.value
    -1.611    -0.455  -4.589  <.0001
    -1.223     0.106  -2.159  0.1349
    -1.778    -0.307  -3.643  0.0015
    -0.134     1.083   2.005  0.1862
    -0.693     0.674  -0.036  1.0000
    -1.243     0.275  -1.640  0.3563

Confidence level used: 0.95
Conf-level adjustment: tukey method for comparing a family of 4 estimates
P value adjustment: tukey method for comparing a family of 4 estimates
```

Wow, the p-values are really similar across the board. Faculty rate these challenges higher than students and NR staff, but not higher than postdocs and staff researchers.

Kruskal-Wallis test for ranking differences between groups

Non-parametric test for the extent of disagreement between groups. Whereas above, we tested for differences in mean ratings, here we are testing for differences in the distributions of ratings for each solution.

```r
cluster2data_numcoded <- cluster2data %>%
  mutate(
    challenge_score = recode(
      challenge_level,
      "Non-applicable" = 0L,
      "Never" = 0L,
```

```
      "Rarely" = 1L,
      "Occasionally" = 2L,
      "Frequently" = 3L,
      "Always" = 4L
    )
  )

cluster2data_numcoded$job_category <- factor(cluster2data_numcoded$job_category)

kw_results <- sapply(split(cluster2data_numcoded, cluster2data_numcoded$challenge), function
  kruskal.test(challenge_score ~ job_category, data = df)$p.value
})

p_adj_kw <- p.adjust(kw_results, "BH")

p_adj_kw < 0.05
```

```
 Finding funding            Hiring Securing funding
            TRUE              TRUE             TRUE
```

```
sum(p_adj_kw < 0.05)
```

```
[1] 3
```

Ok, so K-W test indicates that for all three challenges, there are differences between the groups.

Dunn test as a post-hoc test to see which groups are different from each other.

```
pairwise_results <- lapply(unique(cluster2data_numcoded$challenge), function(chall) {
  df <- subset(cluster2data_numcoded, challenge == chall)
  out <- FSA::dunnTest(challenge_score ~ job_category, data = df, method = "bh")
  cbind(challenge = chall, out$res)
})
pairwise_results <- do.call(rbind, pairwise_results)
```

Let's print the significant pairs.

```
subset(pairwise_results, P.adj < 0.05)
```

```
          challenge                                          Comparison
1            Hiring                        Faculty - Non-research Staff
2            Hiring            Faculty - Postdocs and Staff Researchers
4            Hiring                                  Faculty - Students
7   Finding funding                        Faculty - Non-research Staff
9   Finding funding Non-research Staff - Postdocs and Staff Researchers
10  Finding funding                                  Faculty - Students
13 Securing funding                        Faculty - Non-research Staff
15 Securing funding Non-research Staff - Postdocs and Staff Researchers
16 Securing funding                                  Faculty - Students
           Z      P.unadj        P.adj
1   3.202369 1.363021e-03 4.089062e-03
2   2.483752 1.300062e-02 2.600123e-02
4   4.191500 2.771168e-05 1.662701e-04
7   4.319614 1.563026e-05 9.378155e-05
9  -2.899242 3.740657e-03 1.122197e-02
10  2.664241 7.716231e-03 1.543246e-02
13  4.741135 2.125239e-06 1.275143e-05
15 -3.077589 2.086828e-03 4.173655e-03
16  3.174243 1.502280e-03 4.506841e-03
```

And the non-significant pairs

```
subset(pairwise_results, P.adj >= 0.05)
```

```
          challenge                                          Comparison
3            Hiring Non-research Staff - Postdocs and Staff Researchers
5            Hiring                       Non-research Staff - Students
6            Hiring           Postdocs and Staff Researchers - Students
8   Finding funding            Faculty - Postdocs and Staff Researchers
11  Finding funding                       Non-research Staff - Students
12  Finding funding           Postdocs and Staff Researchers - Students
14 Securing funding            Faculty - Postdocs and Staff Researchers
17 Securing funding                       Non-research Staff - Students
18 Securing funding           Postdocs and Staff Researchers - Students
            Z    P.unadj      P.adj
3  -0.4391213 0.66057367 0.66057367
5   1.8058145 0.07094732 0.08513678
6   2.0236538 0.04300579 0.06450868
8   1.2252483 0.22048168 0.26457802
11 -0.7377959 0.46063850 0.46063850
12  1.5871988 0.11246762 0.16870143
```

```
14  1.4411382 0.14954564 0.17945476
17 -0.5443851 0.58617651 0.58617651
18  1.9069060 0.05653276 0.08479915
```

Cool. In all three cases, faculty are significantly different from NR staff and students.

```r
sessionInfo()
```

```
R version 4.4.2 (2024-10-31)
Platform: aarch64-apple-darwin20
Running under: macOS 26.1

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRlapack.dylib;

locale:
[1] C.UTF-8/C.UTF-8/C.UTF-8/C/C.UTF-8/C.UTF-8

time zone: America/Los_Angeles
tzcode source: internal

attached base packages:
[1] tools     grid      stats     graphics  grDevices datasets  utils
[8] methods   base

other attached packages:
 [1] treemapify_2.5.6      tidyr_1.3.1           svglite_2.2.1
 [4] stringr_1.5.1         scales_1.4.0          readr_2.1.5
 [7] pwr_1.3-0             patchwork_1.3.2       ordinal_2023.12-4.1
[10] lme4_1.1-37           Matrix_1.7-1          languageserver_0.3.16
[13] here_1.0.1            gtools_3.9.5          ggforce_0.5.0
[16] FSA_0.10.0            fpc_2.2-13            forcats_1.0.0
[19] factoextra_1.0.7      ggplot2_3.5.2         emmeans_1.11.2
[22] dplyr_1.1.4           corrplot_0.95         ComplexHeatmap_2.22.0
[25] cluster_2.1.8.1       BiocManager_1.30.26

loaded via a namespace (and not attached):
 [1] Rdpack_2.6.4        dunn.test_1.3.6     rlang_1.1.6
 [4] magrittr_2.0.3      clue_0.3-66         GetoptLong_1.0.5
 [7] matrixStats_1.5.0   compiler_4.4.2      flexmix_2.3-20
[10] systemfonts_1.2.3   png_0.1-8           callr_3.7.6
```

```
[13] vctrs_0.6.5          pkgconfig_2.0.3     shape_1.4.6.1
[16] crayon_1.5.3         fastmap_1.2.0       labeling_0.4.3
[19] utf8_1.2.6           rmarkdown_2.29      ggfittext_0.10.2
[22] tzdb_0.5.0           ps_1.9.1            nloptr_2.2.1
[25] purrr_1.1.0          xfun_0.53           modeltools_0.2-24
[28] jsonlite_2.0.0       tweenr_2.0.3        parallel_4.4.2
[31] prabclus_2.3-4       R6_2.6.1            stringi_1.8.7
[34] RColorBrewer_1.1-3   boot_1.3-31         diptest_0.77-2
[37] numDeriv_2016.8-1.1 estimability_1.5.1  Rcpp_1.1.0
[40] iterators_1.0.14     knitr_1.50          IRanges_2.40.1
[43] splines_4.4.2        nnet_7.3-19         tidyselect_1.2.1
[46] yaml_2.3.10          doParallel_1.0.17   codetools_0.2-20
[49] processx_3.8.6       lattice_0.22-6      tibble_3.3.0
[52] withr_3.0.2          evaluate_1.0.4      polyclip_1.10-7
[55] xml2_1.4.0           circlize_0.4.16     mclust_6.1.1
[58] kernlab_0.9-33       pillar_1.11.0       renv_1.1.5
[61] foreach_1.5.2        stats4_4.4.2        reformulas_0.4.1
[64] generics_0.1.4       rprojroot_2.1.1     S4Vectors_0.44.0
[67] hms_1.1.3            minqa_1.2.8         xtable_1.8-4
[70] class_7.3-22         glue_1.8.0          robustbase_0.99-4-1
[73] mvtnorm_1.3-3        rbibutils_2.3       colorspace_2.1-1
[76] nlme_3.1-166         cli_3.6.5           textshaping_1.0.1
[79] gtable_0.3.6         DEoptimR_1.1-4      digest_0.6.37
[82] BiocGenerics_0.52.0 ucminf_1.2.2        ggrepel_0.9.6
[85] rjson_0.2.23         farver_2.1.2        htmltools_0.5.8.1
[88] lifecycle_1.0.4      GlobalOptions_0.1.2 MASS_7.3-61
```