QTIFS

2025-11-05

This analysis uses data compiled from a comprehensive literature review of publications addressing queer and trans fieldwork safety in ecology and related disciplines. The dataset (`QTIFS lit review data Nov 2025.xlsx`) includes 61 unique articles identified through structured searches and screening.

Each article was coded for one or more recommendations to improve field safety, categorized by theme (e.g., `Climate`, `Protocols`, `Training`, `Accessibility`) and by implementation scale (e.g., `Individual`, `Lab`, `PI`, `Department`, `Institution`, `Outside Institution`). Additional coding captured whether authors described a specific plan for implementation or provided evidence of real-world testing.

The following analyses summarize the distribution of these recommendations, visualize their thematic and structural patterns, and quantify the degree to which certain categories and scales are over- or underrepresented among actionable implementation plans.

INTRO

Attach packages

```
library(tidyverse)   # includes dplyr, ggplot2, readr, forcats, etc.
library(readxl)      # for reading Excel files if used elsewhere
library(janitor)     # for data cleaning and column name formatting
library(ggpubr)      # for ggplot helpers and statistical annotations
library(patchwork)   # for combining multiple ggplots
library(viridis)     # for color palettes
library(ggalluvial)  # for alluvial/flow diagrams
library(scales)      # for axis scaling and labels
```

Load Data

```
q <- read_xlsx("QTIFS lit review data Nov 2025.xlsx") %>%
  janitor::clean_names() %>%
  dplyr::filter(!is.na(recommendation_categories) & !is.na(recommendation_scale)) %>%
  mutate(
    recommendation_categories = fct_relevel(recommendation_categories,
                                            c("Climate",
                                              "Protocols",
                                              "Training",
                                              "Accessibility")),
    recommendation_scale = fct_relevel(recommendation_scale, c(
      "Outside Institution",
      "Institution",
      "Department",
```

```
      "Lab", "PI",
      "Individual"
    ))
  )

arts <- q %>% dplyr::group_by(as.factor(recommendation_scale)) %>%
  dplyr::summarize(n())

sum(arts$`n()`)
```

```
## [1] 165
```

```
print(arts)
```

```
## # A tibble: 6 x 2
##   `as.factor(recommendation_scale)` `n()`
##   <fct>                             <int>
## 1 Outside Institution                  18
## 2 Institution                          42
## 3 Department                           16
## 4 Lab                                  11
## 5 PI                                   50
## 6 Individual                           28
```

```
q %>% dplyr::group_by(as.factor(recommendation_scale)) %>% dplyr::summarize(n()/nrow(q))
```

```
## # A tibble: 6 x 2
##   `as.factor(recommendation_scale)` `n()/nrow(q)`
##   <fct>                                    <dbl>
## 1 Outside Institution                     0.109
## 2 Institution                             0.255
## 3 Department                              0.0970
## 4 Lab                                     0.0667
## 5 PI                                      0.303
## 6 Individual                              0.170
```

Set color palette

```
# define palette
my_colors <- c("#003f5c", "#58508d", "#bc5090", "#ff6361", "#ffa600")
```

Heatmap

Visualize the frequency of recommendations across thematic categories and implementation scales.
wrangle

```r
# summarize number of recommendations per category and scale
q_collect <- q %>%
  select(recommendation_categories, recommendation_scale) %>%
  group_by(recommendation_categories, recommendation_scale, .drop = FALSE) %>%
  dplyr::summarize(
    num = n()
  ) %>%
  ungroup() %>%
  na.omit()
```

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

counts

```r
# count unique recommendations by category
q_unique_rec <- q %>%
  select(article, one_sentence_summary, recommendation_categories, recommendation_scale) %>%
  distinct() %>%
  count(recommendation_categories, name = "num")

# count unique combinations of category and scale
q_heatmap <- q %>%
  select(article, one_sentence_summary, recommendation_categories, recommendation_scale, recommendation
  distinct() %>%
  count(recommendation_categories, recommendation_scale, name = "num")

# count unique articles and recommendations to use in plots later
q_unique_articles <- length(unique(q$article))

q_unique_recs_num <- q %>%
  select(article, recommendation_categories, recommendation_scale) %>%
  distinct() %>%
  summarize(n())
```

PLOT

```r
sum_q <- sum(q_heatmap$num)

heatmap <- ggplot(q_heatmap, aes(x = recommendation_scale, y = recommendation_categories, size = num, fi
  annotate("rect", xmin = 5.5, xmax = Inf, ymin = 0, ymax = Inf, fill = 'gray', alpha = 0.4) +
  geom_vline(xintercept = 5.5, color = "gray", linewidth = 0.75, linetype = "dashed") +
  geom_point(pch = 21) +
  theme_bw() +
  theme(axis.text.y = element_text(face = "italic")) +
  scale_fill_viridis(option = "viridis", name = "Number of\nStudies",
                     breaks = c(1, 3, 9, 15, 21)
                     ) +
  scale_size_continuous(range = c(1,15),
```

```
                        name = "Number of\nStudies",
                        breaks = c(1, 3, 9, 15, 21)
                        ) +

  scale_x_discrete(limits = rev) +
  scale_y_discrete(limits = rev) +

  theme(axis.line.x = element_line(linewidth = 1.5,
                                   color = "grey50",
                                   arrow = grid::arrow(length = unit(0.4, "cm"),
                                                       ends = "last")),
        axis.text.x = element_text(angle = 45, hjust = 1, face = c("plain", "plain", "plain", "plain",
        legend.key.size = unit(1, "cm")
        ) +
  guides(fill = guide_legend(), size = guide_legend()) +
  labs(title = "Distribution of recommendation categories and scales", x = "Scale", y = "Categories", su
```

```
## Warning: Vectorized input to `element_text()` is not officially supported.
## i Results may be unexpected or may change in future versions of ggplot2.
```

```
# heatmap
# ggsave("Heatmap.png", width = 7, height = 5)
```

Bar chart

wrangle

```
# define category levels and matching color palette
category_levels <- levels(q_collect$recommendation_categories)

# create a named vector for consistent color mapping
category_palette <- setNames(my_colors[1:length(category_levels)], category_levels)

# summarize total counts for each category
q_category_totals <- q_collect %>%
  group_by(recommendation_categories) %>%
  summarise(total = sum(num)) %>%
  ungroup()
```

PLOT

```
sum_collect <- sum(q_collect$num)

themes <- ggplot(q_collect, aes(x = recommendation_categories, y = num, fill = recommendation_categories
  geom_bar(stat = "identity") +
  geom_text(
    data = q_category_totals,
    aes(x = recommendation_categories, y = total, label = paste0("N = ", total)),
```

```
    vjust = -1,
    size = 3,
    inherit.aes = FALSE
  ) +
  scale_fill_manual(values = category_palette) +
  labs(x = "Recommendation Category", y = "Number of Unique Recommendations",
       title = "Main Themes Across Articles",
       subtitle = paste0("Each article has  1 recommendations;\nN = ", sum_collect ," recommendations ac
  ylim(c(0, 90)) +
  theme_minimal() +
  theme(legend.position = "none",
        panel.grid.major.x = element_blank(),
        axis.text.x = element_text(angle = 45, hjust = 0.75, size = 10),
        panel.background = element_rect(fill = "white", color = NA),
    plot.background = element_rect(fill = "white", color = NA))

# themes
# ggsave("Recommendation Categories Bar Chart.png", width = 6, height = 4)
```

Recommendation plans

Does each article included an explicit plan for implementing recommendations? ### wrangle

```
# count unique articles with their recommendation plan Y/N
q_plan_status <- q %>%
  filter(!is.na(recommendation_categories) & !is.na(recommendation_scale)) %>%
  select(article, plan_for_implementation_y_n) %>%
  distinct() %>%
  count(plan_for_implementation_y_n, name = "num")

# total counts per plan status
q_plan_totals <- q_plan_status %>%
  group_by(plan_for_implementation_y_n) %>%
  summarise(total = sum(num)) %>%
  ungroup()

# sanity check: check for inconsistencies in plan coding across same articles
inconsistent_plan_check <- q %>%
  select(article, plan_for_implementation_y_n) %>%
  distinct() %>%
  group_by(article) %>%
  summarise(unique_vals = n_distinct(plan_for_implementation_y_n)) %>%
  filter(unique_vals > 1)

# print any inconsistencies: articles that were coded both yes and no
inconsistent_plan_check
```

```
## # A tibble: 0 x 2
## # i 2 variables: article <chr>, unique_vals <int>
```

```
q_unique_plans <- sum(q_plan_status$num)
```

PLOT

```
plan <- ggplot(q_plan_status, aes(x = plan_for_implementation_y_n, y = num, fill = plan_for_implementati
  geom_bar(stat = "identity") +
  geom_text(
    data = q_plan_totals,
    aes(x = plan_for_implementation_y_n, y = total, label = paste0("N = ", total)),
    vjust = -1,
    size = 3,
    inherit.aes = FALSE
  ) +
  scale_fill_manual(values = c(my_colors[1], my_colors[3])) +
  labs(
    x = "Stated Plan for Implementation",
    y = "Number of Articles",
    title = "Did Articles Include a Plan for Implementation?",
    subtitle = paste0("N = ", q_unique_plans, " articles")
  ) +
  expand_limits(y = c(0, 60)) +
  scale_x_discrete(breaks = c("N", "Y"), labels = c("No", "Yes")) +
  theme_minimal() +
  theme(
    legend.position = "none",
    panel.grid.major.x = element_blank(),
    axis.text.x = element_text(hjust = 1, size = 10),
    panel.background = element_rect(fill = "white", color = NA),
    plot.background = element_rect(fill = "white", color = NA)
  )

# plan
# ggsave("Plan Inclusion Bar Chart.png", width = 5, height = 5)
```

Real-world testing

Were recommendations explicitly tested or applied in real-world settings?
wrangle

```
# count unique articles that tested recommendations in the real world
q_real_world_test <- q %>%
  select(article, real_world_implementation_testing_y_n) %>%
  distinct() %>%
  count(real_world_implementation_testing_y_n, name = "num")

# compute total number of articles per response
q_real_totals <- q_real_world_test %>%
  group_by(real_world_implementation_testing_y_n) %>%
  summarise(total = sum(num)) %>%
  ungroup()
```

```r
# check for inconsistencies in coding for real-world testing
realworld <- q %>% dplyr::select(article, recommendation_categories, recommendation_scale, real_world_i

# identify any citations where multiple values exist
inconsistent_test <- realworld %>%
  group_by(article) %>%
  summarise(unique_vals = n_distinct(real_world_implementation_testing_y_n)) %>%
  filter(unique_vals > 1)

# show the inconsistent citations (if any)
inconsistent_test
```

```
## # A tibble: 0 x 2
## # i 2 variables: article <chr>, unique_vals <int>
```

```r
q_real_world_test <- q %>%
  select(article, real_world_implementation_testing_y_n) %>%
  distinct() %>%
  count(real_world_implementation_testing_y_n, name = "num")
```

PLOT

```r
testing <- ggplot(q_real_world_test, aes(x = real_world_implementation_testing_y_n, y = num, fill = rea
  geom_bar(stat = "identity") +
  geom_text(
    data = q_real_totals,
    aes(x = real_world_implementation_testing_y_n, y = total, label = paste0("N = ", total)),
    vjust = -1,
    size = 3,
    inherit.aes = FALSE
  ) +
  scale_fill_manual(values = c(my_colors[3], my_colors[1])) +
  labs(x = "Real World Testing of Recommendations", y = "Number of Articles",
       title = "Were Recommendations Tested in the Real-World?",
       subtitle = paste0("N = ", q_unique_articles, " articles")) +
  expand_limits(y = c(0, 50)) +
  scale_x_discrete(breaks = c("N", "Y"), labels = c("No", "Yes")) +
  theme_minimal() +
  theme(legend.position = "none",
        panel.grid.major.x = element_blank(),
        axis.text.x = element_text(hjust = 1, size = 10),
        panel.background = element_rect(fill = "white", color = NA),
    plot.background = element_rect(fill = "white", color = NA))

# testing
# ggsave("Real World Testing Bar Chart.png", width = 5, height = 5)
```

Implementation Responsibility

Identify where the literature places responsibility for implementing recommendations.
wrangle

```
# count how many recommendations were assigned to each implementation scale
q_responsibility <- q %>%
  select(article, recommendation_categories, recommendation_scale) %>%
  distinct() %>%
  count(recommendation_scale, name = "num")

sum(q_responsibility$num)
```

```
## [1] 165
```

```
q %>% group_by(article, recommendation_scale) %>% dplyr::summarize(n())
```

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

```
## # A tibble: 146 x 3
## # Groups:   article [61]
##    article                   recommendation_scale `n()`
##    <chr>                     <fct>                <int>
##  1 Atchinson 2021            Outside Institution      1
##  2 Atchinson 2021            Institution              1
##  3 Baker 2000                Outside Institution      1
##  4 Baker 2000                Institution              1
##  5 Beltran et al 2021        Institution              1
##  6 Beltran et al 2021        Department               1
##  7 Blonder 2022              PI                       3
##  8 Bowser & Cid 2021         Institution              1
##  9 Bowser & Cid 2021         PI                       1
## 10 Bracken and Mawdsley 2004 Individual               1
## # i 136 more rows
```

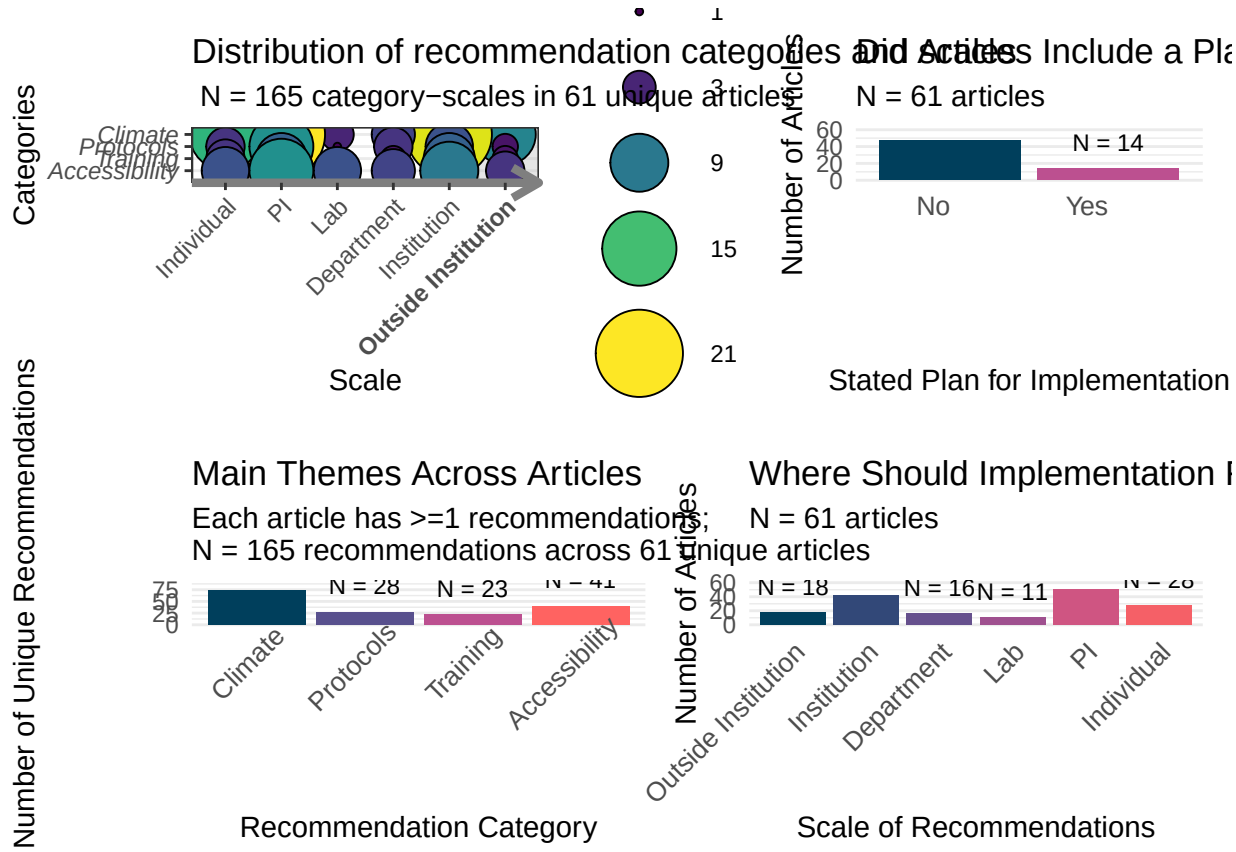
PLOT

```
pal <- colorRampPalette(my_colors[1:5])

responsibility <- ggplot(q_responsibility, aes(x = recommendation_scale, y = num, fill = recommendation_
  geom_bar(stat = "identity") +
  geom_text(
    aes(x = recommendation_scale, y = num, label = paste0("N = ", num)), vjust = -1, size = 3) +
  scale_fill_manual(values = pal(8)) +
  labs(x = "Scale of Recommendations", y = "Number of Articles",
       title = "Where Should Implementation Fall?",
       subtitle = paste0("N = ", q_unique_articles, " articles")) +
  expand_limits(y = c(0, 60)) +
  theme_minimal() +
```

```
  theme(legend.position = "none",
        panel.grid.major.x = element_blank(),
        axis.text.x = element_text(angle = 45, hjust = 1, size = 10),
        panel.background = element_rect(fill = "white", color = NA),
    plot.background = element_rect(fill = "white", color = NA))

# responsibility
# ggsave("Implementation Level.png", width = 6, height = 4)
```

COMBINED PLOTS

```
(heatmap + plan) / plot_spacer() / (themes + responsibility) +
  plot_layout(heights = c(3, 0.25, 2.5))
```



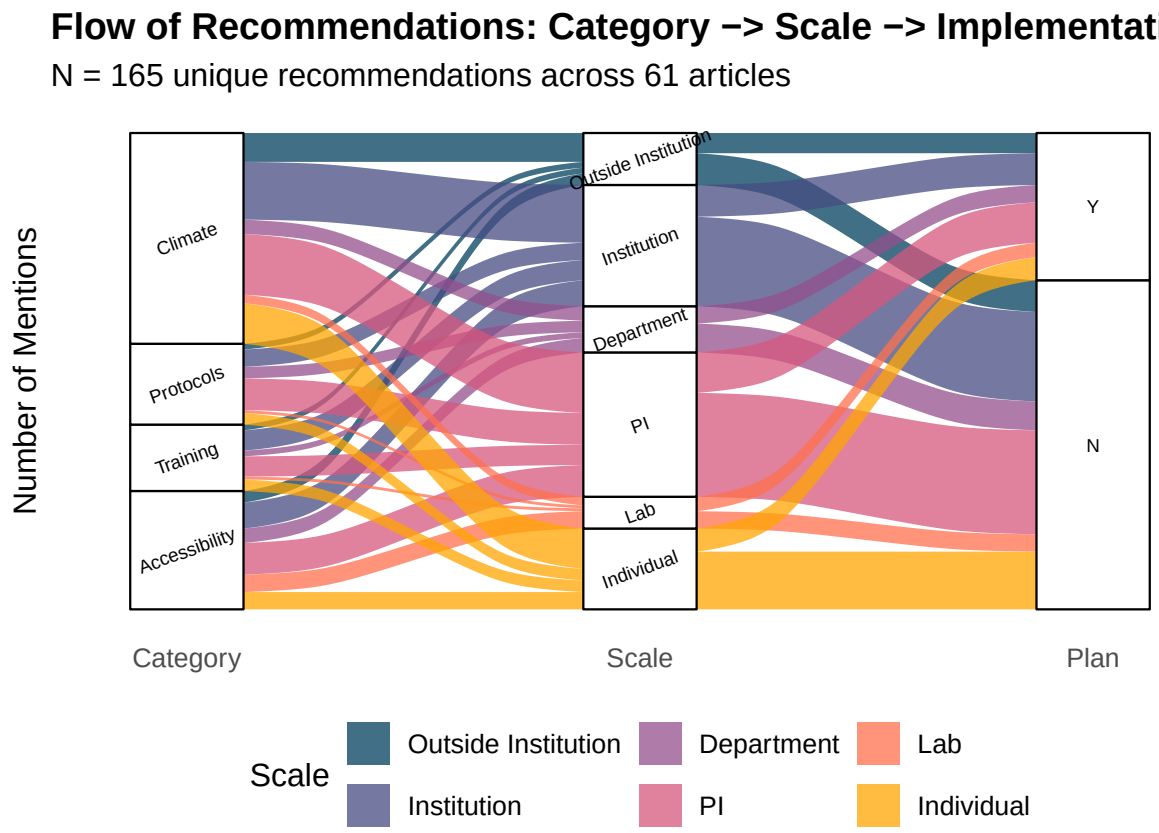

```
# ggsave("combined_plots.png", height = 10, width = 12)
```

Sankey

PLOT

```
pal6 <- as.list(pal(6))
pal6 <- unlist(pal6)

# Prepare alluvial data
alluvial_data <- q %>%
  select(
    category = recommendation_categories,
    scale = recommendation_scale,
    plan = plan_for_implementation_y_n
  ) %>%
  na.omit() %>%
  mutate(
    category = fct_relevel(category, c("Climate", "Protocols", "Training",  "Accessibility")),
    scale = factor(scale, levels = c("Outside Institution", "Institution", "Department", "PI", "Lab", "I
    plan = factor(plan, levels = c("Y", "N"))
  ) %>%
  count(category, scale, plan, name = "n") %>%
  mutate(alluvium_id = paste(category, scale, sep = " → "))


# assign color palette to scales instead of categories
scale_levels <- levels(alluvial_data$scale)
scale_palette <- setNames(pal6[1:length(scale_levels)], scale_levels)


# plot with wider strata and custom colors
ggplot(alluvial_data,
       aes(axis1 = category, axis2 = scale, axis3 = plan, y = n)) +
  geom_flow(aes(fill = scale), width = 0.25, alpha = 0.75) +
  geom_stratum(width = 0.25, lwd = 0.4, color = "black") +
  geom_text(
    stat = "stratum",
    aes(
      label = after_stat(stratum),
      angle = dplyr::case_when(
        after_stat(x) %in% c(1, 2) ~ 20,
        TRUE ~ 0
      ),
      hjust = 0.5
    ),
    size = 2.5
  ) +
  scale_x_discrete(limits = c("Category", "Scale", "Plan"), expand = c(0.1, 0.1)) +
  scale_fill_manual(values = scale_palette) +
  labs(
    title = "Flow of Recommendations: Category → Scale → Implementation Plan",
    y = "Number of Mentions",
    x = NULL,
    fill = "Scale",
```

```
  subtitle = paste0("N = ", q_unique_recs_num, " unique recommendations across ", q_unique_articles,
) +
theme_minimal(base_size = 12) +
theme(
  legend.position = "bottom",
  plot.title = element_text(size = 14, face = "bold"),
  axis.text.y = element_blank(),
  axis.ticks.y = element_blank(),
  panel.grid = element_blank(),
  panel.background = element_rect(fill = "white", color = NA),
  plot.background = element_rect(fill = "white", color = NA)
) +
guides(fill = guide_legend(override.aes = list(size = 1)))
```




```
# ggsave("flow_of_recs.png", width = 8, height = 6)
```

Disproportionality

The disproportionality analysis tests whether certain recommendation categories or scales were over- or under-represented among those with stated implementation plans.
Calculate Skew - CATEGORY

```r
# calculate overall baseline proportions of categories across all recommendations
category_proportions <- q %>%
  count(recommendation_categories, name = "total_n") %>%
  mutate(category_total_prop = total_n / sum(total_n)) %>%
  select(recommendation_categories, category_total_prop)

# calculate proportions within each plan group (Y/N)
category_plan_distribution <- q %>%
  mutate(plan = factor(plan_for_implementation_y_n, levels = c("Y", "N"))) %>%
  count(recommendation_categories, plan, name = "plan_n") %>%
  group_by(plan) %>%
  mutate(plan_total = sum(plan_n),
         category_within_plan_prop = plan_n / plan_total) %>%
  ungroup()

# join proportions and calculate difference (disproportionality)
comparison_by_category <- category_plan_distribution %>%
  left_join(category_proportions, by = "recommendation_categories") %>%
  mutate(disproportionality = category_within_plan_prop - category_total_prop) %>%
  arrange(recommendation_categories, plan)

# format as percentages
comparison_by_category_formatted <- comparison_by_category %>%
  mutate(across(
    c(category_total_prop, category_within_plan_prop, disproportionality),
    ~ percent(.x, accuracy = 0.1)
  ))

comparison_by_category_formatted
```

```
## # A tibble: 8 x 7
##   recommendation_categories plan  plan_n plan_total category_within_plan_prop
##   <fct>                     <fct>  <int>      <int> <chr>
## 1 Climate                   Y         18         51 35.3%
## 2 Climate                   N         55        114 48.2%
## 3 Protocols                 Y         14         51 27.5%
## 4 Protocols                 N         14        114 12.3%
## 5 Training                  Y          8         51 15.7%
## 6 Training                  N         15        114 13.2%
## 7 Accessibility             Y         11         51 21.6%
## 8 Accessibility             N         30        114 26.3%
## # i 2 more variables: category_total_prop <chr>, disproportionality <chr>
```

```r
write_csv(comparison_by_category_formatted, "comparison_ordered_category Nov 2025.csv")
```

Calculate Skew - SCALE

```r
# factor order
scale_order <- c("Individual", "Lab", "PI", "Department", "Institution", "Outside Institution")

# calculate compute overall baseline proportions by scale
```

```r
scale_proportions <- q %>%
  mutate(recommendation_scale = factor(recommendation_scale, levels = scale_order)) %>%
  count(recommendation_scale) %>%
  mutate(scale_total_prop = n / sum(n)) %>%
  select(recommendation_scale, scale_total_prop)

# calculate within-plan proportions by scale
scale_plan_distribution <- q %>%
  mutate(
    recommendation_scale = factor(recommendation_scale, levels = scale_order),
    plan = factor(plan_for_implementation_y_n, levels = c("Y", "N"))
  ) %>%
  count(recommendation_scale, plan) %>%
  group_by(plan) %>%
  mutate(plan_total = sum(n),
         scale_within_plan_prop = n / plan_total) %>%
  ungroup()

# join and calculate disproportionality between within-plan and overall proportions
comparison_ordered <- scale_plan_distribution %>%
  left_join(scale_proportions, by = "recommendation_scale") %>%
  mutate(disproportionality = scale_within_plan_prop - scale_total_prop) %>%
  arrange(recommendation_scale, plan) %>%
  mutate(across(
    c(scale_total_prop, scale_within_plan_prop, disproportionality),
    percent_format(accuracy = 0.1)
  ))

comparison_ordered
```

```
## # A tibble: 12 x 7
##    recommendation_scale plan      n plan_total scale_within_plan_prop
##    <fct>                <fct> <int>      <int> <chr>
##  1 Individual           Y         8         51 15.7%
##  2 Individual           N        20        114 17.5%
##  3 Lab                  Y         5         51 9.8%
##  4 Lab                  N         6        114 5.3%
##  5 PI                   Y        14         51 27.5%
##  6 PI                   N        36        114 31.6%
##  7 Department           Y         6         51 11.8%
##  8 Department           N        10        114 8.8%
##  9 Institution          Y        11         51 21.6%
## 10 Institution          N        31        114 27.2%
## 11 Outside Institution  Y         7         51 13.7%
## 12 Outside Institution  N        11        114 9.6%
## # i 2 more variables: scale_total_prop <chr>, disproportionality <chr>
```

```r
write_csv(comparison_ordered, "comparison_ordered_scale Nov 2025.csv")
```