Colombian trans wellbeing

Code and analyses

Maria Fernanda Reyes-Rodríguez [1,*] Juan David Leongómez [2]

11 March, 2025

[1] Department of Psychology, University of Los Andes, Bogota 111211, Colombia
[2] CODEC: Ciencias Cognitivas y del Comportamiento, Universidad El Bosque, Bogotá 110121, Colombia.

[*] Correspondence: m.reyes8@uniandes.edu.co

Description

This document contains all code, and step by step explanations for all analyses, figures and tables (including supplementary figures and tables) for:

Reyes-Rodríguez, M. F., & Leongómez, J. D. (in prep). *Colombian trans wellbeing*

Data are available on the Open Science Framework (OSF): https://doi.org/10.XXXX/OSF.IO/XXXXX. The analyses were designed by Maria Fernanda Reyes-Rodríguez and Juan David Leongómez. This document and its underlying code were created in R Markdown by Juan David Leongómez using R and LaTeX, ensuring full reproducibility.

Contents

1 Preliminaries

1.1 Load packages

This file was created using `knitr` (Xie, 2014), mostly using `tidyverse` (Wickham et al., 2019) syntax. As such, data wrangling was mainly done using packages such as `dplyr` (Wickham et al., 2023), and most figures were created or modified using `ggplot2` (Wickham, 2016). Tables were created using `knitr::kable` and `kableExtra` (Zhu, 2021).

Multi-model inference and model averaging was achieved using `MuMIn` (Bartoń, 2024), and model assumptions were performed using `performance` (Lüdecke et al., 2021).

All packages used in this file can be directly installed from the Comprehensive R Archive Network (CRAN). For a complete list of packages used to create this file, and their versions, see section 4, at the end of the document.

```r
library(ltm)
library(psych)        # For statistical functions (e.g., Cronbach's alpha)
library(MuMIn)        # For model selection and averaging
library(performance)  # For model performance metrics
library(readr)        # For reading data files
library(scales)       # For percent formatting
library(knitr)
library(kableExtra)
library(car)
library(tidyverse)    # For data manipulation and piping
```

1.2 Custom functions

1.2.1 `pval.lev` and `pval.stars`

These functions take p-values and formats them, either in LaTeXand highlighting significant p-values in bold and representing all in an appropriate level, or as stars.

```r
# Function to format p-values for LaTeX output, highlighting significant values in bold
pval.lev <- function(pvals) {
  ifelse(pvals < 0.0001, "\\textbf{< 0.0001}", # Highlight very small p-values
         ifelse(pvals < 0.001, "\\textbf{< 0.001}", # Bold p-values < 0.001
                ifelse(pvals < 0.05, paste0("\\textbf{", round(pvals, 4), "}"), # Bold p-values < 0.05
                       round(pvals, 2) # Round non-significant values to two decimal places
                )
         )
  )
}

# Function to add significance stars based on p-value thresholds
pval.stars <- function(pvals) {
  ifelse(pvals < 0.0001, "****", # Four stars for p < 0.0001
         ifelse(pvals < 0.001, "***", # Three stars for p < 0.001
                ifelse(pvals < 0.01, "**", # Two stars for p < 0.01
                       ifelse(pvals < 0.05, "*", NA) # One star for p < 0.05, NA otherwise
                )
```

```
        )
  )
}
```

1.2.2 avg.model.anova

XXXX

```
avg.model.anova <- function(avg_model, data, response) {
  # Extract predictor names (remove intercept)
  selected_vars <- names(coef(avg_model))[-1]

  # Ensure selected_vars exist in the dataset
  selected_vars <- selected_vars[selected_vars %in% names(data)]

  if (length(selected_vars) == 0) {
    stop("No valid predictors found in model-averaged object.")
  }

  # Refit model using selected (averaged) predictors
  weighted_model <- lm(reformulate(selected_vars, response = response), data = data)

  # Compute Type III ANOVA
  anova_table <- weighted_model |>
    Anova(type = "III") |>
    broom::tidy() |>
    mutate_at("term", str_replace_all, "_", " ") |>
    mutate(df = paste(df, weighted_model$df.residual, sep = ", "),
           p.value = pval.lev(p.value)) |>
    filter(term != "Residuals") |> # Remove Residuals row
    kable(digits = 3,
          booktabs = TRUE,
          linesep = "",
          align = c("l", rep("c", 4)),
          caption = "XXXXXX",
          col.names = c("Term", "$SS_{term}$", "$df$", "$F$", "$p$"),
          escape = FALSE) |>
    kable_styling(latex_options = "HOLD_position") |>
    footnote(
      general = "This ANOVA table was generated based on model-averaged estimates from
      multimodel inference. The predictor terms included in the model were selected based on
      their relative importance across candidate models ($\\\\Delta AICc$ < 2).
      Sum of squares ($SS_{term}$) values correspond to Type III ANOVA calculations,
      which test each term's contribution while controlling for all other predictors.
      Degrees of freedom ($df$) are presented as term $df$ and residual $df$, where residual
      $df$ reflects the remaining degrees of freedom in the model. The $F$ and $p$ values were
      computed from the refitted model using only the selected predictors.
      Significant effects are in bold.",
      threeparttable = TRUE,
      footnote_as_chunk = TRUE,
      escape = FALSE
    )
  return(anova_table)
}
```

1.2.3 `avg.mod.plot`

XXXX

```r
avg.mod.plot <- function(avg_mod) {
  # Extract model summary and transform into a tidy format
  x <- summary(avg_mod)$coefmat.full |>
  as_tibble(rownames = "key") |>  # Convert row names to a "key" column
  bind_cols(
    confint(avg_mod, full = TRUE) |> as_tibble(),  # Add confidence intervals
    summary(avg_mod)$coef.nmod |>
      as_tibble() |>
      pivot_longer(cols = everything(), names_to = "model", values_to = "value")  # Gather number of model
  ) |>
  mutate(
    avmod = deparse(substitute(avg_mod)) |>
      factor(),  # Store model name as a factor
    value = value / max(value, na.rm = TRUE),  # Normalize 'value' column
    sig = pval.stars(`Pr(>|z|)`) |>
      str_replace("\\.", "†"),  # Convert p-values into significance stars
    key = key |>
      str_replace_all("Gender", "Gender: ") |>
      str_replace_all("Housing", "Housing: ") |>
      str_replace_all("_", " "))

  x <- x |>
    mutate(key = factor(key, levels = as.character(unique(x$key))))

  # Get the number of averaged models
  nMods <- dim(avg_mod$msTable)[1]

  # Create the plot
  ggplot(x, aes(x = key, y = Estimate)) +
    # Add horizontal reference line at zero
    geom_hline(yintercept = 0, color = "grey") +
    # Add points sized and colored by importance
    geom_point(aes(size = value, color = value), alpha = 0.5) +
    # Add error bars (confidence intervals)
    geom_errorbar(aes(ymin = `2.5 %`, ymax = `97.5 %`),
                  colour = "black", width = 0.1) +
    # Add additional points for emphasis
    geom_point(size = 1) +
    # Apply theme and labels
    theme_bw() +
    labs(x = NULL, y = "Estimate") +
    theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
    # Scale for importance (value) size
    scale_size_continuous(range = c(2, 8),
                          breaks = seq(0, 1, by = 0.2)) +
    # Legends for size and color
    guides(size = guide_legend(title = "Importance"),
           color = guide_legend(title = "Importance")) +
    # Ensure x-axis labels remain in the correct order
    scale_x_discrete(labels = levels(x$key),
                     expand = c(0, 0.5)) +
    # Use plasma color scale, reversed so darker colors represent higher values
    scale_colour_viridis_c(option = "plasma", direction = -1) +
    # Add significance labels next to points
    geom_text(aes(label = sig), y = x$`97.5 %`, vjust = -0.4) +
    # Add annotation indicating the number of averaged models
    geom_text(aes(x = Inf, y = -Inf,
                  label = paste("Models averaged = ", nMods)),
              #fontface = "italic",
              size = 3,
              hjust = 1.1,
              vjust = -0.5,
              inherit.aes = FALSE)
}
```

1.3 Set sum-to-zero contrasts for factors (needed for Type III ANOVA)

```
options(contrasts = c("contr.sum", "contr.poly"))
```

1.4 Load data

Load raw CSV data

```
data_RAW <- read_csv("data/data.csv")
```

1.4.1 Define PANAS Subscales (Positive & Negative Affect)

XXXXXXX

```
# List of PANAS Positive Affect (PANAS_P) items
PANAS_P <- c("PANASB_1", "PANASB_3", "PANASB_5", "PANASB_9",
             "PANASB_10", "PANASB_12", "PANASB_14", "PANASB_16",
             "PANASB_17", "PANASB_19")

# List of PANAS Negative Affect (PANAS_N) items
PANAS_N <- c("PANASB_2", "PANASB_4", "PANASB_6", "PANASB_7",
             "PANASB_8", "PANASB_11", "PANASB_13", "PANASB_15",
             "PANASB_18", "PANASB_20")
```

1.5 Internal consistency

1.5.1 Calculate Cronbach's Alpha for Different Scales

To measure the internal consistency of these tests, we used standardized Cronbach's alpha (α or Tau-equivalent reliability: ρ_T) coefficients, using the function `cronbach.alpha` from the package `ltm` (Rizopoulos, 2006).

```
# Compute Cronbach's alpha for the Self-Efficacy (EAG) scale
alpha_EAG <- data_RAW |>
  mutate(across(where(is.numeric), ~ na_if(., 99))) |>  # Replace 99 with NA (missing values)
  select(starts_with("EAG_")) |>  # Select all columns starting with "EAG_"
  drop_na() |>
  cronbach.alpha(CI = TRUE, standardized = TRUE)  # Compute Cronbach’s alpha

# Compute Cronbach's alpha for the Life-Satisfaction (SWLS) scale
alpha_SWLS <- data_RAW |>
  mutate(across(where(is.numeric), ~ na_if(., 99))) |>  # Replace 99 with NA
  select(starts_with("SWLS_")) |>  # Select all columns starting with "SWLS_"
  drop_na() |>
  cronbach.alpha(CI = TRUE, standardized = TRUE)
```

```r
# Compute Cronbach's alpha for the Resilience (EBR) scale
alpha_EBR <- data_RAW |>
  mutate(across(where(is.numeric), ~ na_if(., 99))) |>  # Replace 99 with NA
  select(starts_with("EBR_")) |>  # Select all columns starting with "EBR_"
  drop_na() |>
  cronbach.alpha(CI = TRUE, standardized = TRUE)

# Compute Cronbach's alpha for the Depression (EBD) scale (after recoding responses)
alpha_EBD <- data_RAW |>
  mutate(across(where(is.numeric), ~ na_if(., 99))) |>  # Replace 99 with NA
  select(starts_with("EBD_")) |>  # Select all columns starting with "EBD_"
  mutate(across(everything(), ~ ifelse(is.na(.x), NA, .x - 1))) |>  # Adjust values
  drop_na() |>
  cronbach.alpha(CI = TRUE, standardized = TRUE)

# Compute Cronbach's alpha for the Social Support (MOS2) scale
alpha_MOS2 <- data_RAW |>
  mutate(across(where(is.numeric), ~ na_if(., 99))) |>  # Replace 99 with NA
  select(starts_with("MOS2_")) |>  # Select all columns starting with "MOS2_"
  drop_na() |>
  cronbach.alpha(CI = TRUE, standardized = TRUE)

# Compute Cronbach's alpha for PANAS Positive Affect (PANAS_P)
alpha_PANAS_P <- data_RAW |>
  mutate(across(where(is.numeric), ~ na_if(., 99))) |>  # Replace 99 with NA
  select(all_of(PANAS_P)) |>  # Select PANAS_P variables
  drop_na() |>
  cronbach.alpha(CI = TRUE, standardized = TRUE)

# Compute Cronbach's alpha for PANAS Negative Affect (PANAS_N)
alpha_PANAS_N <- data_RAW |>
  mutate(across(where(is.numeric), ~ na_if(., 99))) |>  # Replace 99 with NA
  select(all_of(PANAS_N)) |>  # Select PANAS_N variables
  drop_na() |>
  cronbach.alpha(CI = TRUE, standardized = TRUE)

# Compute Cronbach's alpha for Community Cohesion (PCPS3) scale
alpha_PCPS3 <- data_RAW |>
  mutate(across(where(is.numeric), ~ na_if(., 99))) |>  # Replace 99 with NA
  select(starts_with("PCPS3_")) |>  # Select all columns starting with "PCPS3_"
  drop_na() |>
  cronbach.alpha(CI = TRUE, standardized = TRUE)
```

1.5.2 Table S1. Internal consistency of measured scales

The internal consistency of the measured scales was generally strong, with Cronbach's α values ranging from 0.767 to 0.977. In particular, the Social Support (MOS2) and Self-Efficacy (EAG) scales exhibited excellent internal consistency, while the Depression (EBD) and Community Cohesion (PCPS3) scales had acceptable reliability, suggesting a slightly lower but still adequate level of internal consistency.

```r
tibble(
  Scale = c("Self-Efficacy$^1$",
            "Life-Satisfaction$^2$",
            "Resilience$^3$",
            "Depression$^4$",
            "Social Support$^5$",
            "PANAS Positive$^6$",
            "PANAS Negative$^6$",
            "Community Cohesion$^x$"),
  p = c(alpha_EAG$p, alpha_SWLS$p, alpha_EBR$p, alpha_EBD$p, alpha_MOS2$p,
        alpha_PANAS_P$p, alpha_PANAS_N$p, alpha_PCPS3$p),
  n = c(alpha_EAG$n, alpha_SWLS$n, alpha_EBR$n, alpha_EBD$n, alpha_MOS2$n,
        alpha_PANAS_P$n, alpha_PANAS_N$n, alpha_PCPS3$n),
  alpha = c(alpha_EAG$alpha, alpha_SWLS$alpha, alpha_EBR$alpha, alpha_EBD$alpha,
            alpha_MOS2$alpha, alpha_PANAS_P$alpha, alpha_PANAS_N$alpha, alpha_PCPS3$alpha),
  ci2.5 = c(alpha_EAG$ci[1], alpha_SWLS$ci[1], alpha_EBR$ci[1], alpha_EBD$ci[1],
            alpha_MOS2$ci[1], alpha_PANAS_P$ci[1], alpha_PANAS_N$ci[1], alpha_PCPS3$ci[1]),
  ci97.5 = c(alpha_EAG$ci[2], alpha_SWLS$ci[2], alpha_EBR$ci[2], alpha_EBD$ci[2],
             alpha_MOS2$ci[2], alpha_PANAS_P$ci[2], alpha_PANAS_N$ci[2], alpha_PCPS3$ci[2])) |>
  mutate(across(starts_with("ci"), round, 3)) |>
  unite(col = "CI", ci2.5:ci97.5, sep = " - ") |>
  kable(digits = 3,
        booktabs = TRUE,
        linesep = "",
        align = c("l", rep("c", 4)),
        caption = "Internal consistency of measured scales",
        col.names = c("Variable", "Items", "$n$", "$\\alpha$", "$95\\% CI$"),
        escape = FALSE) |>
  kable_styling(latex_options = "HOLD_position") |>
  footnote(
    general = "95\\\\% confidence intervals were calculated with 1,000 bootstrap samples.
           Standardized Cronbach's alpha ($\\\\alpha$) coefficients were computed.
           $^1$\\\\cite{EAG};
           $^2$\\\\cite{SWLS};
           $^3$\\\\cite{EBR};
           $^4$\\\\cite{EBD};
           $^5$\\\\cite{MOS};
           $^6$\\\\cite{PANAS}.",
    threeparttable = TRUE, footnote_as_chunk = TRUE, escape = FALSE
  )
```

Table S1. *Internal consistency of measured scales*

Variable	Items	n	α	$95\% CI$
Self-Efficacy[1]	10	223	0.905	0.877 — 0.928
Life-Satisfaction[2]	5	253	0.869	0.838 — 0.895
Resilience[3]	4	278	0.861	0.825 — 0.892
Depression[4]	7	223	0.767	0.712 — 0.809
Social Support[5]	19	195	0.977	0.97 — 0.982
PANAS Positive[6]	10	285	0.884	0.858 — 0.906
PANAS Negative[6]	10	282	0.827	0.794 — 0.868
Community Cohesion[x]	3	281	0.769	0.703 — 0.824

Note: 95% confidence intervals were calculated with 1,000 bootstrap samples. Standardized Cronbach's alpha (α) coefficients were computed. [1]Baessler and Schwarzer, 1996; [2]Diener et al., 1985; [3]Sinclair and Wallston, 2004; [4]Andresen et al., 1994; [5]Sherbourne and Stewart, 1991; [6]Watson et al., 1988.

2 Data Preprocessing

2.1 Renaming, recoding, and filtering

XXXX

```r
data <- data_RAW |>
  # Rename columns to meaningful names
  rename(
    Age = SD1,
    City = SD2,
    Gender = SD3,
    Sexualientation = SD4,
    Sex = SD5,
    Ethnicity = SD6,
    Farmer = SD7,
    Marital_Status = SD8,
    SES = SD9, # Socioeconomic Status
    Education = SD10,
    Children = SD11,
    Housing = SD12,
    Cohabitant = SD13,
    Monthly_Income = SD14,
    Income_Source = SD15,
    Employment = SD16,
    Job = SD17,
    # Disabilities and difficulties
    Hearing_Difficulties = SD18_1,
    Speaking_Difficulties = SD18_2,
    Seeing_Difficulties = SD18_3,
    Moving_Difficulties = SD18_4,
    Grabing_Difficulties = SD18_5,
    Understanding_Difficulties = SD18_6,
    Interacting_Difficulties = SD18_7,
    # Lifetime Prevalence (LP) of substance use
    LP_Alcohol = SD19_1_A,
    LP_Cigarette = SD19_2_A,
    LP_Cannabis = SD19_3_A,
    LP_Cocaine = SD19_4_A,
    LP_Basuco = SD19_5_A,
    LP_Inhalant = SD19_6_A,
    LP_Ecstasy = SD19_7_A,
    LP_Psilocybin = SD19_8_A,
    LP_LSD = SD19_9_A,
    LP_Tranquilizer = SD19_10_A,
    LP_Popper = SD19_11_A,
    LP_Anfetamines = SD19_12_A,
    LP_Heroine = SD19_13_A,
    # Last Month (LM) substance use
    LM_Alcohol = SD19_1_B,
    LM_Cigarette = SD19_2_B,
    LM_Cannabis = SD19_3_B,
    LM_Cocaine = SD19_4_B,
    LM_Basuco = SD19_5_B,
    LM_Inhalant = SD19_6_B,
    LM_Ecstasy = SD19_7_B,
    LM_Psilocybin = SD19_8_B,
    LM_LSD = SD19_9_B,
    LM_TRAN = SD19_10_B,
    LM_Popper = SD19_11_B,
    LM_Anfetamines = SD19_12_B,
    LM_Heroine = SD19_13_B,
    # Last Week (LW) substance use
    LW_Alcohol = SD19_1_C,
    LW_Cigarette = SD19_2_C,
    LW_Cannabis = SD19_3_C,
    LW_Cocaine = SD19_4_C,
    LW_Basuco = SD19_5_C,
    LW_Inhalant = SD19_6_C,
    LW_Ecstasy = SD19_7_C,
    LW_Psilocybin = SD19_8_C,
    LW_LSD = SD19_9_C,
    LW_Tranquilizer = SD19_10_C,
    LW_Popper = SD19_11_C,
    LW_Anfetamines = SD19_12_C,
    LW_Heroine = SD19_13_C,
    Health = SD20_1,
    # Health and other variables
    Illness = SD21,
    Disease_Other = SD22_13_TEXT,
    PCPS1_4_Other = PCPS1_4_texto,
    eed1_7_Other = EED1_7_TEXT
  ) |>
  # Replace character "99" with NA for missing values
  mutate(across(where(is.character), ~ na_if(., "99"))) |>
  # Replace numeric 99 with NA for missing values
  mutate(across(where(is.numeric), ~ na_if(., 99))) |>
  # Recode gender categories into descriptive labels
  mutate(
    Gender = recode(
      Gender,
      "1" = "Male",
      "2" = "Female",
      "3" = "Androgynous",
      "4" = "Trans woman",
      "5" = "Trans man",
      "6" = "Trans feminine",
      "7" = "Trans masculine",
      "8" = "Queer",
      "9" = "Non-binary",
      "10" = "Don't know",
      "11" = "Other"
    )) |>
  # Create a broader Gender category for analysis
  mutate(Gender = if_else(Gender %in% c(
    "Woman", "Trans feminine", "Transexual", "Travesti", "Trans woman"),
    "Trans woman",
    if_else(Gender %in% c("Man", "Trans masculine", "Trans man"),
            "Trans man",
            "Non-binary")
  )) |>
  # Recode housing categories into descriptive labels
  mutate(
    Housing = recode(
      Housing,
      "1" = "Home-owner",
      "2" = "Renting (entire home)",
      "3" = "Living with family",
      "4" = "Shared rental (room)",
      "5" = "Without permanent housing"
    )
  ) |>
  # Recode substance use responses from text to binary (1 = Yes, 0 = No)
  mutate_at(
    c(
      "LP_Alcohol",
      "LP_Cigarette",
      "LP_Cannabis",
      "LP_Cocaine",
      "LP_Basuco",
      "LP_Inhalant",
      "LP_Ecstasy",
      "LP_Psilocybin",
      "LP_LSD",
      "LP_Tranquilizer",
      "LP_Popper",
      "LP_Anfetamines",
      "LP_Heroine",
      "LM_Alcohol",
      "LM_Cigarette",
      "LM_Cannabis",
      "LM_Cocaine",
      "LM_Basuco",
      "LM_Inhalant",
      "LM_Ecstasy",
      "LM_Psilocybin",
      "LM_LSD",
      "LM_TRAN",
      "LM_Popper",
      "LM_Anfetamines",
      "LM_Heroine",
      "LW_Alcohol",
      "LW_Cigarette",
      "LW_Cannabis",
      "LW_Cocaine",
      "LW_Basuco",
      "LW_Inhalant",
      "LW_Ecstasy",
      "LW_Psilocybin",
      "LW_LSD",
      "LW_Tranquilizer",
      "LW_Popper",
      "LW_Anfetamines",
      "LW_Heroine"
    ),
    ~ recode(.x, "1" = 1, "2" = 0)
  ) |>
  # Select only relevant variables
  select(
    -c(
      Codigo,
      ends_with("_TEXT"),
      Sexualientation,
      ends_with("_texto")
    )
  ) |>
  # Recode ethnicity, farmer status, marital status, SES, and education
  mutate(
    Ethnicity = recode(
      Ethnicity,
      "1" = "Indigenous",
      "2" = "Rrom",
      "3" = "Afro-Colombian",
      "4" = "Afro-Colombian",
      "5" = "Afro-Colombian",
      "6" = "Afro-Colombian",
      "7" = "Afro-Colombian",
      "8" = "Afro-Colombian",
      "9" = "None"
    )
  ) |>
  mutate(Farmer = recode(
    Farmer,
    "1" = "Yes",
    "2" = "No",
    "5" = NA_character_
  )) |>
  mutate(
    Marital_Status = recode(
      Marital_Status,
      "1" = "Married",
      "2" = "Single",
      "3" = "Widow/er",
      "4" = "Divorced",
      "5" = "Civil union",
      "6" = "Stable relationship"
    )
  ) |>
  mutate(
    SES = recode_factor(
      SES,
      "1" = "Low",
      "2" = "Low",
      "7" = "Low",
      "3" = "Middle-low",
      "4" = "Middle-high",
      "5" = "High",
      "6" = "High"
    )
  ) |>
  mutate(
    Education = recode_factor(
      Education,
      "1" = "No studies, illiterate",
      "2" = "No studies, literate",
      "3" = "Primary school (unfinished)",
      "4" = "Primary school",
      "5" = "Secondary school (unfinished)",
      "6" = "Secondary school",
      "7" = "Technical degree",
      "8" = "University (unfinished)",
      "9" = "University",
      "10" = "Postgraduate studies"
    )
  ) |>
  mutate(across(c(SD22_1:SD22_13,
                  EED1_1,
                  EED1_2,
                  EED1_3,
                  EED1_4,
                  EED1_5,
                  EED1_6,
                  EED1_7,
                  EED2_1:EED2_5),
                ~ as.numeric(
                  recode(
                    as.character(.x),
                    "1" = "1",
                    "2" = "0",
                    .default = NA_character_,
                    .missing = NA_character_
                  )
                ))) |>
  # Convert disability variables to binary (1 = Has difficulty, 0 = No difficulty)
  mutate(across(
    ends_with("_Difficulties"),
    ~ case_when(.x == 99 ~ NA_real_, is.na(.x) ~ NA_real_, .x == 4 ~ 1, TRUE ~ 0)
  )) |>
  # Create a new variable 'difficulty_dichotomous' to indicate whether a person has
  # any difficulties across different categories
  mutate(Difficulty_Dichotomous = if_else(
    # If any of the difficulties variables (e.g., Hearing_Difficulties, Speaking_Difficulties, etc.) are N
    rowSums(across(ends_with("_Difficulties"), ~ is.na(.))) > 0,
    NA_real_, # Assign NA if any difficulty is missing
    # Otherwise, check if all seven difficulties are marked as '1' (indicating impairment)
    if_else(rowSums(across(ends_with("_Difficulties"), ~ . == 1)) == 7,
            1, # Assign 1 if the person has all seven difficulties
            0 # Assign 0 otherwise
    )
  )) |>
  # Recode PCPS1_1 to PCPS1_5: Convert 1 to 1 (yes) and 2 to 0 (no), with NA for other values
  mutate(across(PCPS1_1:PCPS1_5, ~ case_when(
    . == 1 ~ 1, # Yes
    . == 2 ~ 0, # No
    TRUE ~ NA_real_ # Missing or other values
  ))) |>
  # Recode PCPS2_1 to PCPS2_5: Convert 1 to 0 (no engagement), and 2-5 to 1 (some engagement)
  mutate(across(PCPS2_1:PCPS2_5, ~ case_when(
    . == 1 ~ 0, # No engagement
    . %in% 2:5 ~ 1, # Some engagement
    TRUE ~ NA_real_ # Missing or other values
```

```r
))) |>

mutate(across(starts_with("EBD_"), ~ ifelse(is.na(.x), NA, .x - 1))) |>
# Compute aggregate variables summarizing different aspects
mutate(
  # Count the number of substances used in the last month
  Polyconsumption_Month = rowSums(across(LM_Alcohol:LM_Heroine, ~.), na.rm = TRUE),
  # Count the number of reported diseases or health conditions
  Disease_Burden = rowSums(across(SD22_1:SD22_13, ~.), na.rm = TRUE),
  # Count the number of group memberships (sum of binary indicators)
  Group_Membership = rowSums(across(PCPS1_1:PCPS1_5, ~.), na.rm = TRUE),
  # Count the number of community engagement activities
  Community_Engagement = rowSums(across(PCPS2_1:PCPS2_5, ~.), na.rm = TRUE),
  # The following line is commented out: it would sum discrimination experiences
  Discrimination = rowSums(across(EED1_1:EED1_7, ~.), na.rm = TRUE),
  Discrimination = ifelse(Discrimination >= 1, 1, 0),

  Self_Efficacy = if_else(rowSums(!is.na(across(starts_with("EAG_")))) >= 5,
                          rowMeans(across(starts_with("EAG_")), na.rm = TRUE),
                          NA_real_),

  Life_Satisfaction = if_else(rowSums(!is.na(across(starts_with("SWLS_")))) >= 3,
                              rowMeans(across(starts_with("SWLS_")), na.rm = TRUE),
                              NA_real_),

  Resilience = if_else(rowSums(!is.na(across(starts_with("EBR_")))) >= 3,
                       rowMeans(across(starts_with("EBR_")), na.rm = TRUE),
                       NA_real_),

  Depression = if_else(rowSums(!is.na(across(starts_with("EBD_")))) >= 6,
                       rowMeans(across(starts_with("EBD_")), na.rm = TRUE),
                       NA_real_),

  Social_Support = if_else(rowSums(!is.na(across(starts_with("MOS2_")))) >= 10,
                           rowMeans(across(starts_with("MOS2_")), na.rm = TRUE),
                           NA_real_),

  Positive_Affect = if_else(rowSums(!is.na(across(all_of(PANAS_P)))) >= 8,
                            rowMeans(across(all_of(PANAS_P)), na.rm = TRUE),
                            NA_real_),

  Negative_Affect = if_else(rowSums(!is.na(across(all_of(PANAS_N)))) >= 9,
                            rowMeans(across(all_of(PANAS_N)), na.rm = TRUE),
                            NA_real_),

  Community_Cohesion = if_else(rowSums(!is.na(across(starts_with("PCPS3_")))) >= 2,
                               rowMeans(across(starts_with("PCPS3_")), na.rm = TRUE),
                               NA_real_),

) |>
select(Age, Gender, Ethnicity, Marital_Status, SES, Education, Housing,
       Health, Polyconsumption_Month:Community_Cohesion) |>
# Convert categorical variables Housing to Job into factors
mutate(Housing = as.factor(Housing)) |>
# Convert all remaining character variables to factors
mutate_if(is.character, as.factor) |>
```

```
  filter(Age >= 18)
```

2.2 Missing Data

Create a summary of missing data for each variable in the final dataset

```
Missing_data <- data |>
  # Summarize across all columns, counting the number of NA values in each column
  summarise(across(everything(), ~ sum(is.na(.)))) |>
  # Convert the summary from wide format (one row, many columns) to long format
  pivot_longer(
    everything(),               # Select all columns
    names_to = "Variable",      # Store column names in a new variable "Variable"
    values_to = "NA_count"      # Store the count of NAs in a new variable "NA_count"
  ) |>
  # Compute the proportion of missing values for each variable
  mutate(Proportion = NA_count / dim(data)[1])  # Divide NA count by total number of rows in 'data'
```

2.2.1 Fig. S1. Proportion of missing data

To apply multi-model inference techniques such as dredge and model averaging, models must be fitted with complete data. Therefore, assessing the proportion of missing data per variable was crucial. While excessive missingness could lead to unreliable models, imputing missing values might reduce data credibility. Since no variable had an unacceptably high proportion of missing data, we opted not to impute missing values.

```
Missing_data |>
  mutate_at("Variable", str_replace_all, "_", " ") |>
  ggplot(aes(
    x = fct_reorder(Variable, Proportion, .desc = TRUE), # Reorder variables from highest to lowest missin
    y = Proportion,
    fill = Proportion  # Use fill color to indicate proportion of missing data
  )) +
  geom_col() +  # Create bar plot
  # Add percentage labels on top of bars
  geom_text(aes(label = percent(Proportion, accuracy = 1)),
            vjust = -0.5, size = 2) +
  # Apply color gradient: Green (low missing data) → Yellow (moderate) → Red (high missing data)
  scale_fill_viridis_c(
    option = "plasma",  # Define color range
    direction = -1,  # Reverse the color scale
    labels = percent_format(accuracy = 1)  # Convert legend values to percentage format
  ) +
  # Convert Y-axis (proportion of missing data) to a percentage scale
  scale_y_continuous(labels = scales::percent_format(accuracy = 1)) +
  # Add axis labels
  labs(
    y = "Percentage of Missing Data",  # Label for Y-axis
    x = "Variable"  # Label for X-axis
  ) +
  # Use a minimal theme for a cleaner visual appearance
  theme_minimal() +
  # Rotate X-axis labels for better readability
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

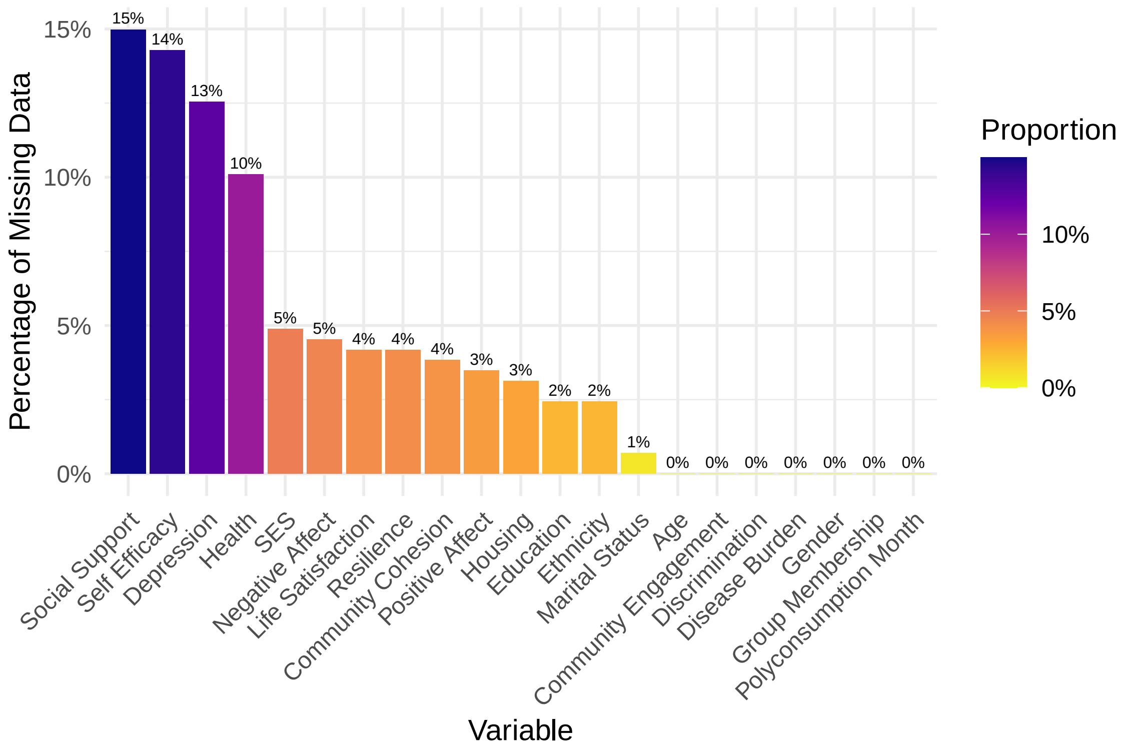

Figure S1. Proportion of missing data per variable. Variables are ordered from highest to lowest proportion of missing values. The color gradient indicates the proportion of missingness, with darker shades representing higher percentages.

3 Multi-model inference

XXXX

3.1 Life Satisfaction model

XXXX

```r
dat_LS <- data |>
  select(Life_Satisfaction, Age, Gender, Ethnicity, Marital_Status, Education, Housing,
         Self_Efficacy, Community_Cohesion, Depression, Social_Support, Polyconsumption_Month,
         Disease_Burden, Discrimination, Group_Membership, Community_Engagement)|>
  drop_na()

global_LS <- lm(Life_Satisfaction ~ Age + Gender + Ethnicity + Marital_Status + Education +
                  Housing + Self_Efficacy + Community_Cohesion + Depression +
                  Social_Support + Polyconsumption_Month + Disease_Burden + Discrimination +
                  Group_Membership + Community_Engagement,
                data = dat_LS,
                na.action = "na.fail")
```

3.1.1 Dredge

XXXX

```r
dr_LS<- dredge(global_LS,
               trace = 2 #para ver barra de progreso
)
```

```
##      |                                                                              |
```

```
plot(dr_LS[1:100,])
```

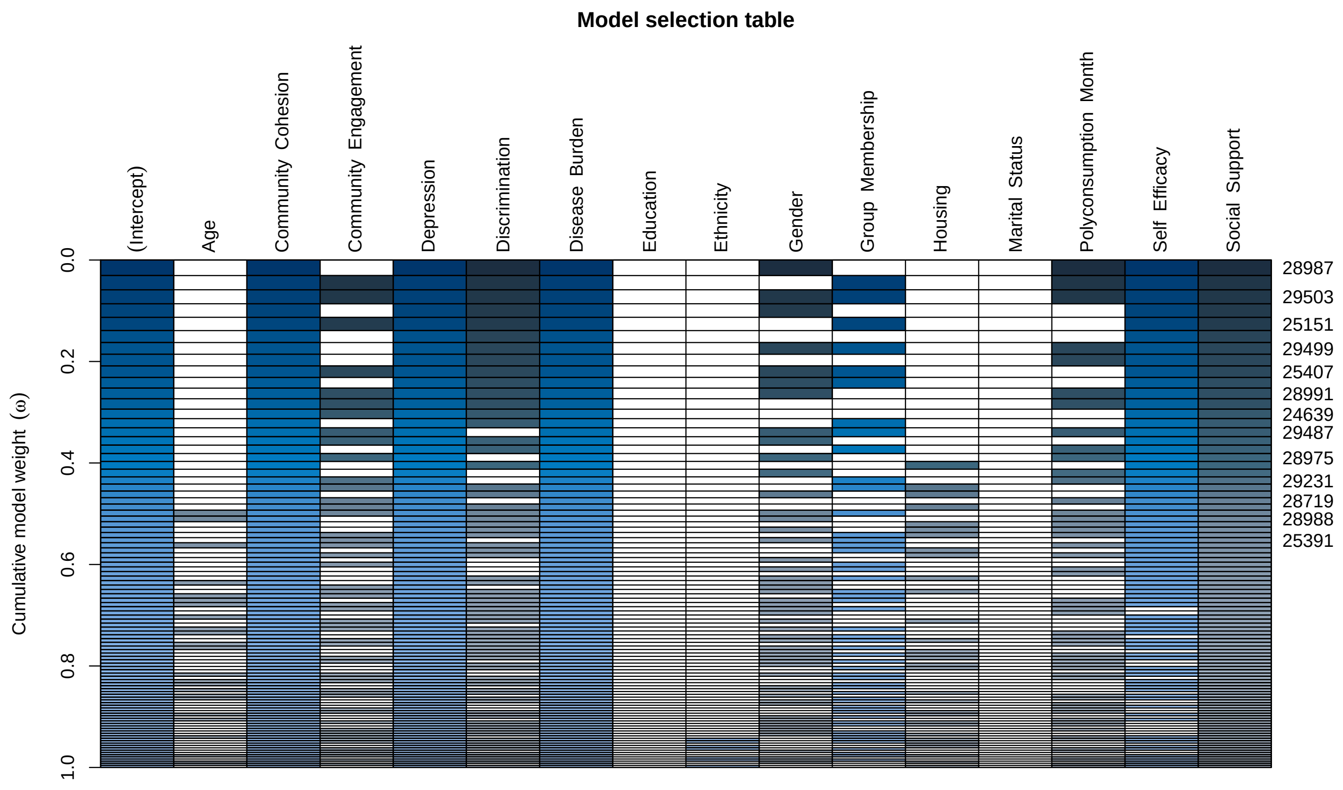

Figure S2. XXXX.

3.1.2 Average model

XXXX

```
avg_LS <- model.avg(dr_LS, subset = delta < 2, fit = TRUE)
```

3.1.2.1 Table S2. **XXXX** XXXX

```
avg.model.anova(avg_LS, data = dat_LS, response = "Life_Satisfaction")
```

Table S2. *XXXXXX*

Term	SS_{term}	*df*	*F*	*p*
(Intercept)	45.147	1, 193	23.014	**< 0.0001**
Community Cohesion	20.366	1, 193	10.382	**0.0015**
Depression	18.169	1, 193	9.262	**0.0027**
Discrimination	6.953	1, 193	3.544	0.06
Disease Burden	13.223	1, 193	6.740	**0.0102**
Polyconsumption Month	4.738	1, 193	2.415	0.12
Self Efficacy	10.125	1, 193	5.161	**0.0242**
Social Support	20.028	1, 193	10.209	**0.0016**
Community Engagement	6.513	1, 193	3.320	0.07
Group Membership	5.470	1, 193	2.788	0.1
Age	0.363	1, 193	0.185	0.67

Note:
This ANOVA table was generated based on model-averaged estimates from multimodel inference. The predictor terms included in the model were selected based on their relative importance across candidate models ($\Delta AICc < 2$). Sum of squares (SS_{term}) values correspond to Type III ANOVA calculations, which test each term's contribution while controlling for all other predictors. Degrees of freedom (*df*) are presented as term *df* and residual *df*, where residual *df* reflects the remaining degrees of freedom in the model. The *F* and *p* values were computed from the refitted model using only the selected predictors. Significant effects are in bold.

3.1.2.2 Fig. S3. Dredge results of the Life Satisfaction model XXXX

```r
avg.mod.plot(avg_LS)
```

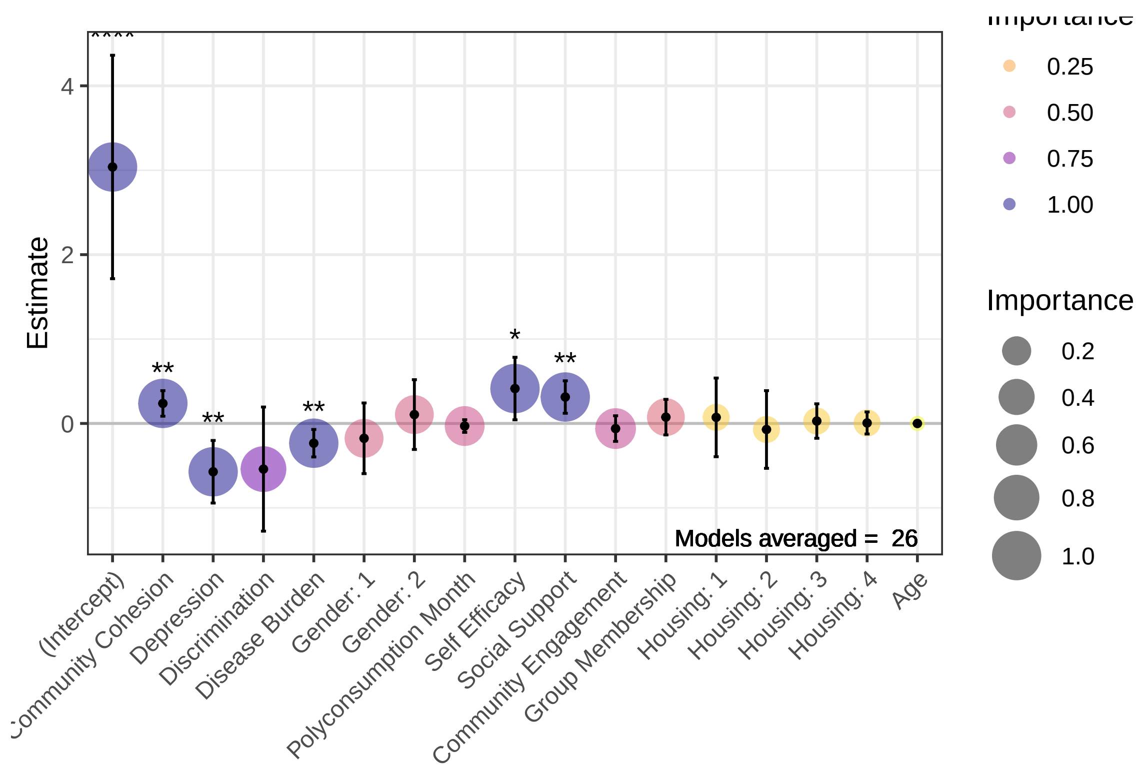

Figure S3. XXXX.

4 Session info (for reproducibility)

```r
# Display session information for reproducibility
# - Uses `pander()` for better formatting
# - `locale = FALSE` to exclude locale-specific info (reduces clutter)
library(pander)
pander(sessionInfo(), locale = FALSE)
```

R version 4.4.3 (2025-02-28)

Platform: x86_64-pc-linux-gnu

attached base packages: *stats*, *graphics*, *grDevices*, *utils*, *datasets*, *methods* and *base*

other attached packages: *pander(v.0.6.6)*, *lubridate(v.1.9.4)*, *forcats(v.1.0.0)*, *stringr(v.1.5.1)*, *dplyr(v.1.1.4)*, *purrr(v.1.0.4)*, *tidyr(v.1.3.1)*, *tibble(v.3.2.1)*, *ggplot2(v.3.5.1)*, *tidyverse(v.2.0.0)*, *car(v.3.1-3)*, *carData(v.3.0-5)*, *kableExtra(v.1.4.0)*, *scales(v.1.3.0)*, *readr(v.2.1.5)*, *performance(v.0.13.0)*, *MuMIn(v.1.48.4)*, *psych(v.2.4.12)*, *ltm(v.1.2-0)*, *polycor(v.0.8-1)*, *msm(v.1.8.2)*, *MASS(v.7.3-64)* and *knitr(v.1.49)*

loaded via a namespace (and not attached): *gtable(v.0.3.6)*, *xfun(v.0.51)*, *insight(v.1.1.0)*, *lattice(v.0.22-6)*, *tzdb(v.0.4.0)*, *vctrs(v.0.6.5)*, *tools(v.4.4.3)*, *generics(v.0.1.3)*, *stats4(v.4.4.3)*, *parallel(v.4.4.3)*, *pkgconfig(v.2.0.3)*, *Matrix(v.1.7-2)*, *lifecycle(v.1.0.4)*, *farver(v.2.1.2)*, *compiler(v.4.4.3)*, *munsell(v.0.5.1)*, *mnormt(v.2.1.1)*, *htmltools(v.0.5.8.1)*, *yaml(v.2.3.10)*, *Formula(v.1.2-5)*, *crayon(v.1.5.3)*, *pillar(v.1.10.1)*, *admisc(v.0.37)*, *abind(v.1.4-8)*, *nlme(v.3.1-167)*, *tidyselect(v.1.2.1)*, *digest(v.0.6.37)*, *mvtnorm(v.1.3-3)*, *stringi(v.1.8.4)*, *bookdown(v.0.42)*, *labeling(v.0.4.3)*, *splines(v.4.4.3)*, *fastmap(v.1.2.0)*, *grid(v.4.4.3)*, *archive(v.1.1.11)*, *colorspace(v.2.1-1)*, *expm(v.1.0-0)*, *cli(v.3.6.3)*, *magrittr(v.2.0.3)*, *survival(v.3.8-3)*, *broom(v.1.0.7)*, *withr(v.3.0.2)*, *backports(v.1.5.0)*, *bit64(v.4.6.0-1)*, *timechange(v.0.3.0)*, *rmarkdown(v.2.29)*, *bit(v.4.5.0.1)*, *hms(v.1.1.3)*, *evaluate(v.1.0.3)*, *viridisLite(v.0.4.2)*, *rlang(v.1.1.5)*, *Rcpp(v.1.0.14)*, *glue(v.1.8.0)*, *xml2(v.1.3.6)*, *vroom(v.1.6.5)*, *svglite(v.2.1.3)*, *rstudioapi(v.0.17.1)*, *R6(v.2.5.1)* and *systemfonts(v.1.2.1)*

5 Supplementary references

Andresen, E. M., Malmgren, J. A., Carter, W. B., & Patrick, D. L. (1994). Screening for Depression in Well Older Adults: Evaluation of a Short Form of the CES-D. *American Journal of Preventive Medicine*, *10*(2), 77–84. https://doi.org/10.1016/S0749-3797(18)30622-6

Baessler, J., & Schwarzer, R. (1996). Evaluación de la autoeficacia: Adaptación española de la Escala de Autoeficacia general. [Measuring optimistic self-beliefs: A Spanish adaptation of the General Self-Efficacy Scale.] *Ansiedad y Estrés*, *2*(1), 1–8. https://psycnet.apa.org/record/1999-00958-001

Bartoń, K. (2024). *Mumin: Multi-model inference* [R package version 1.48.4]. https://CRAN.R-project.org/package=MuMIn

Diener, E., Emmons, R. A., Larsen, R. J., & Griffin, S. (1985). The Satisfaction With Life Scale. *Journal of Personality Assessment*, *49*(1), 71–75. https://doi.org/10.1207/s15327752jpa4901_13

Lüdecke, D., Ben-Shachar, M. S., Patil, I., Waggoner, P., & Makowski, D. (2021). performance: An R package for assessment, comparison and testing of statistical models. *Journal of Open Source Software*, *6*(60), 3139. https://doi.org/10.21105/joss.03139

Rizopoulos, D. (2006). ltm: An R package for latent variable modeling and item response theory analyses. *Journal of Statistical Software*, *17*(5), 1–25. https://doi.org/10.18637/jss.v017.i05

Sherbourne, C. D., & Stewart, A. L. (1991). The MOS social support survey. *Social Science & Medicine*, *32*(6), 705–714. https://doi.org/10.1016/0277-9536(91)90150-B

Sinclair, V. G., & Wallston, K. A. (2004). The Development and Psychometric Evaluation of the Brief Resilient Coping Scale. *Assessment*, *11*(1), 94–101. https://doi.org/10.1177/1073191103258144

Watson, D., Clark, L. A., & Tellegen, A. (1988). Development and validation of brief measures of positive and negative affect: The PANAS scales. *Journal of Personality and Social Psychology*, *54*(6), 1063–1070. https://doi.org/10.1037/0022-3514.54.6.1063

Wickham, H. (2016). *Ggplot2: Elegant graphics for data analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., . . . Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. https://doi.org/10.21105/joss.01686

Wickham, H., François, R., Henry, L., Müller, K., & Vaughan, D. (2023). *Dplyr: A grammar of data manipulation* [R package version 1.1.3]. https://CRAN.R-project.org/package=dplyr

Xie, Y. (2014). Knitr: A comprehensive tool for reproducible research in R [ISBN 978-1466561595]. In V. Stodden, F.

Leisch & R. D. Peng (Eds.), *Implementing reproducible computational research*. Chapman and Hall/CRC. https://doi.org/10.1201/9781315373461-1

Zhu, H. (2021). *Kableextra: Construct complex table with 'kable' and pipe syntax* [R package version 1.3.4]. https://CRAN.R-project.org/package=kableExtra