Funciones evolutivas del habla dirigida a bebés: Impacto en la atención, las preferencias auditivas y el desarrollo lingüístico y musical temprano

Estimación de poder con base en simulaciones

Juan David Leongómez [1,*]

06 de julio de 2025

[1] CODEC: Cognitive and Behavioural Sciences, Faculty of Psychology, Universidad El Bosque, Bogotá 110121, Colombia.

[*] Correspondencia: jleongomez@unbosque.edu.co

Description

Data simulation and power analysis approach:

This technical appendix reports a Monte Carlo simulation study conducted to estimate the statistical power to detect main and interaction effects in a planned 2×2×2 within-subjects design. The outcome variable is total fixationtime, modeled as a function of three binary acoustic predictors: pitchvariability (f_0 SD), pitch mean (f_0 mean), and perceptual distance (D_f).Realistic effect sizes and noise were incorporated to reflect expectedexperimental conditions. Simulated datasets were analyzed using linearmixed-effects models with random intercepts for participants. Resultsdemonstrate that with approximately 160 participants, the study would achieve>80% power to detect all main effects. Interaction effects are weaker and wouldrequire larger samples to detect reliably. These findings support the proposedsample size and analytic strategy in the accompanying grant application.

Reproducibility:

This document contains all code, and step by step explanations for all analyses, figures and tables (including supplementary figures and tables) for the simulation-based power analysis for ***Funciones evolutivas del habla dirigida a bebés: Impacto en la atención, las preferencias auditivas y el desarrollo lingüístico y musical temprano***

Data are available on the Open Science Framework (OSF): https://doi.org/10.XXXXXX . The power analysis strategy was designed by Juan David Leongómez, and the main analyses/models were designed by Christopher D. Watkins and Juan David Leongómez. This document and all its underlying code were created in R Markdown by Juan David Leongómez using R and LaTeX.

Contents

1 Preliminaries

1.1 Load Packages

This file was created using knitr (Xie, 2014, 2025), and analyses were performed using tidyverse packages for data manipulation and plotting (Wickham et al., 2019), including dplyr (Wickham et al., 2023), stringr (Wickham, 2023), and ggplot2 (Wickham, 2016). Power simulations were run using purrr and furrr (Vaughan & Dancho, 2022) for functional and parallel programming, and models were fit using lmerTest (Kuznetsova et al., 2017), an extension of lme4 that provides p-values for mixed-effects models. Effect size estimates were calculated using effectsize (Ben-Shachar et al., 2020), and visualizations include enhanced plots using ggdist (Kay, 2024, 2025) for raincloud plots and tidyquant (Dancho & Vaughan, 2025) for custom color palettes. All packages can be installed from the Comprehensive R Archive Network (CRAN). For a complete list of package versions used in this analysis, see the Session Info section at the end of the document.

```
# Load required packages
library(tidyverse)   # Data manipulation and plotting
library(ggdist)      # For raincloud plots
library(tidyquant)   # For custom themes and color scales
library(effectsize)  # For effect size estimates
library(lmerTest)    # Linear mixed effects model with p-values
library(furrr)       # Parallel processing with purrr
library(scales)      # Scaling functions for plots
library(stringr)     # String handling
library(ggpubr)      # Plot arrangement
library(truncnorm)   # Truncated normal distribution
library(TOSTER)      # For power analsis and equivalence testing
```

2 Study 1

2.1 Simulation Strategy

This simulation models **total fixation time** (in milliseconds) as a function of three binary within-subject factors that are experimentally manipulated:

- f_0 *SD*: Pitch variability (Low vs High)
- f_0 *mean*: Mean pitch (Low vs High)
- D_f: Formant dispersion (Low vs High)

Each of **1000 participants** views stimuli representing **all 8 combinations** of these factors. Fixation time is sampled from a normal distribution, with the following logic:

- **Main effects**:
- f_0 *SD*: Strongest effect. Adds 100 ms when High.
- f_0 *mean*: Adds 75 ms when High.

- D_f: Adds 50 ms when High.
- **Interactions**:
- f_0 *SD* $\times$ f_0 *mean*: Adds 100 ms when both High.
- f_0 *SD* $\times$ D_f: Adds 100 ms when both High.
- f_0 *mean* $\times$ D_f: Adds 50 ms when both High.
- 3-way interaction: Adds 120 ms when all three are High.

A normal distribution with SD = 1000 ms is used to generate noisy trial-level data. Values are clamped between 0 and 5000 ms.

2.2 Simulate the Data

```r
set.seed(42)

# Create all 8 condition combinations
stimulus_conditions <- expand_grid(
  f0_sd    = c("Low", "High"),
  f0_mean  = c("Low", "High"),
  Df       = c("Low", "High")
)

# Generate 1000 participant IDs
participant_ids <- str_c("P", str_pad(1:1000, width = 4, pad = "0"))

# Full crossed design: each participant sees every condition
design <- expand_grid(
  ID = participant_ids,
  stimulus_conditions
)

# Simulate fixation times based on main and interaction effects
simulated_data <- design |>
  mutate(
    f0_sd_val   = if_else(f0_sd == "Low", 0, 1),
    f0_mean_val = if_else(f0_mean == "Low", 0, 1),
    Df_val      = if_else(Df == "Low", 0, 1),

    base_mean = 2500 + 100 * f0_sd_val,
    effect_mean = 75 * f0_mean_val,
    effect_df   = 50 * Df_val,

    interaction_effect = 100 * f0_sd_val * f0_mean_val +
      100 * f0_sd_val * Df_val +
      50  * f0_mean_val * Df_val +
      120 * f0_sd_val * f0_mean_val * Df_val,

    mu = base_mean + effect_mean + effect_df + interaction_effect,

    fixation_time = round(rnorm(n(), mean = mu, sd = 1000)),
    fixation_time = pmin(pmax(fixation_time, 0), 5000)
  ) |>
  select(ID, f0_sd, f0_mean, Df, fixation_time)
```

2.2.1 Fit the Linear Mixed-Effects Model

We fit a linear mixed model with participant as a random effect, and test all main effects and interactions.

The equation for the model is:

$$
\begin{aligned}
&\text{fixation\_time}_i \sim N\left(\mu, \sigma^2\right) \\
&\qquad\qquad \mu = \alpha_{j[i]} + \beta_1(\text{f0\_sd}_{\text{Low}}) + \beta_2(\text{f0\_mean}_{\text{Low}}) + \beta_3(\text{Df}_{\text{Low}}) + \beta_4(\text{f0\_mean}_{\text{Low}} \times \text{f0\_sd}_{\text{Low}}) + \beta_5(\text{Df}_{\text{Low}} \times \text{f0\_sd}_{\text{Low}}) + \\
&\qquad\qquad \alpha_j \sim N\left(\mu_{\alpha_j}, \sigma^2_{\alpha_j}\right), \text{ for ID j} = 1, \dots, \text{J}
\end{aligned}
$$

```r
mod <- lmer(fixation_time ~ f0_sd * f0_mean * Df + (1 | ID), data = simulated_data)
anova(mod)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##                      Sum Sq   Mean Sq NumDF DenDF  F value    Pr(>F)
## f0_sd             123939509 123939509     1  6993 126.3168 < 2.2e-16 ***
## f0_mean            55029861  55029861     1  6993  56.0854 7.791e-14 ***
## Df                 48917697  48917697     1  6993  49.8560 1.814e-12 ***
## f0_sd:f0_mean       6209270   6209270     1  6993   6.3284  0.011904 *
## f0_sd:Df            3393438   3393438     1  6993   3.4585  0.062968 .
## f0_mean:Df          9645563   9645563     1  6993   9.8306  0.001723 **
## f0_sd:f0_mean:Df      60836     60836     1  6993   0.0620  0.803365
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

2.2.2 Estimate Partial Eta-Squared for Fixed Effects

We estimate partial omega squared (ω_p^2) to quantify the effect size of each fixed term in the model.

```r
pop_effect <- omega_squared(mod, partial = TRUE)
pop_effect
```

```
## # Effect Size for ANOVA (Type III)
##
## Parameter         | Omega2 (partial) |       95% CI
## ---------------------------------------------------
## f0_sd             |             0.02 | [0.01, 1.00]
## f0_mean           |         7.81e-03 | [0.00, 1.00]
## Df                |         6.94e-03 | [0.00, 1.00]
## f0_sd:f0_mean     |         7.61e-04 | [0.00, 1.00]
## f0_sd:Df          |         3.51e-04 | [0.00, 1.00]
## f0_mean:Df        |         1.26e-03 | [0.00, 1.00]
## f0_sd:f0_mean:Df  |             0.00 | [0.00, 1.00]
##
## - One-sided CIs: upper bound fixed at [1.00].
```

2.2.3 Visualizing Effects: Raincloud Plot

Below is a raincloud plot showing fixation time distributions for each condition. The plot allows visual inspection of differences by f_0 *SD*, split by f_0 *mean* and D_f.

```r
# Prepare data with clean factor levels
plot_data <- simulated_data |>
  mutate(
    f0_mean = factor(f0_mean, levels = c("Low", "High")),
    Df = factor(Df, levels = c("Low", "High")),
    f0_sd = factor(f0_sd, levels = c("Low", "High"))
  )

# Labels for facets
```

```r
label_f0_mean <- c("Low" = "italic(f)[0]*' mean'",
                   "High" = "italic(f)[0]*' mean'")
label_Df <- c("Low" = "italic(D)[f]",
              "High" = "italic(D)[f]")

# Raincloud plot
ggplot(plot_data, aes(x = f0_sd, y = fixation_time, fill = f0_sd)) +
  # Distribution cloud
  stat_halfeye(
    adjust = 0.5,
    justification = -0.3,
    .width = 0,
    point_colour = NA,
    alpha = 0.6
  ) +
  # Raw points
  geom_jitter(
    aes(color = f0_sd),
    width = 0.1,
    alpha = 0.05,
    size = 0.7
  ) +
  # Mean point
  stat_summary(
    fun = mean,
    geom = "point",
    size = 2,
    color = "black",
    position = position_nudge(x = 0.2)
  ) +
  # Error bar
  stat_summary(
    fun.data = mean_se,
    geom = "errorbar",
    width = 0.1,
    color = "black",
    position = position_nudge(x = 0.2)
  ) +
  # Facet by f0_mean × Df
  facet_grid(
    Df ~ f0_mean,
    labeller = labeller(
      f0_mean = as_labeller(label_f0_mean, default = label_parsed),
      Df = as_labeller(label_Df, default = label_parsed)
    )
  ) +
  labs(
    title = "Raincloud Plot of Fixation Times",
    x = expression(italic(f)[0]*" SD"),
    y = "Fixation Time (ms)",
    fill = expression(italic(f)[0]*" SD"),
    color = expression(italic(f)[0]*" SD")
  ) +
  scale_colour_tq() +
  scale_fill_tq() +
  theme_tq(base_size = 14)
```

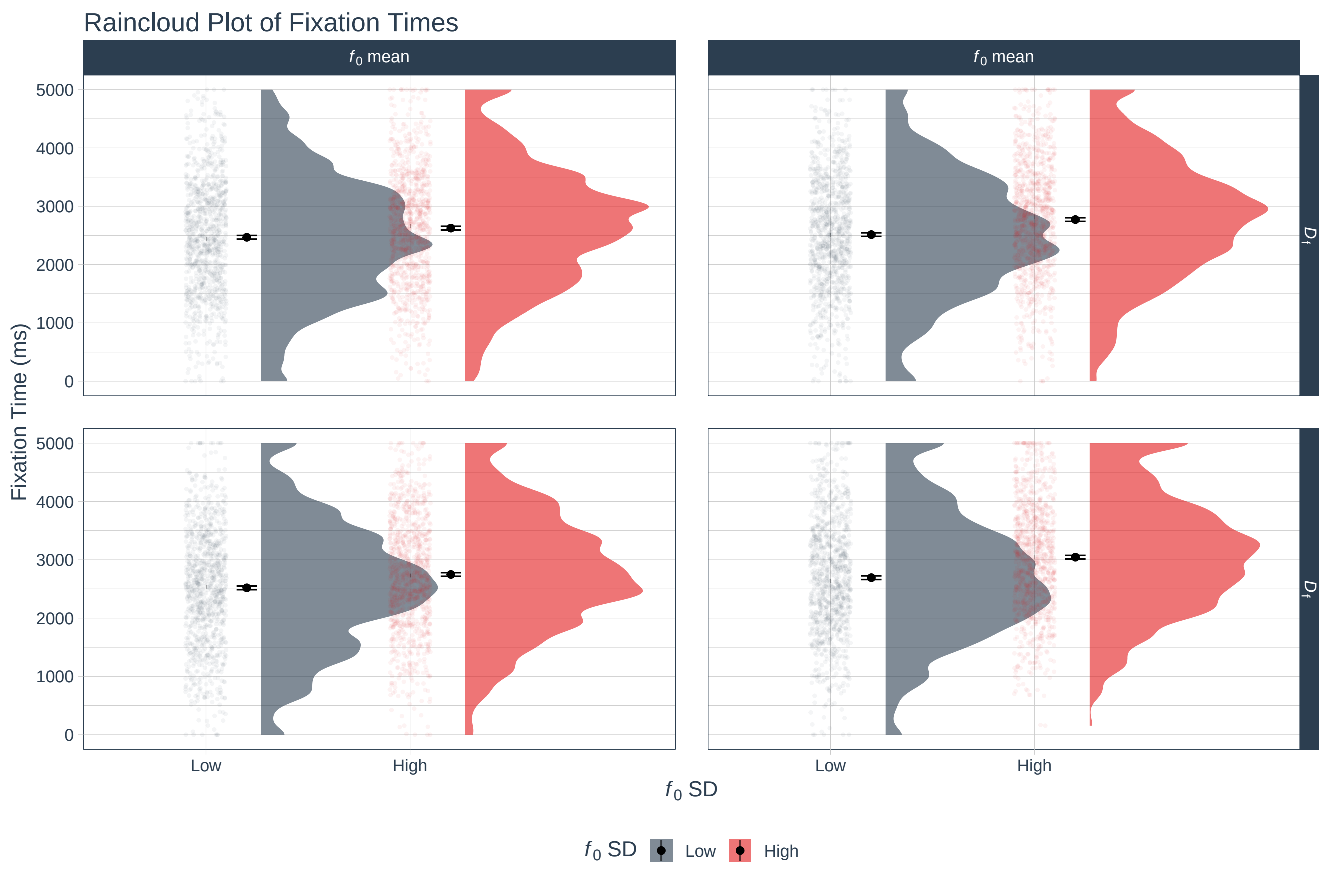

Figure S1. Distribution of fixation times as a function of f_0 *SD* (x-axis), split by f_0 *mean* (columns) and D_f (rows). Each panel includes both the density distribution, individual data points (jittered dots), and condition means with error bars. Fixation times increase with higher f_0 *SD*, and effects are modulated by the other acoustic features.

2.3 Power Estimation via Monte Carlo Simulation

We estimate statistical power for detecting each effect by simulating 1 000 random samples from the simulated population at varying sample sizes. Specifically, from the simulated population ($N = 1\ 000$), we draw 1 000 random samples of size $n = 10$, another 1 000 samples of size $n = 20$, and so on, continuing in this way until we draw 1 000 samples of size $n = 200$. For each of the 20 000 samples, we fit the same linear mixed model, examine the resulting distribution of p-values, and estimate the probability of detecting a statistically significant effect for each model term.

```
# Define alpha level
alpha_lev = 0.05

# Function to run multiple power simulations for each n
run_power_sim <- function(dat, n_sample, n_sim, alpha = alpha_lev) {
  # Run `n_sim` simulations for each sample size
  map_dfr(seq_len(n_sim), \(i) {
    # Sample participants
    ids <- dat |> distinct(ID) |> slice_sample(n = n_sample) |> pull(ID)
    sampled_data <- dat |> filter(ID %in% ids)

    # Fit model
    mod <- lmer(fixation_time ~ f0_sd * f0_mean * Df + (1 | ID), data = sampled_data)
    # Extract ANOVA results
    anova_res <- anova(mod)

    # Grab all terms of interest
    term_names <- rownames(anova_res)

    # Loop over all fixed terms
    map_dfr(term_names, \(term) {
      tibble(
        sim = i,
        term = term,
        n_sample = n_sample,
        p_value = anova_res[term, "Pr(>F)"],
        signif = ifelse(p_value < alpha, "Significant", "Non-significant")
      )
    })
  }) |>
    # Clean up and aggregate
    filter(!is.na(p_value))
}

# Define sample sizes and run the simulations
sample_sizes <- seq(10, 200, by = 10)
term_order <- c(
  "f0_sd",
  "f0_mean",
  "Df",
  "f0_sd:f0_mean",
  "f0_sd:Df",
  "f0_mean:Df",
  "f0_sd:f0_mean:Df"
)

plan(multisession)  # Parallel if available

power_results <- map_dfr(
  sample_sizes,
  ~ run_power_sim(simulated_data, n_sample = .x, n_sim = 10),
  .id = "sample_step"
) |>
  mutate(term = factor(term, levels = term_order))
```

2.3.1 Power summary

This aggregates the simulation results to compute average power for each term and sample size.

```
power_summ <- power_results |>
  group_by(term, n_sample) |>
  summarise(
    power = mean(p_value < alpha_lev, na.rm = TRUE),
    .groups = "drop"
  )
```

2.3.2 Power curve

This plot shows the power curves for all main effects and interactions, highlighting the sample sizes where power exceeds 80%.

```
# Setup labels and clean grouping
term_labels <- c(
  "f0_sd"              = "italic(f)[0]*' SD'",
  "f0_mean"            = "italic(f)[0]*' mean'",
  "Df"                 = "italic(D)[f]",
  "f0_sd:f0_mean"      = "italic(f)[0]*' SD × '*italic(f)[0]*' mean'",
  "f0_sd:Df"           = "italic(f)[0]*' SD × '*italic(D)[f]",
  "f0_mean:Df"         = "italic(f)[0]*' mean × '*italic(D)[f]",
  "f0_sd:f0_mean:Df"   = "italic(f)[0]*' SD × '*italic(f)[0]*' mean × '*italic(D)[f]"
)

# Split by main vs interaction effects
main_power_summ <- power_summ |>
  filter(!str_detect(term, ":"))
int_power_summ <- power_summ |>
  filter(str_detect(term, ":"))

# Find minimum n where power exceeds 0.8
sample_power_08 <- main_power_summ |>
  filter(power > 0.8) |>
  group_by(term) |>
  filter(power == min(power, na.rm = TRUE))

# Use largest of those ns for final recommendation,
# to ensure appropriate power for each main effects
final_sample_size_st1 <- max(sample_power_08$n_sample)

# Power at final n
sample_fin <- main_power_summ |>
  filter(n_sample == final_sample_size_st1) |>
  arrange(desc(power))

# Plot power curves for main and interaction effects
ggarrange(
  ggplot(main_power_summ, aes(x = n_sample, y = power)) +
    geom_line(linewidth = 0.5) +
    geom_point(aes(color = power > 0.8), size = 2) +  # Color by condition
    geom_hline(yintercept = 0.8, linetype = "dashed", color = "red") +
    scale_colour_tq(
      labels = c("FALSE" = "<= 80%", "TRUE" = "> 80%"),
      name = "Power threshold"
    ) +
    scale_y_continuous(labels = scales::percent_format(accuracy = 1)) +
    labs(
      title = "Power Curves for Main Effects",
      x = "Number of Participants",
      y = "Estimated Power"
    ) +
    theme_tq() +
    facet_wrap(~term, labeller = as_labeller(term_labels, label_parsed)),
  ggplot(int_power_summ, aes(x = n_sample, y = power)) +
    geom_line(linewidth = 0.5) +
    geom_point(aes(color = power > 0.8), size = 2) +  # Color by condition
    geom_hline(yintercept = 0.8, linetype = "dashed", color = "red") +
    scale_colour_tq(
      labels = c("FALSE" = "<= 80%", "TRUE" = "> 80%"),
      name = "Power threshold"
    ) +
    scale_y_continuous(labels = scales::percent_format(accuracy = 1)) +
    labs(
      title = "Power Curves for Interactions",
      x = "Number of Participants",
      y = "Estimated Power"
    ) +
    theme_tq() +
    facet_wrap(~term, labeller = as_labeller(term_labels, label_parsed)),
  labels = "AUTO",
  common.legend = TRUE,
  legend = "bottom"
)
```

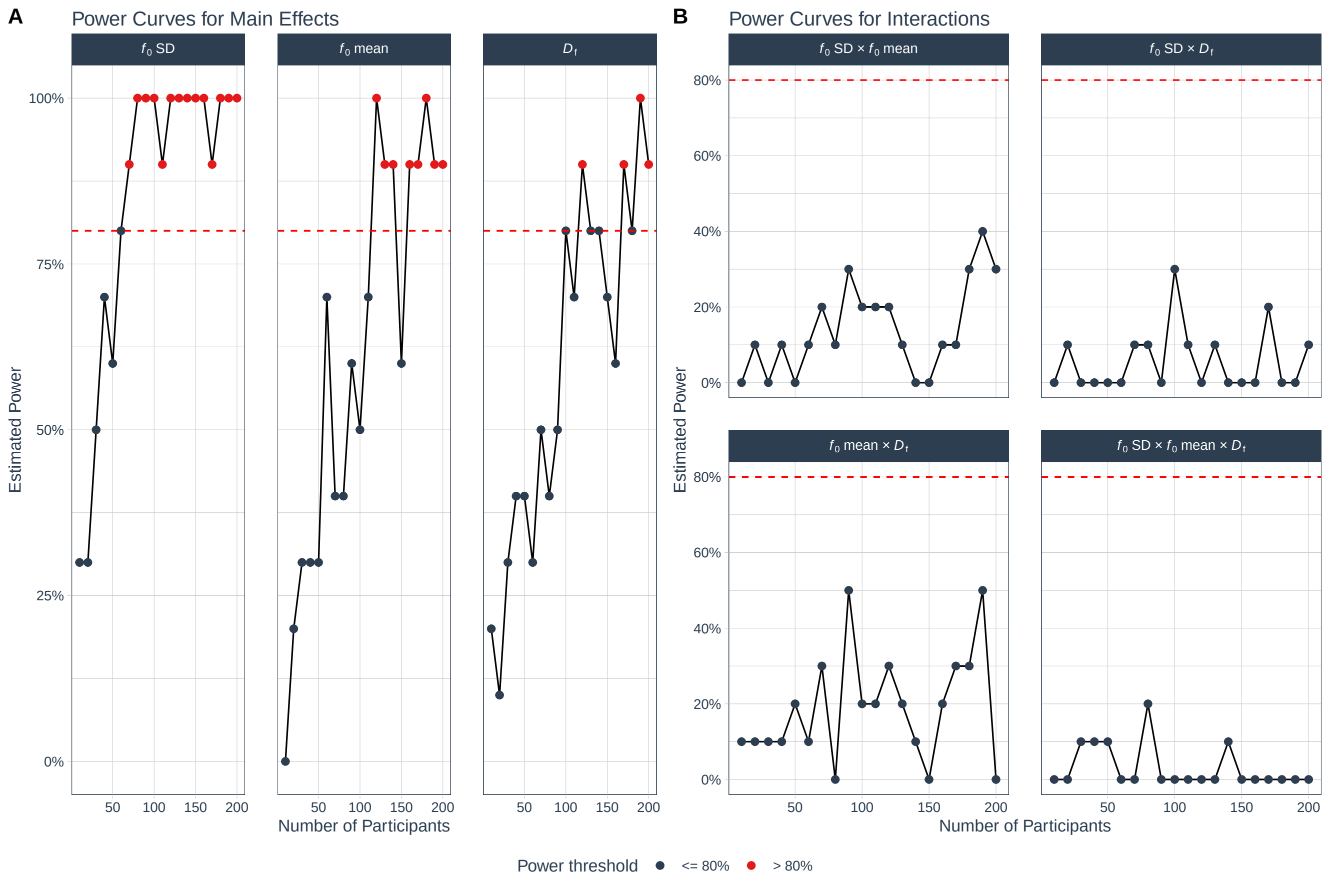

Figure S2. Power curves for detecting main effects and interactions. Panels show statistical power as a function of sample size for each fixed effect (A) and interaction (B). Dots indicate simulation-based power estimates at each sample size, with red dashed lines marking the 80% power threshold. Most main effects achieve high power with sample sizes under 160, whereas interaction terms remain underpowered across this range, reflecting smaller true effects.

2.3.3 Estimating a sample size

In the simulated population, the effects of the interactions are extremely weak and would require a very large sample size to be reliably detected. In contrast, the main effects of f_0 *SD*, f_0 *mean*, and D_f are detectable with a more reasonable sample size. If the true population effects are as simulated ($\omega_p^2 = 0.018, 0.008, 0.007$, respectively), a statistical power of 0.8 would be reached with a sample size of $n = 70, 110, 170$ for f_0 *SD*, $n = 130, 140, 160, 170, 190, 200$ for f_0 *mean*, and $n = 120, 170, 200$ for D_f.

Based on simulations:

- Power exceeds 80% for the main effects at $n = 200$
- Corresponding partial omega squared values were: 0.018, 0.008, 0.007

Therefore, $n = 200$ participants would reliably detect all main effects. Interaction effects require larger samples due to weaker true effects.

2.4 Power Distribution at Final Sample

Below we visualize the distribution of p-values at the final sample size for all effects, illustrating expected Type I/II error profiles.

```r
# Filter for final n
final_power <- power_summ |>
  filter(n_sample == final_sample_size_st1)

# Get full p-value data at that n
main_power_results <- power_results |>
  filter(!str_detect(term, ":")) |>
  group_by(term) |>
  filter(n_sample == final_sample_size_st1) |>
  left_join(final_power, by = c("term", "n_sample"))
int_power_results <- power_results |>
  filter(str_detect(term, ":")) |>
  group_by(term) |>
  filter(n_sample == final_sample_size_st1) |>
  left_join(final_power, by = c("term", "n_sample"))

# Plot distributions
ggarrange(
  ggplot(main_power_results, aes(x = p_value, fill = signif, colour = signif)) +
    geom_histogram(
      bins = 100,
      breaks = seq(0, 1, 0.01),
      alpha = 0.8
    ) +
    geom_text(
      data = main_power_results |> distinct(term, power),
      aes(
        x = Inf, y = Inf,
        label = paste0("Power = ", round(power, 2))
      ),
      vjust = 1.5, hjust = 1.1,
      inherit.aes = FALSE
    ) +
    labs(
      x = "p-value",
      y = "Count",
      fill = "Significance",
      title = "Distribution of p values for Main Effects"
    ) +
    scale_fill_hue(direction = -1) +
    scale_colour_hue(direction = -1) +
    scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
    facet_wrap(~term, labeller = as_labeller(term_labels, label_parsed)) +
    scale_colour_tq() +
    scale_fill_tq() +
    theme_tq() +
    guides(fill = guide_legend(reverse = TRUE)),
  ggplot(int_power_results, aes(x = p_value, fill = signif, colour = signif)) +
    geom_histogram(
      bins = 100,
      breaks = seq(0, 1, 0.01),
      alpha = 0.8
    ) +
    geom_text(
      data = int_power_results |> distinct(term, power),
      aes(
        x = Inf, y = Inf,
        label = paste0("Power = ", round(power, 2))
      ),
      vjust = 1.5, hjust = 1.1,
      inherit.aes = FALSE
    ) +
    labs(
      x = "p-value",
      y = "Count",
      fill = "Significance",
      title = "Distribution of p values for Interactions"
    ) +
    scale_fill_hue(direction = -1) +
    scale_colour_hue(direction = -1) +
    scale_x_continuous(breaks = scales::pretty_breaks(n = 10)) +
    facet_wrap(~term, labeller = as_labeller(term_labels, label_parsed)) +
    scale_colour_tq() +
    scale_fill_tq() +
    theme_tq() +
    guides(fill = guide_legend(reverse = TRUE)),
  labels = "AUTO",
  common.legend = TRUE,
  legend = "bottom"
)
```

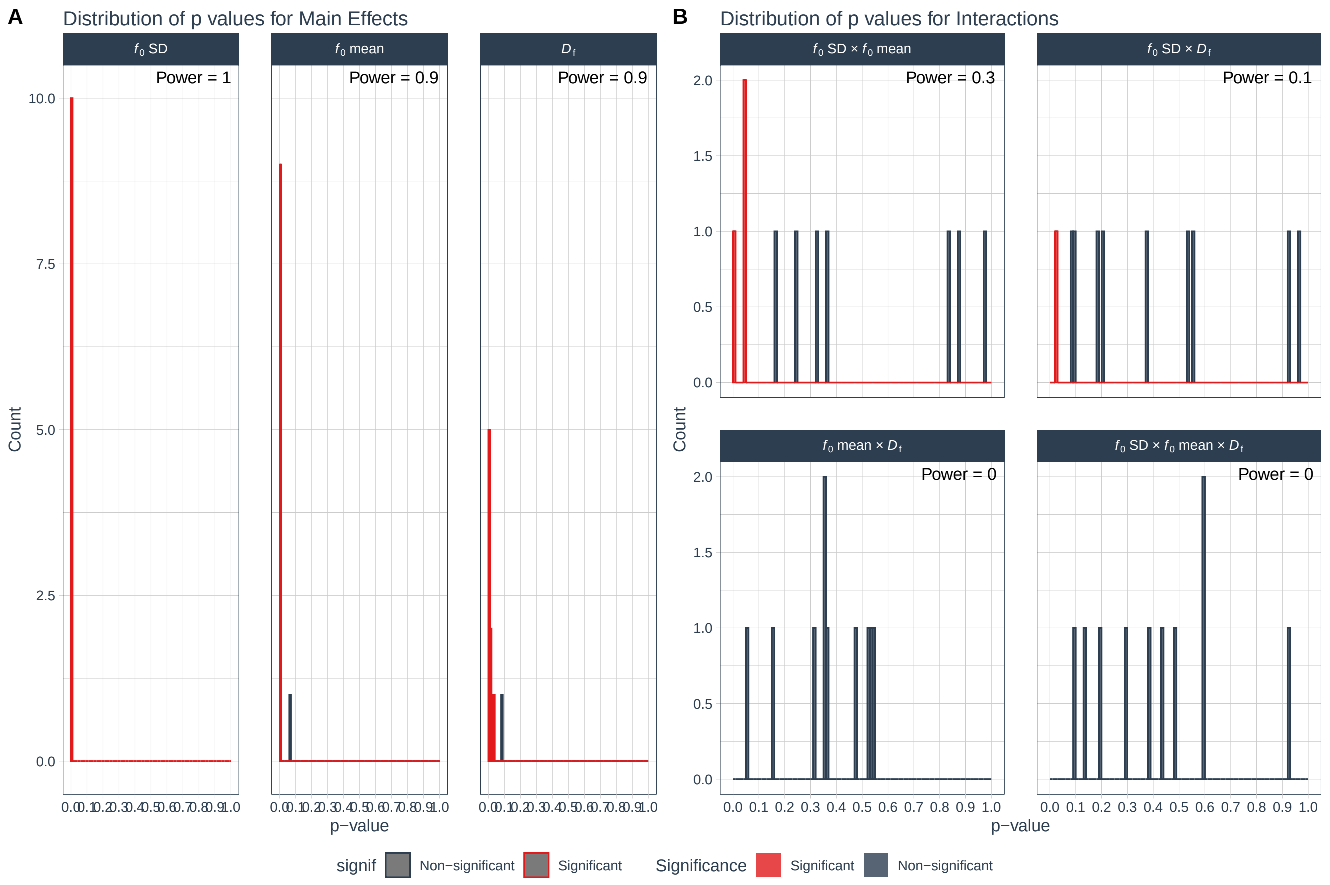

Figure S3. Distribution of p-values across simulations at n = `r final_sample_size_st1`. This figure shows the p-value distribution for each effect, based on 1000 simulation runs using the final recommended sample size. Main effects show a strong skew toward low p-values, consistent with high power. Interactions yield more uniform distributions, indicating limited sensitivity to detect those effects at this sample size.

2.5 Notes

- This simulation estimates **power for detecting the main effect of** f_0 *SD* only.
- You can modify the function to test other terms (e.g., interactions) by indexing a different row in the `anova()` output.
- For precision in planning, repeat this with multiple `n_sample` values.

---

3 Study 2

This section reports a power analysis for Study 2, which investigates the relationship between two continuous variables using Pearson's correlation. We evaluate two complementary statistical goals:

1. **Detecting a correlation** of at least $\rho = 0.30$ (directional, one-sided test).
2. **Testing for equivalence**, to determine whether an observed correlation is statistically indistinguishable from zero, within a predefined margin of ± 0.25.

We used the `TOSTER::power_z_cor()` function to compute power analytically across a range of sample sizes (Caldwell, 2022; Lakens, 2017).

```
# Parameters
eq_bounds <- 0.15
rho_pop <- 0.30
```

```r
alpha <- 0.05
n_sim <- 100000

# Panel A: Power curves
sample_sizes <- 5:600

power_detect <- map_dbl(sample_sizes, ~
                          power_z_cor(
                            alternative = "greater",
                            alpha = alpha,
                            rho = rho_pop,
                            n = .x,
                            power = NULL
                          )$power
)

power_equiv <- map_dbl(sample_sizes, ~
                         power_z_cor(
                           alternative = "equivalence",
                           alpha = alpha,
                           null = eq_bounds,
                           rho = 0,
                           n = .x,
                           power = NULL
                         )$power
)

power_df <- tibble(
  n = sample_sizes,
  Detection = power_detect,
  Equivalence = power_equiv
) |>
  pivot_longer(-n, names_to = "Test", values_to = "Power")

subtitle_text <- substitute(
  expression("Effect size for detection: " * rho == r * ", Equivalence bounds: ±" * eq),
  list(r = rho_pop, eq = eq_bounds)
)

final_sample_size_st2 <- power_df |>
  filter(Test == "Equivalence", Power >= 0.9) |>
  summarise(min_n = min(n)) |>
  pull(min_n)

p_sample_size_st2 <- power_df |>
  filter(Test == "Detection", Power >= 0.9) |>
  summarise(min_n = min(n)) |>
  pull(min_n)

# Power curves
ggplot(power_df, aes(x = n, y = Power, color = Test)) +
  geom_line(linewidth = 1) +
  geom_hline(yintercept = 0.8, linetype = "dashed") +
  geom_hline(yintercept = 0.9, linetype = "dotted") +
  geom_vline(xintercept = final_sample_size_st2, colour = "#E31A1C") +
  annotate("text", y = 0.5, x = final_sample_size_st2,
           label = bquote(italic(n) == .(final_sample_size_st2)),
           hjust = 0.5, vjust = 1.3, angle = 90,
           color = "#E31A1C", size = 3) +
  geom_vline(xintercept = p_sample_size_st2, colour = "#2c3e50") +
  annotate("text", y = 0.5, x = p_sample_size_st2,
           label = bquote(italic(n) == .(p_sample_size_st2)),
           hjust = 0.5, vjust = 1.3, angle = 90,
           color = "#2c3e50", size = 3) +
  scale_y_continuous(labels = scales::percent_format(accuracy = 1)) +
  labs(
    title = "Power Curves for Correlation Detection and Equivalence Testing",
    subtitle = eval(subtitle_text),
    x = "Sample Size (n)",
    y = "Statistical Power",
    color = "Test Type"
  ) +
  scale_colour_tq() +
  theme_tq()
```

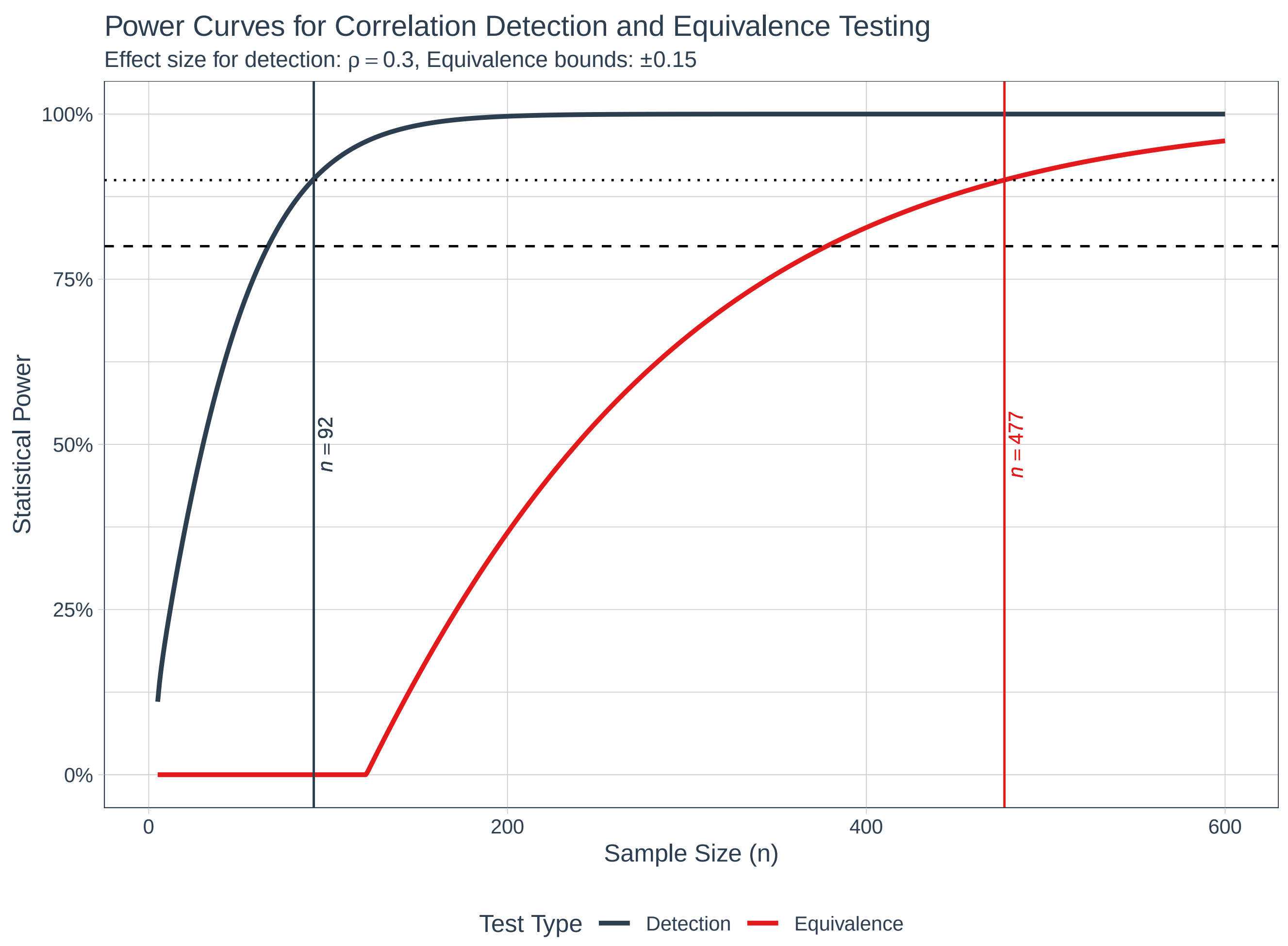

Figure S4. Power curves for detecting a positive correlation (gray line) and for concluding statistical equivalence to zero (red line) across different sample sizes. Dashed and dotted lines represent 80% and 90% power thresholds, respectively. Vertical dotted lines represents the sample size where power reaches $1 - \beta = 0.9$.

In general, with a sample of 477 participants, correlations of $\rho \geq 0.23$ will likely produce conclusive evidence of a positive effect **not** equivalent to zero; correlations of $\rho \leq 0.07$ will lead to conclusive evidence of equivalence to zero. Correlations between $\rho = 0.08$ and 0.22 will typically result in inconclusive outcomes (possibly statistically significant, but not distinguishable from equivalence).

```r
# Panel B: Simulated outcomes under $\rho$ = 0
set.seed(123)
z_sim_0 <- rnorm(n_sim, mean = atanh(0), sd = 1 / sqrt(final_sample_size_st2 - 3))
r_sim_0 <- tanh(z_sim_0)
ci_low_0 <- tanh(z_sim_0 - qnorm(1 - alpha) * (1 / sqrt(final_sample_size_st2 - 3)))
ci_high_0 <- tanh(z_sim_0 + qnorm(1 - alpha) * (1 / sqrt(final_sample_size_st2 - 3)))
equiv_0 <- ci_low_0 > -eq_bounds & ci_high_0 < eq_bounds

df_sim_0 <- tibble(r_obs = r_sim_0, equiv = equiv_0)

# Panel C: Simulated positive detections under $\rho$ = 0.3
# Simulate observed correlations for true $\rho$ = 0.3
z_sim_3 <- rnorm(n_sim, mean = atanh(0.3), sd = 1 / sqrt(final_sample_size_st2 - 3))
r_sim_3 <- tanh(z_sim_3)

# Compute 90% CI
se_z <- 1 / sqrt(final_sample_size_st2 - 3)
ci_low_3 <- tanh(z_sim_3 - qnorm(1 - alpha) * se_z)
ci_high_3 <- tanh(z_sim_3 + qnorm(1 - alpha) * se_z)

# Classify results
test_outcome <- case_when(
  ci_low_3 > eq_bounds ~ "Positive effect",
  ci_low_3 < 0 & ci_high_3 > 0 ~ "Not significant",
  TRUE ~ "Inconclusive"
)

df_sim_3 <- tibble(
  r_obs = r_sim_3,
  outcome = factor(test_outcome,
                   levels = c("Positive effect", "Inconclusive", "Not significant"))
)

# Arrange all three plots
ggarrange(
  # Panel A: Simulated equivalence under $\rho$ = 0
  ggplot(df_sim_0, aes(x = r_obs, fill = equiv)) +
    geom_rect(aes(xmin = -eq_bounds, xmax = eq_bounds, ymin = 0, ymax = Inf),
              fill = "gray90", alpha = 0.4) +
    geom_histogram(binwidth = 0.01, center = 0, alpha = 0.8, color = "gray30") +
    geom_vline(xintercept = c(-eq_bounds, eq_bounds), linetype = "dashed", color = "gray40") +
    geom_vline(xintercept = 0, linetype = "dashed", color = "black") +
    scale_x_continuous(breaks = scales::pretty_breaks(n = 10), limits = c(-0.3, 0.3)) +
    scale_fill_manual(
      values = c("TRUE" = "#D55E00", "FALSE" = "gray70"),
      labels = c("Inconclusive", "Equivalence concluded"),
      name = NULL
    ) +
    labs(
      title = "Simulated Equivalence Test Outcomes ($\rho$ = 0)",
      subtitle = "Shaded bars show CIs fully within ±0.15 (90% CI)",
      x = "Observed Correlation (r)",
      y = "Count"
    ) +
    theme_tq(),
  # Panel B: Simulated non-equivalence under $\rho$ = 0.3
  ggplot(df_sim_3, aes(x = r_obs, fill = outcome)) +
    geom_rect(aes(xmin = -eq_bounds, xmax = eq_bounds, ymin = 0, ymax = Inf),
              fill = "gray90", alpha = 0.4) +
    geom_histogram(binwidth = 0.01, center = 0, alpha = 0.85, color = "gray30") +
    geom_vline(xintercept = c(-eq_bounds, eq_bounds), linetype = "dashed", color = "gray40") +
    scale_x_continuous(breaks = scales::pretty_breaks(n = 10), limits = c(-0.15, 0.45)) +
    scale_fill_manual(
      values = c("Positive effect" = "forestgreen",
                 "Inconclusive" = "gray70",
                 "Not significant" = "red"),
      name = NULL
    ) +
    labs(
      title = "Simulated Test Outcomes Under $\rho$ = 0.3",
      subtitle = "Shaded bars show CIs fully within ±0.15 (90% CI)",
      x = "Observed Correlation (r)",
      y = "Count"
    ) +
    theme_tq(),
  labels = "AUTO",
  nrow = 1,
  common.legend = FALSE
)
```

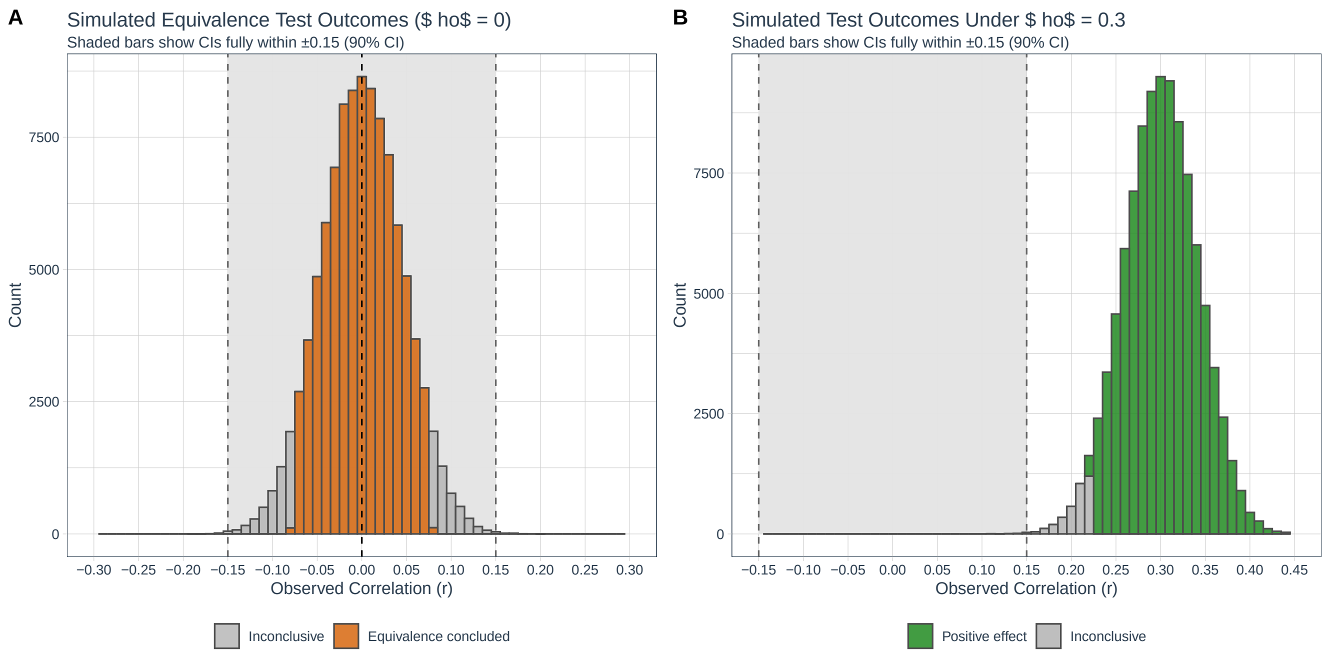

Figure S5. Power analysis for detecting or rejecting a correlation of $ho = 0.30$. **(A)** Simulated observed correlations assuming the true population value is $ho = 0$ (i.e., the null hypothesis is true). Orange bars indicate cases where the 90% confidence interval (CI) falls entirely within the equivalence bounds (±0.15), allowing equivalence to be concluded. Gray bars are inconclusive. **(B)** Simulated observed correlations assuming the true population correlation is $ho = 0.3$. Green bars indicate clear positive effects (lower bound of the 90% CI > 0.15), red bars show nonsignificant results (CI overlaps 0), and gray bars represent inconclusive outcomes (CI lies entirely above 0 but also within the equivalence bounds).

4 Session info (for reproducibility)

```r
library(pander)
pander(sessionInfo(), locale = FALSE)
```

R version 4.5.1 (2025-06-13)

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)*, *TOSTER(v.0.8.4)*, *truncnorm(v.1.0-9)*, *ggpubr(v.0.6.0)*, *scales(v.1.4.0)*, *furrr(v.0.3.1)*, *future(v.1.58.0)*, *lmerTest(v.3.1-3)*, *lme4(v.1.1-37)*, *Matrix(v.1.7-3)*, *effectsize(v.1.0.0)*, *PerformanceAnalytics(v.2.0.8)*, *quantmod(v.0.4.28)*, *TTR(v.0.24.4)*, *xts(v.0.14.1)*, *zoo(v.1.8-14)*, *tidyquant(v.1.0.11)*, *ggdist(v.3.3.3)*, *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)*, *readr(v.2.1.5)*, *tidyr(v.1.3.1)*, *tibble(v.3.2.1)*, *ggplot2(v.3.5.2)*, *tidyverse(v.2.0.0)* and *knitr(v.1.50)*

loaded via a namespace (and not attached): *Rdpack(v.2.6.4)*, *gridExtra(v.2.3)*, *sandwich(v.3.1-1)*, *rlang(v.1.1.6)*, *magrittr(v.2.0.3)*, *multcomp(v.1.4-28)*, *compiler(v.4.5.1)*, *vctrs(v.0.6.5)*, *quadprog(v.1.5-8)*, *pkgconfig(v.2.0.3)*, *fastmap(v.1.2.0)*, *backports(v.1.5.0)*, *labeling(v.0.4.3)*, *rmarkdown(v.2.29)*, *tzdb(v.0.5.0)*, *nloptr(v.2.2.1)*, *xfun(v.0.52)*, *broom(v.1.0.8)*, *parallel(v.4.5.1)*, *R6(v.2.6.1)*, *stringi(v.1.8.7)*, *RColorBrewer(v.1.1-3)*, *parallelly(v.1.45.0)*, *car(v.3.1-3)*, *boot(v.1.3-31)*, *numDeriv(v.2016.8-1.1)*, *estimability(v.1.5.1)*, *Rcpp(v.1.0.14)*, *bookdown(v.0.43)*, *parameters(v.0.24.2)*, *splines(v.4.5.1)*, *timechange(v.0.3.0)*, *tidyselect(v.1.2.1)*, *rstudioapi(v.0.17.1)*, *abind(v.1.4-8)*, *yaml(v.2.3.10)*, *codetools(v.0.2-20)*, *curl(v.6.2.2)*, *listenv(v.0.9.1)*, *lattice(v.0.22-7)*, *withr(v.3.0.2)*, *bayestestR(v.0.15.2)*, *coda(v.0.19-4.1)*, *evaluate(v.1.0.4)*, *survival(v.3.8-3)*, *pillar(v.1.10.2)*, *carData(v.3.0-5)*, *reformulas(v.0.4.0)*, *insight(v.1.1.0)*, *distributional(v.0.5.0)*, *generics(v.0.1.3)*, *hms(v.1.1.3)*, *minqa(v.1.2.8)*, *globals(v.0.18.0)*, *xtable(v.1.8-4)*, *glue(v.1.8.0)*, *emmeans(v.1.11.0)*, *tools(v.4.5.1)*, *ggsignif(v.0.6.4)*, *mvtnorm(v.1.3-3)*, *cowplot(v.1.1.3)*, *grid(v.4.5.1)*, *rbibutils(v.2.3)*, *RobStatTM(v.1.0.11)*, *datawizard(v.1.0.2)*, *nlme(v.3.1-168)*, *Formula(v.1.2-5)*, *cli(v.3.6.5)*, *gtable(v.0.3.6)*, *rstatix(v.0.7.2)*, *digest(v.0.6.37)*, *TH.data(v.1.1-3)*, *farver(v.2.1.2)*, *htmltools(v.0.5.8.1)*, *lifecycle(v.1.0.4)* and *MASS(v.7.3-65)*

5 Supplementary references

Ben-Shachar, M. S., Lüdecke, D., & Makowski, D. (2020). effectsize: Estimation of effect size indices and standardized parameters. *Journal of Open Source Software*, *5*(56), 2815. https://doi.org/10.21105/joss.02815

Caldwell, A. R. (2022). Exploring equivalence testing with the updated toster r package. *PsyArXiv*. https://doi.org/10.31234/osf.io/ty8de

Dancho, M., & Vaughan, D. (2025). *Tidyquant: Tidy quantitative financial analysis* [R package version 1.0.11]. https://doi.org/10.32614/CRAN.package.tidyquant

Kay, M. (2024). ggdist: Visualizations of distributions and uncertainty in the grammar of graphics. *IEEE Transactions on Visualization and Computer Graphics*, *30*(1), 414–424. https://doi.org/10.1109/TVCG.2023.3327195

Kay, M. (2025). *ggdist: Visualizations of distributions and uncertainty* [R package version 3.3.3]. https://doi.org/10.5281/zenodo.3879620

Kuznetsova, A., Brockhoff, P. B., & Christensen, R. H. B. (2017). lmerTest package: Tests in linear mixed effects models. *Journal of Statistical Software*, *82*(13), 1–26. https://doi.org/10.18637/jss.v082.i13

Lakens, D. (2017). Equivalence tests: A practical primer for t-tests, correlations, and meta-analyses. *Social Psychological and Personality Science*, *1*, 1–8. https://doi.org/10.1177/1948550617697177

Vaughan, D., & Dancho, M. (2022). *Furrr: Apply mapping functions in parallel using futures* [R package version 0.3.1]. https://doi.org/10.32614/CRAN.package.furrr

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

Wickham, H. (2023). *Stringr: Simple, consistent wrappers for common string operations* [R package version 1.5.1]. https://doi.org/10.32614/CRAN.package.stringr

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 1.1.4]. https://doi.org/10.32614/CRAN.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

Xie, Y. (2025). *Knitr: A general-purpose package for dynamic report generation in R* [R package version 1.50]. https://yihui.org/knitr/