Posterior Predictive Checks: Reaction Time Example

brms Workshop

Posterior Predictive Checks for RT Data

After fitting your model, validate that it generates data similar to what you observed.

Why Posterior Predictive Checks Matter

Posterior predictive checks answer: "If I were to generate new data from my fitted model, would it look like my actual data?" This validates that your model has captured the essential structure of your data.

Setup

```r
library(brms)
library(tidyverse)
library(bayesplot)

# Set seed for reproducibility
set.seed(42)
```

Load or Fit Model

```r
# Create example RT data (same as in prior checks for consistency)
n_subj <- 20
n_trials <- 50
n_items <- 30
```

```
rt_data <- expand.grid(
  trial = 1:n_trials,
  subject = 1:n_subj,
  item = 1:n_items
) %>%
  filter(row_number() <= n_subj * n_trials * 3) %>%
  mutate(
    condition = rep(c("A", "B"), length.out = n()),
    log_rt = rnorm(n(), mean = 6, sd = 0.3) +
             (condition == "B") * 0.15 +
             rnorm(n(), mean = 0, sd = 0.1),
    rt = exp(log_rt)
  )

# Define priors
rt_priors <- c(
  prior(normal(6, 1.5), class = Intercept),
  prior(normal(0, 0.5), class = b),
  prior(exponential(1), class = sigma),
  prior(exponential(1), class = sd),
  prior(lkj(2), class = cor)
)

# Check if model exists, otherwise fit it
model_file <- "fits/fit_rt.rds"
if (file.exists(model_file)) {
  cat("Loading saved model from:", model_file, "\n")
  fit_rt <- readRDS(model_file)
} else {
  cat("Fitting model (this may take a while)...\n")
  cat("Note: Model fitting requires significant computational resources.\n")
  cat("Consider fitting the model separately and saving to fits/fit_rt.rds\n\n")

  # Fit with reduced complexity for demonstration
  fit_rt <- brm(
    log_rt ~ condition + (1 + condition | subject) + (1 | item),
    data = rt_data,
    family = gaussian(),
    prior = rt_priors,
    chains = 2,  # Reduced for faster fitting
    iter = 1000,
    cores = 2,
```

```
    backend = "rstan",
    refresh = 0
  )
  # Save for future use
  dir.create("fits", showWarnings = FALSE, recursive = TRUE)
  saveRDS(fit_rt, model_file)
  cat("Model saved to:", model_file, "\n")
}
```

```
Loading saved model from: fits/fit_rt.rds
```

```
# Display model summary
cat("\n=== Model Summary ===\n")
```

```

=== Model Summary ===
```

```
print(summary(fit_rt))
```

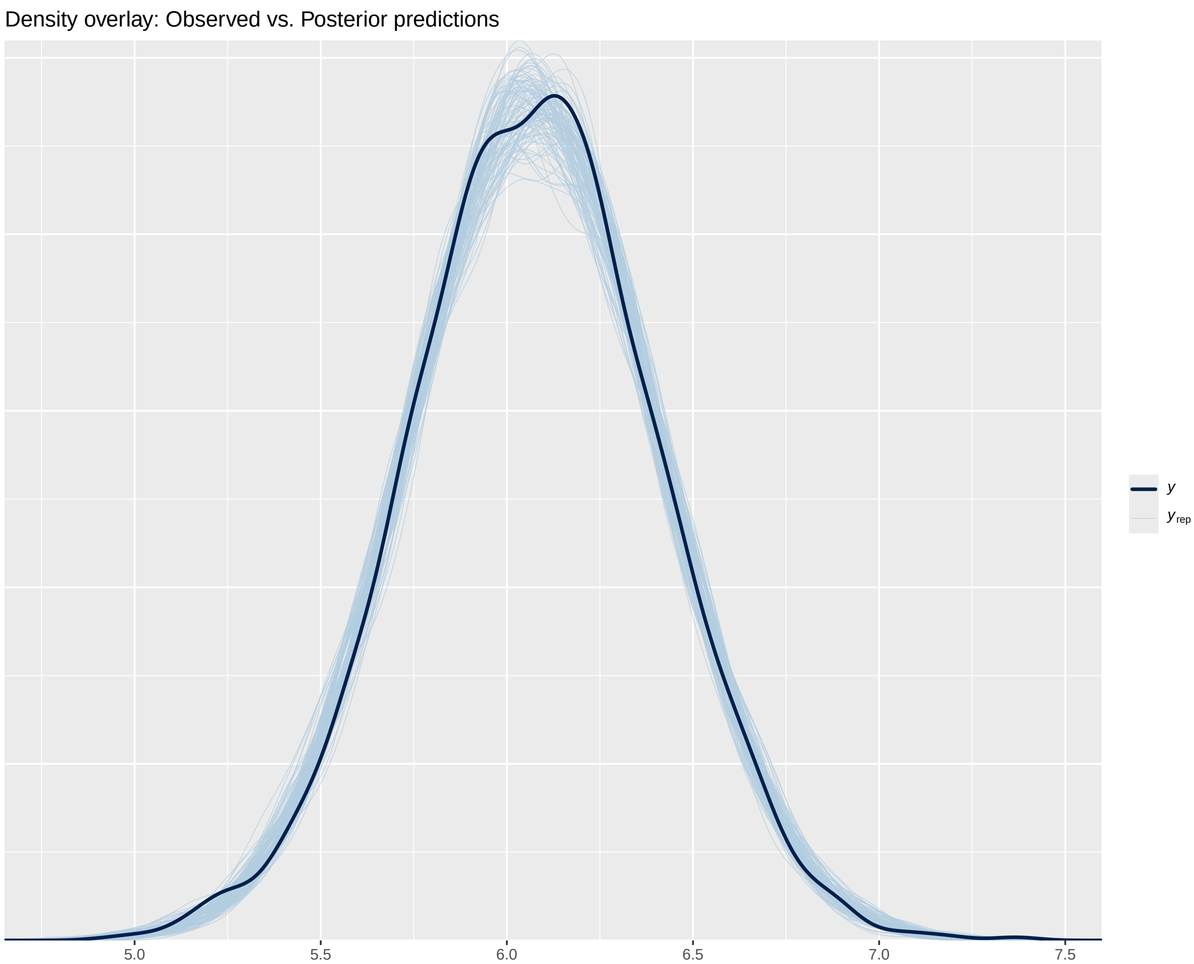
```
 Family: gaussian
  Links: mu = identity
Formula: log_rt ~ condition + (1 + condition | subject) + (1 | item)
   Data: rt_data (Number of observations: 3000)
  Draws: 2 chains, each with iter = 1000; warmup = 500; thin = 1;
         total post-warmup draws = 1000

Multilevel Hyperparameters:
~item (Number of levels: 3)
              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)     0.02      0.02     0.00     0.07 1.01      392      323

~subject (Number of levels: 20)
                          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
sd(Intercept)                 0.02      0.01     0.00     0.04 1.00      432
sd(conditionB)                0.02      0.01     0.00     0.06 1.00      343
cor(Intercept,conditionB)    -0.07      0.46    -0.82     0.83 1.01      513
                          Tail_ESS
sd(Intercept)                  528
sd(conditionB)                 461
cor(Intercept,conditionB)      591
```

```
Regression Coefficients:
           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept      5.99      0.01     5.96     6.02 1.00      700      436
conditionB     0.16      0.01     0.13     0.19 1.00     1339      699

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     0.32      0.00     0.31     0.33 1.01     2149      698

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

Basic Posterior Predictive Checks

Visual Checks

```r
# Default: density overlay of observed vs. simulated data
pp_check(fit_rt, ndraws = 100) +
  ggtitle("Density overlay: Observed vs. Posterior predictions")
```



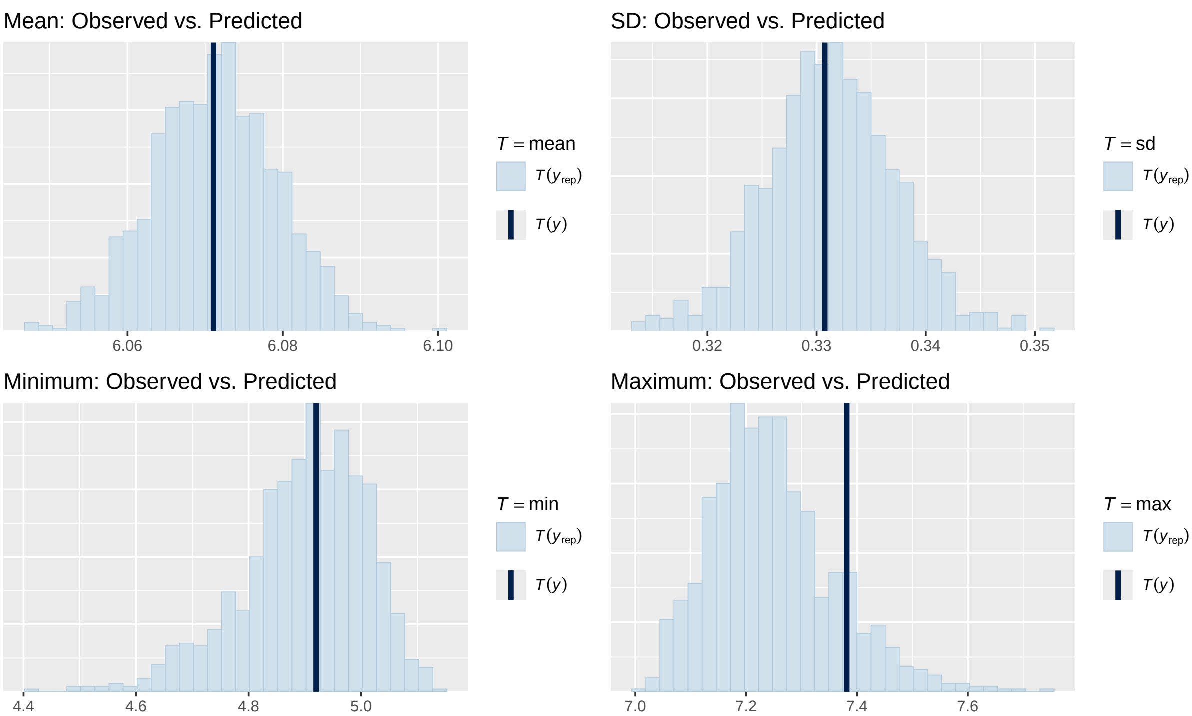
Interpretation:

- **Blue line** (observed data) should be among the dark lines (posterior predictions)
- If blue line is far from the bundle → model missed something important
- Small discrepancies are normal; large ones suggest model misspecification

Check Specific Statistics

```
# Did we get the mean right?
p1 <- pp_check(fit_rt, type = "stat", stat = "mean") +
  ggtitle("Mean: Observed vs. Predicted")

# Did we get the spread right?
p2 <- pp_check(fit_rt, type = "stat", stat = "sd") +
```

```r
  ggtitle("SD: Observed vs. Predicted")

# Extreme values?
p3 <- pp_check(fit_rt, type = "stat", stat = "min") +
  ggtitle("Minimum: Observed vs. Predicted")

p4 <- pp_check(fit_rt, type = "stat", stat = "max") +
  ggtitle("Maximum: Observed vs. Predicted")

# Display all plots
library(patchwork)
(p1 | p2) / (p3 | p4)
```



Extract and Analyze Posterior Predictions

```r
# Draw from posterior predictive distribution
post_pred <- posterior_predict(fit_rt, ndraws = 1000)
dim(post_pred)  # 1000 draws × n observations
```

```
[1] 1000 3000
```

```r
cat("\nDimensions of posterior predictions:\n")
```

```
Dimensions of posterior predictions:
```

```r
cat("Draws:", nrow(post_pred), "\n")
```

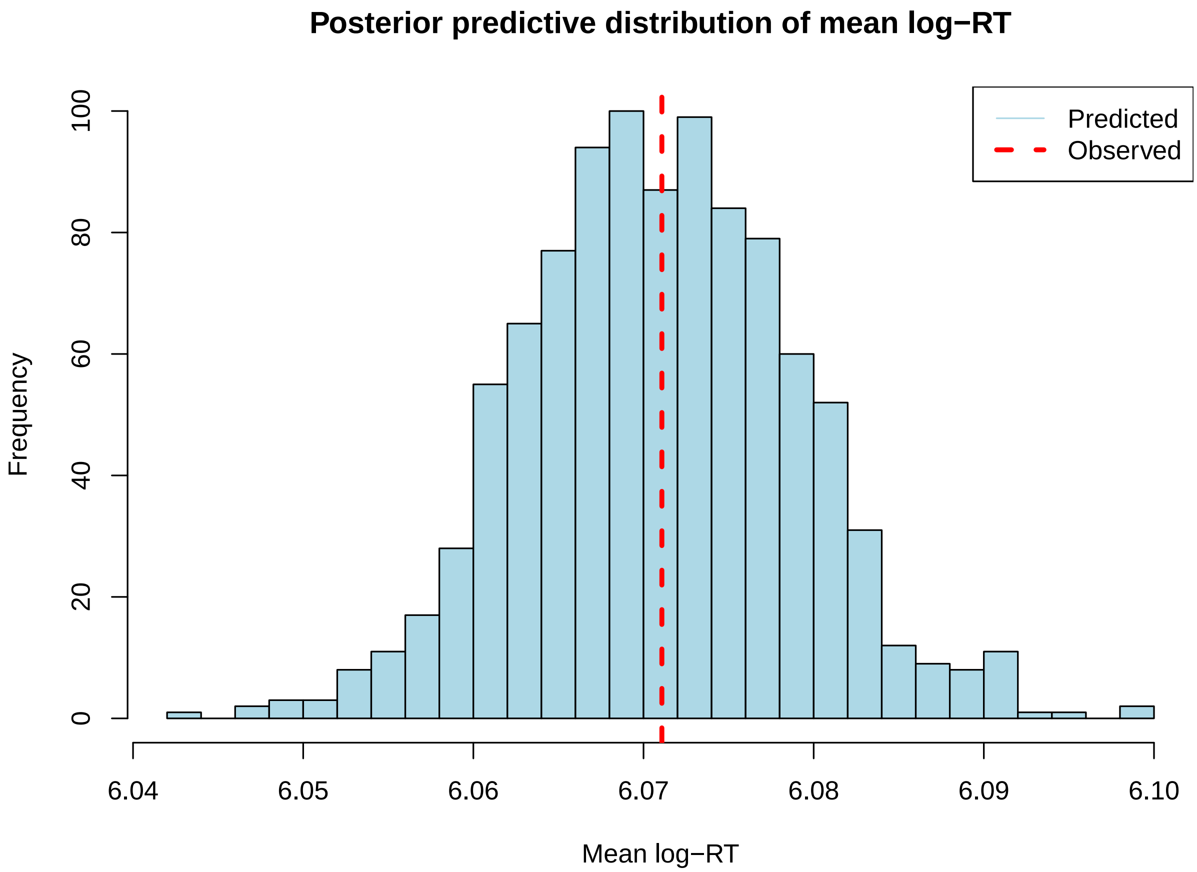
```
Draws: 1000
```

```r
cat("Observations:", ncol(post_pred), "\n")
```

```
Observations: 3000
```

Compare Observed vs. Predicted

```r
# Compare observed vs. predicted means
obs_mean <- mean(rt_data$log_rt)
pred_mean <- apply(post_pred, 1, mean)

hist(pred_mean,
     main = "Posterior predictive distribution of mean log-RT",
     xlab = "Mean log-RT",
     col = "lightblue",
     breaks = 30)
abline(v = obs_mean, col = "red", lwd = 3, lty = 2)
legend("topright",
       legend = c("Predicted", "Observed"),
       col = c("lightblue", "red"),
       lwd = c(1, 3),
       lty = c(1, 2))
```




```r
cat("\nObserved mean log-RT:", round(obs_mean, 3), "\n")
```

```
Observed mean log-RT: 6.071
```

```r
cat("Predicted mean log-RT (median):", round(median(pred_mean), 3), "\n")
```

```
Predicted mean log-RT (median): 6.071
```

```r
cat("95% CI for predicted mean:",
    round(quantile(pred_mean, c(0.025, 0.975)), 3), "\n")
```

```
95% CI for predicted mean: 6.056 6.087
```

Check Posterior Predictive Intervals

```r
# Check 95% posterior predictive interval
post_pred_interval <- apply(post_pred, 2, quantile, c(0.025, 0.975))

# Roughly 95% of observed values should fall within their interval
coverage <- mean(rt_data$log_rt > post_pred_interval[1,] &
                 rt_data$log_rt < post_pred_interval[2,])

cat("\nPosterior Predictive Interval Coverage:\n")
```

```
Posterior Predictive Interval Coverage:
```

```r
cat("Proportion of observations within 95% interval:", round(coverage, 3), "\n")
```

```
Proportion of observations within 95% interval: 0.951
```

```r
cat("Expected: ~0.95\n")
```

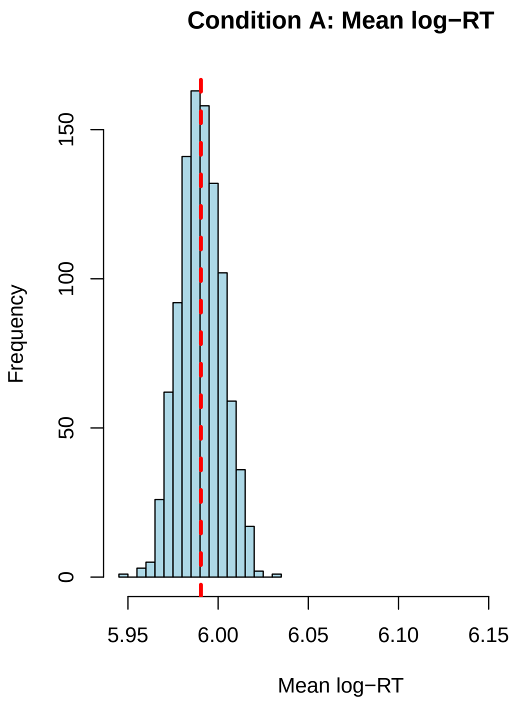
```
Expected: ~0.95
```

```r
if (coverage < 0.90) {
  cat("\n  Warning: Coverage is lower than expected.\n")
  cat("Model may be overconfident or missing important structure.\n")
} else if (coverage > 0.98) {
  cat("\n  Warning: Coverage is higher than expected.\n")
  cat("Model may be too uncertain or overfitted.\n")
} else {
  cat("\n  Coverage looks good!\n")
}
```

```
  Coverage looks good!
```

Check by Condition

```r
# Group predictions by condition
rt_data_cond <- rt_data %>%
  group_by(condition) %>%
  summarise(
    obs_mean = mean(log_rt),
    obs_sd = sd(log_rt)
  )

# Get posterior predictions by condition
post_pred_A <- posterior_predict(fit_rt,
                                 newdata = filter(rt_data, condition == "A"),
                                 ndraws = 1000)
post_pred_B <- posterior_predict(fit_rt,
                                 newdata = filter(rt_data, condition == "B"),
                                 ndraws = 1000)

pred_mean_A <- apply(post_pred_A, 1, mean)
pred_mean_B <- apply(post_pred_B, 1, mean)

# Plot comparison
par(mfrow = c(1, 2))

hist(pred_mean_A,
     main = "Condition A: Mean log-RT",
     xlab = "Mean log-RT",
     col = "lightblue",
     breaks = 30,
     xlim = range(c(pred_mean_A, pred_mean_B)))
abline(v = rt_data_cond$obs_mean[1], col = "red", lwd = 3, lty = 2)

hist(pred_mean_B,
     main = "Condition B: Mean log-RT",
     xlab = "Mean log-RT",
     col = "lightgreen",
     breaks = 30,
     xlim = range(c(pred_mean_A, pred_mean_B)))
abline(v = rt_data_cond$obs_mean[2], col = "red", lwd = 3, lty = 2)
```




```r
cat("\nBy Condition:\n")
```

```
By Condition:
```

```r
cat("Condition A - Observed:", round(rt_data_cond$obs_mean[1], 3), "\n")
```

```
Condition A - Observed: 5.99
```

```r
cat("Condition A - Predicted:", round(median(pred_mean_A), 3), "\n")
```

```
Condition A - Predicted: 5.99
```

```r
cat("Condition B - Observed:", round(rt_data_cond$obs_mean[2], 3), "\n")
```

```
Condition B - Observed: 6.152
```

```r
cat("Condition B - Predicted:", round(median(pred_mean_B), 3), "\n")
```

```
Condition B - Predicted: 6.151
```

Summary

Key Diagnostics Checked

1. **Visual inspection** - Observed data overlaps with posterior predictions
2. **Mean** - Central tendency captured correctly
3. **SD** - Spread of data captured correctly
4. **Extreme values** - Min/max are reasonable
5. **Predictive intervals** - Coverage is appropriate
6. **By condition** - Model captures group differences

Common Problems and Solutions

Problem	Diagnosis	Solution
Model predictions too narrow	SD of posterior predictions < SD of data	Relax priors, check formula
Model predictions too wide	SD of posterior predictions » SD of data	Tighten priors, add more structure
Misses condition effects	Mean differs dramatically by condition	Add condition × random effect interaction
Extreme value mismatch	Min/max far from observed	Check for outliers, consider robust models

Next Steps

If posterior predictive checks reveal problems:

1. **Adjust model formula** - Add missing predictors or interactions
2. **Revise priors** - May be too restrictive or too vague
3. **Consider alternative families** - E.g., Student's t for robust modeling
4. **Check for outliers** - May need to handle separately

```r
sessionInfo()
```

```
R version 4.4.1 (2024-06-14)
Platform: x86_64-pc-linux-gnu
Running under: Ubuntu 22.04.5 LTS

Matrix products: default
BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.20.so;  LAPACK version 3

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

time zone: Etc/UTC
tzcode source: system (glibc)

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

other attached packages:
 [1] patchwork_1.3.2  bayesplot_1.14.0 lubridate_1.9.3  forcats_1.0.0
 [5] stringr_1.5.1    dplyr_1.1.4      purrr_1.0.2      readr_2.1.5
 [9] tidyr_1.3.1      tibble_3.2.1     ggplot2_4.0.0    tidyverse_2.0.0
[13] brms_2.23.0      Rcpp_1.0.13

loaded via a namespace (and not attached):
 [1] gtable_0.3.6          tensorA_0.36.2.1      xfun_0.54
 [4] QuickJSR_1.8.1        inline_0.3.21         lattice_0.22-6
 [7] tzdb_0.4.0            vctrs_0.6.5           tools_4.4.1
[10] generics_0.1.3        stats4_4.4.1          parallel_4.4.1
[13] fansi_1.0.6           pkgconfig_2.0.3       Matrix_1.7-0
[16] checkmate_2.3.3       RColorBrewer_1.1-3    S7_0.2.0
[19] distributional_0.5.0  RcppParallel_5.1.11-1 lifecycle_1.0.4
[22] compiler_4.4.1        farver_2.1.2          Brobdingnag_1.2-9
[25] tinytex_0.53          codetools_0.2-20      htmltools_0.5.8.1
[28] yaml_2.3.10           pillar_1.9.0          StanHeaders_2.32.10
[31] bridgesampling_1.1-2  abind_1.4-8           nlme_3.1-164
[34] posterior_1.6.1.9000  rstan_2.32.7          tidyselect_1.2.1
[37] digest_0.6.37         mvtnorm_1.3-3         stringi_1.8.4
[40] reshape2_1.4.4        labeling_0.4.3        fastmap_1.2.0
```

```
[43] grid_4.4.1           cli_3.6.5           magrittr_2.0.3
[46] loo_2.8.0            pkgbuild_1.4.8      utf8_1.2.4
[49] withr_3.0.2          scales_1.4.0        backports_1.5.0
[52] estimability_1.5.1   timechange_0.3.0    rmarkdown_2.30
[55] matrixStats_1.5.0    emmeans_2.0.0       gridExtra_2.3
[58] hms_1.1.3            coda_0.19-4.1       evaluate_1.0.1
[61] knitr_1.50           rstantools_2.5.0    rlang_1.1.6
[64] xtable_1.8-4         glue_1.8.0          jsonlite_1.8.9
[67] plyr_1.8.9           R6_2.5.1
```