Posterior Predictive Checks: Reaction Time Example

Bayesian Mixed Effects Models with brms for Linguists

Job Schepens

2025-12-17

Table of contents

1 Posterior Predictive Checks for RT Data

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

1.1 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.

1.2 Setup

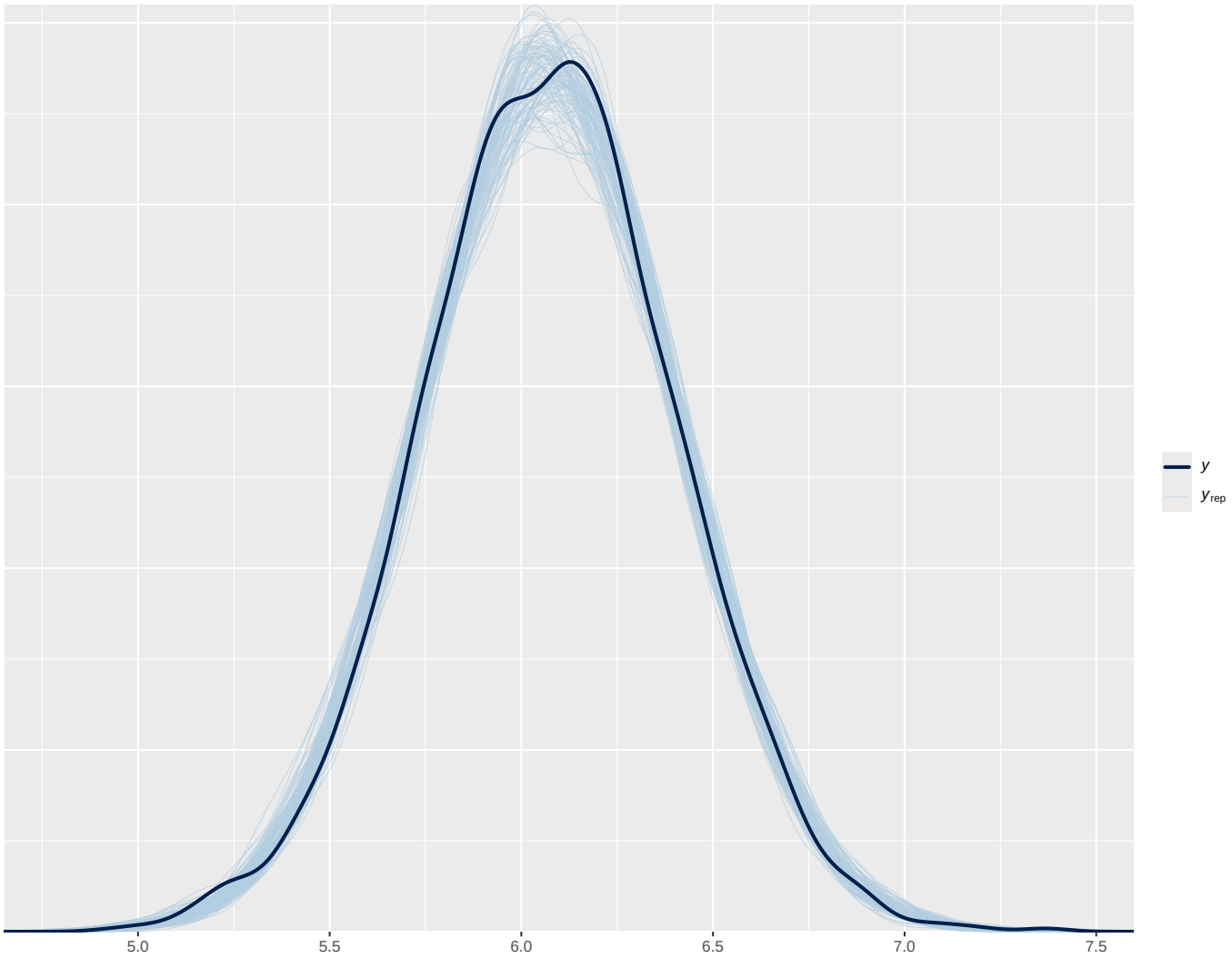
1.3 Load or Fit Model

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


=== Model Summary ===

 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).
```

1.4 Basic Posterior Predictive Checks

1.4.1 Visual Checks

Density overlay: Observed vs. Posterior predictions

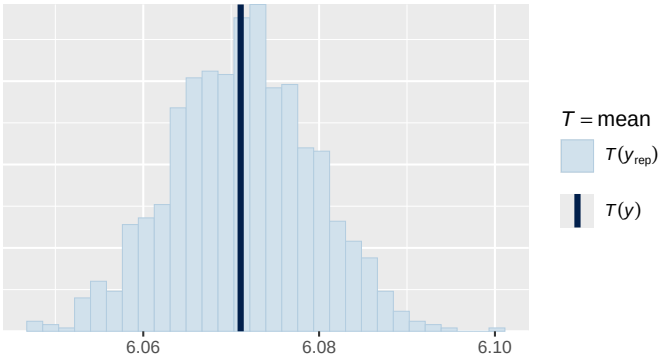


Interpretation:

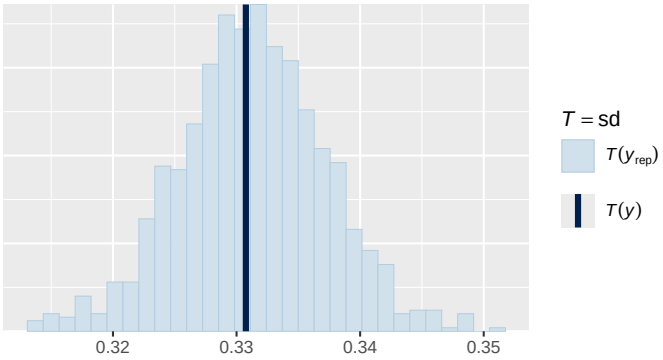
- **Black line** (y = observed data) should be among the blue lines (`yrep` = posterior predictions)
- If black line is far from the bundle → model missed something important
- Small discrepancies are normal; large ones suggest model misspecification

1.4.2 Check Specific Statistics

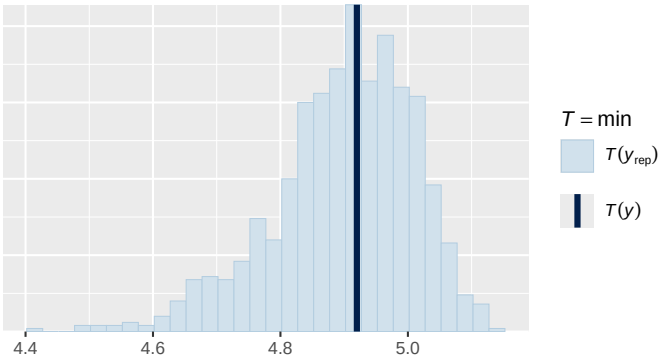
Mean: Observed vs. Predicted



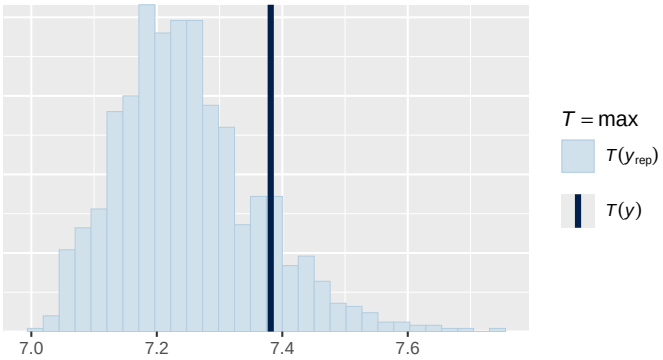
SD: Observed vs. Predicted



Minimum: Observed vs. Predicted



Maximum: Observed vs. Predicted

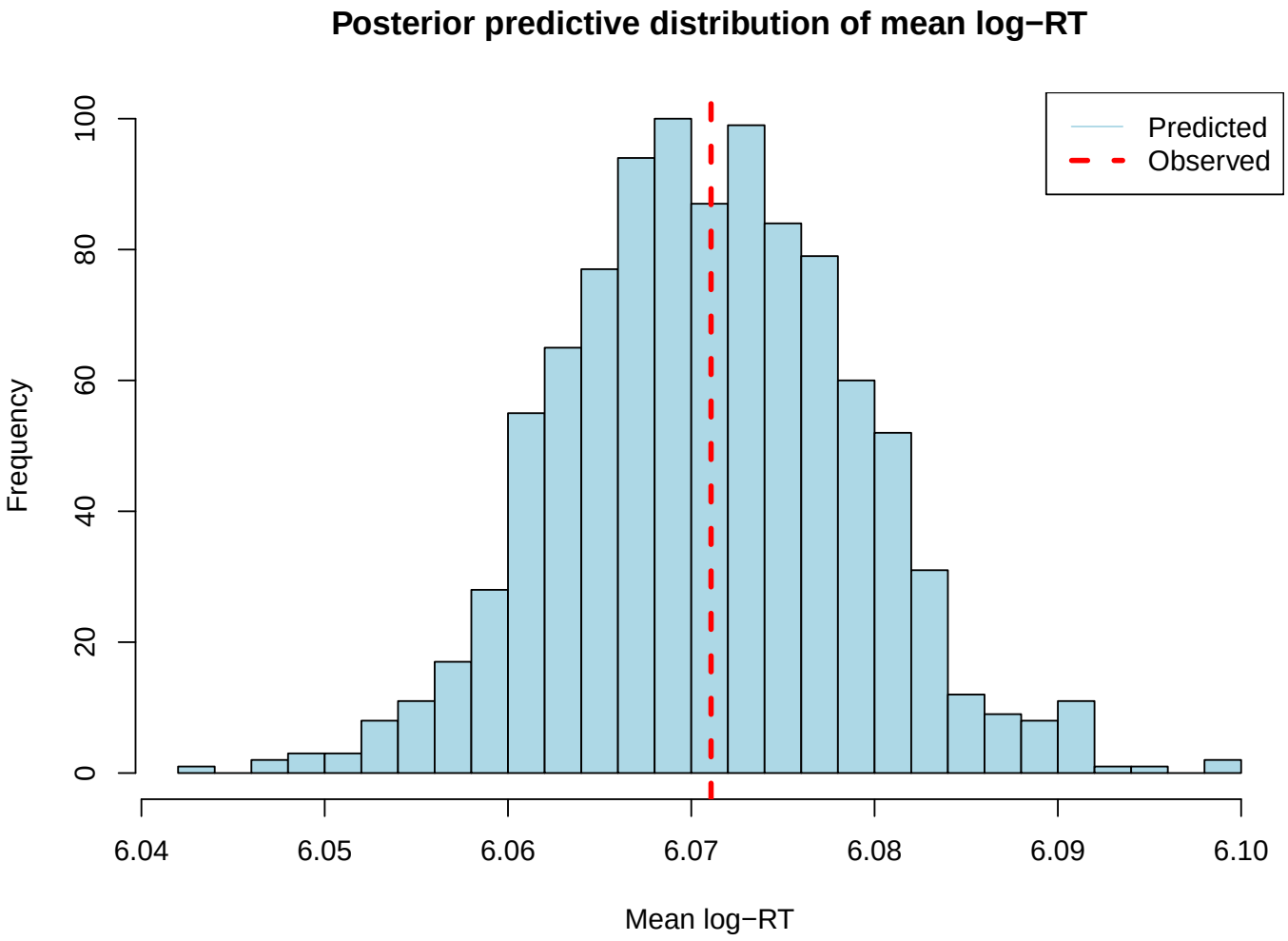


1.5 Extract and Analyze Posterior Predictions

```
[1] 1000 3000
```

```
Dimensions of posterior predictions:
Draws: 1000
Observations: 3000
```

1.5.1 Compare Observed vs. Predicted

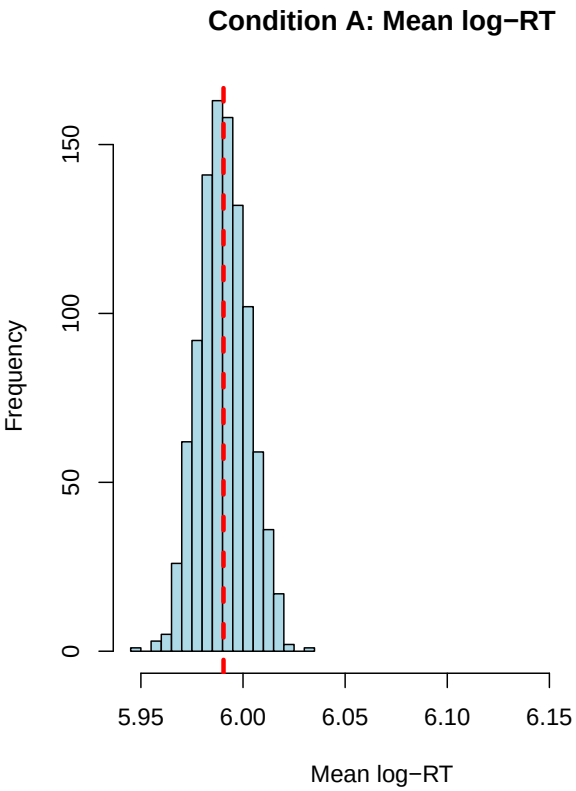


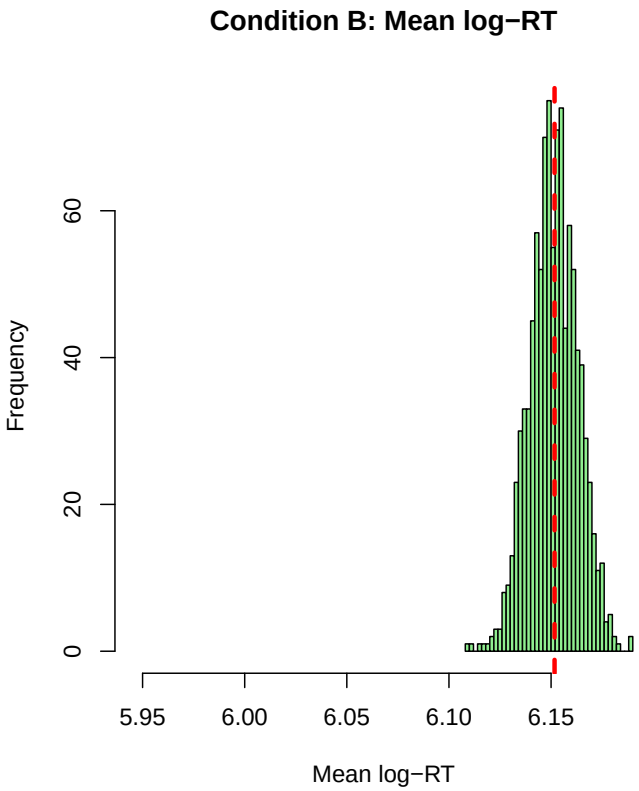

```
Observed mean log-RT: 6.071

Predicted mean log-RT (median): 6.071

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

1.5.2 Check Posterior Predictive Intervals

```
Posterior Predictive Interval Coverage:

Proportion of observations within 95% interval: 0.951

Expected: ~0.95

  Coverage looks good!
```

1.6 Check by Condition





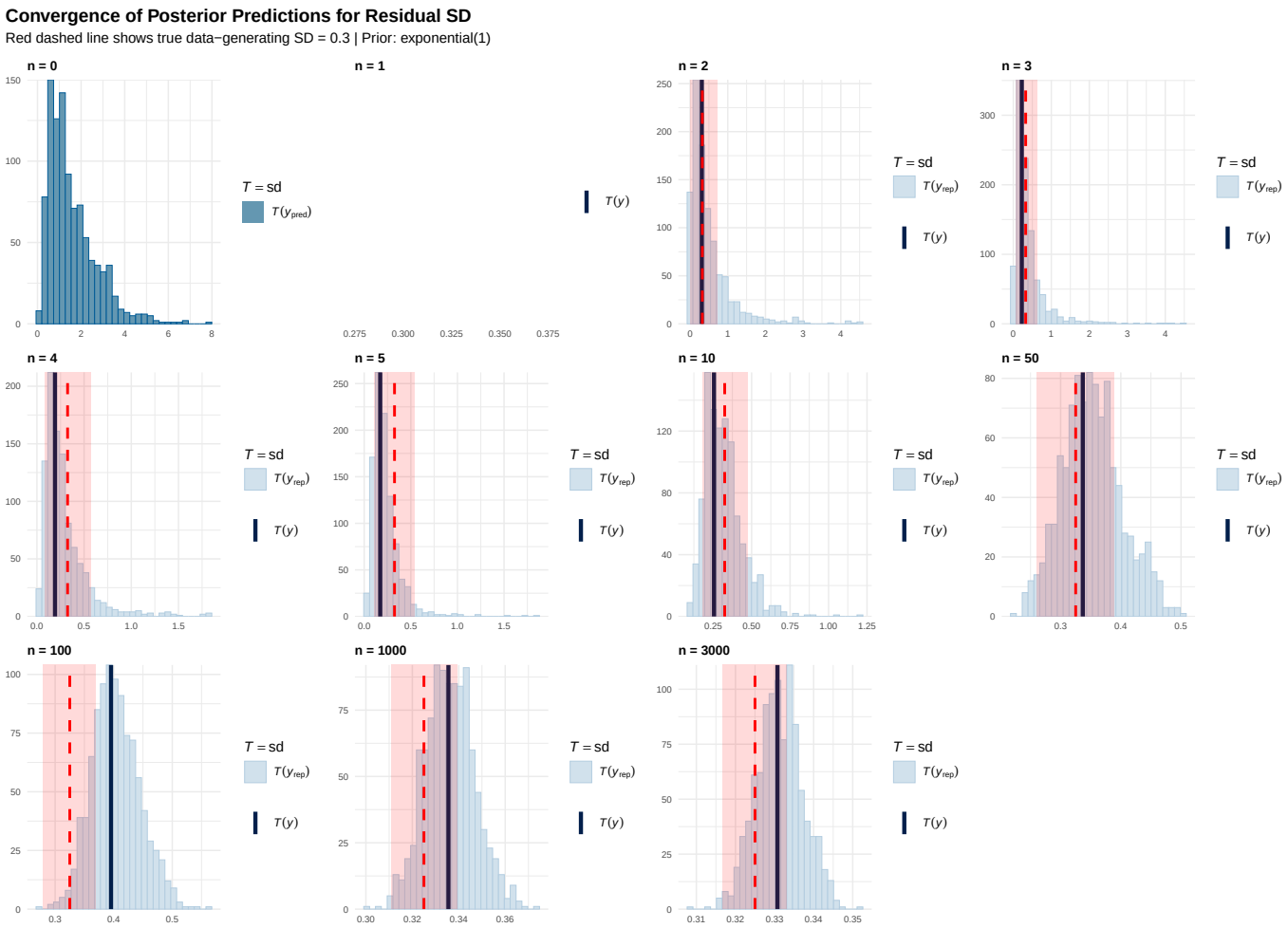

```
By Condition:

Condition A - Observed: 5.99

Condition A - Predicted: 5.99

Condition B - Observed: 6.152

Condition B - Predicted: 6.151
```

1.7 How Data Quantity Affects Posterior Predictions

This section demonstrates how the posterior predictive distribution converges to the true data-generating value as we add more observations. We'll fit models with increasing amounts of data and examine how well each recovers the residual SD.

```
Fitting models with varying sample sizes...

This will take a few minutes. Progress:

  [ 1/11] n =    0 observations... (loaded from cache)
  [ 2/11] n =    1 observations... (loaded from cache)
  [ 3/11] n =    2 observations... (loaded from cache)
  [ 4/11] n =    3 observations... (loaded from cache)
  [ 5/11] n =    4 observations... (loaded from cache)
  [ 6/11] n =    5 observations... (loaded from cache)
  [ 7/11] n =   10 observations... (loaded from cache)
```

```
[ 8/11] n =   50 observations... (loaded from cache)
[ 9/11] n =  100 observations... (loaded from cache)
[10/11] n = 1000 observations... (loaded from cache)
[11/11] n = 3000 observations... (loaded from cache)

Generating posterior predictive checks...
```



1.7.1 Understanding the Plots

Each subplot shows a posterior predictive check for the standard deviation statistic:

Visual Elements:

- **Light blue histogram (T(y_pred))**: Distribution of SD values from *predicted* datasets
  - Each bar represents how often the model predicts a certain SD value
  - Generated by: (1) drawing parameter values from the posterior, (2) simulating new datasets, (3) calculating SD of each simulated dataset
  - Wide spread = model is uncertain about what SD to expect
- **Dark blue vertical line (T(y))**: SD of the *actual observed* data
  - This is the single SD value calculated from your real data
  - Should ideally fall within the light blue distribution

  - If it's far outside → model is misspecified
- **Red dashed line (0.325)**: True total SD of the data-generating process
  - Includes both residual variance (√(0.3² + 0.1²) 0.316) and fixed effect variance from balanced conditions
  - Shows the "ground truth" SD of the raw data
  - In real analysis, you wouldn't know this value
  - Here, it helps us verify the model is learning the right thing
- **Light red shaded band**: Where T(y) typically falls when sampling from the true model
  - Shows 95% interval for a single observed SD from n draws with total SD 0.325
  - Derived from chi-squared distribution of sample variance
  - Dark blue line (T(y)) falling in this band = consistent with true model
  - Band gets narrower with larger n → less sampling variation
  - **Different from light blue histogram**: Red band = where one T(y) falls; Blue histogram = distribution of T(y_pred) from posterior

What convergence looks like:

- **Wide histogram**: Model is uncertain (small sample size or strong prior)
- **Narrow histogram**: Model is confident (large sample size)
- **Histogram centered on red line**: Model correctly estimates true value
- **Dark blue line inside histogram**: Model predictions match observed data

1.7.2 Interpretation

Key observations from the progression:

1. **n = 0 (Prior only)**: Wide uncertainty, no learning from data yet. Prior allows SD anywhere from 0 to ~3.
2. **n = 1-5 (Very few observations)**: Posterior still heavily influenced by prior. High uncertainty remains. T(y) can be far from truth.
3. **n = 10-50 (Small samples)**: Beginning to converge toward true value (0.3), but still substantial uncertainty. Histogram narrowing.
4. **n = 100-1000 (Medium samples)**: Clear convergence visible. Posterior concentrates around 0.3. T(y) aligns with T(y_pred).
5. **n = 3000 (Full data)**: Tight posterior distribution centered on true value. Prior influence minimal. Very narrow histogram.

This demonstrates:

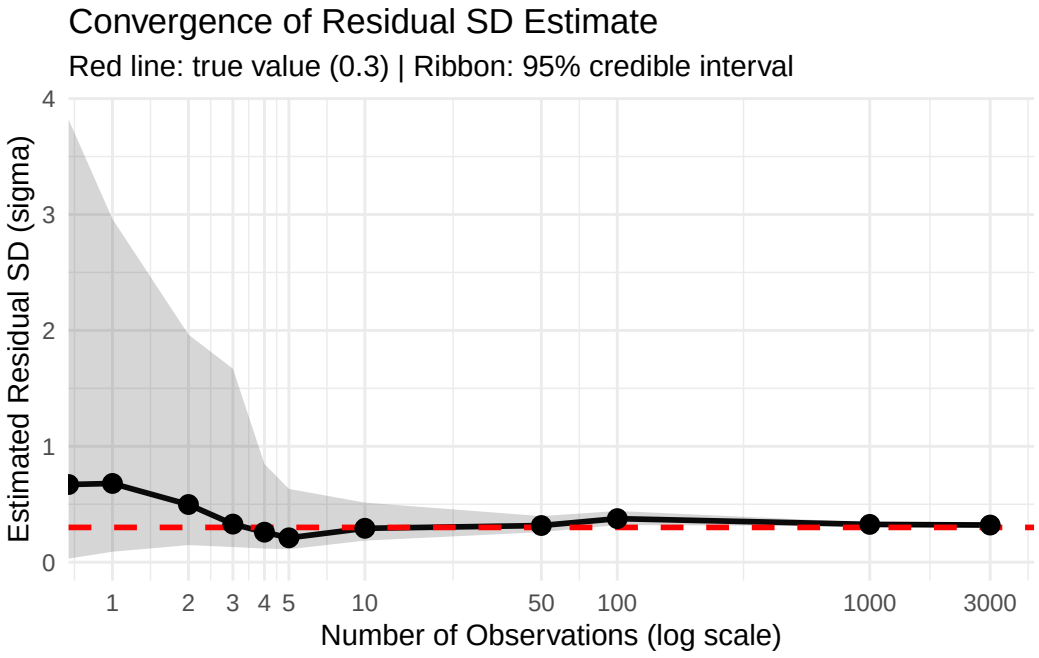
- **Prior dominance** with very little data (n < 10)
- **Gradual transition** where data and prior both matter (n = 10-100)
- **Data dominance** with sufficient observations (n > 100)
- The importance of **sample size** for overcoming prior assumptions

An important cautionary case: n = 100

Notice at n = 100, T(y) falls **outside** the red sampling bands. This is a Type I error scenario (occurring ~5% of the time by chance). What are the consequences?

- **Overestimated variance**: The model learns 0.39 from this unlucky sample, substantially higher than the true 0.325
- **Inflated uncertainty**: Credible intervals for all parameters become wider than they should be
- **Biased effect size inference**: The condition effect estimate remains unbiased (~0.15), BUT its standardized effect size (Cohen's d = 0.15/) will be underestimated because we're dividing by an inflated
- **Conservative inference**: Hypothesis tests become less powerful - we're more likely to fail to detect real effects
- **Misleading predictions**: Prediction intervals will be too wide, suggesting more uncertainty about future observations than warranted

Key insight: Even when the model "correctly" learns from the data (blue histogram centered on T(y)), if the observed data arose from sampling variability (T(y) outside red bands), downstream inferences about effect sizes and standardized parameters will be systematically biased. As sample size increases (n = 1000, 3000), the probability of such extreme sampling variation diminishes, and estimates converge to truth.



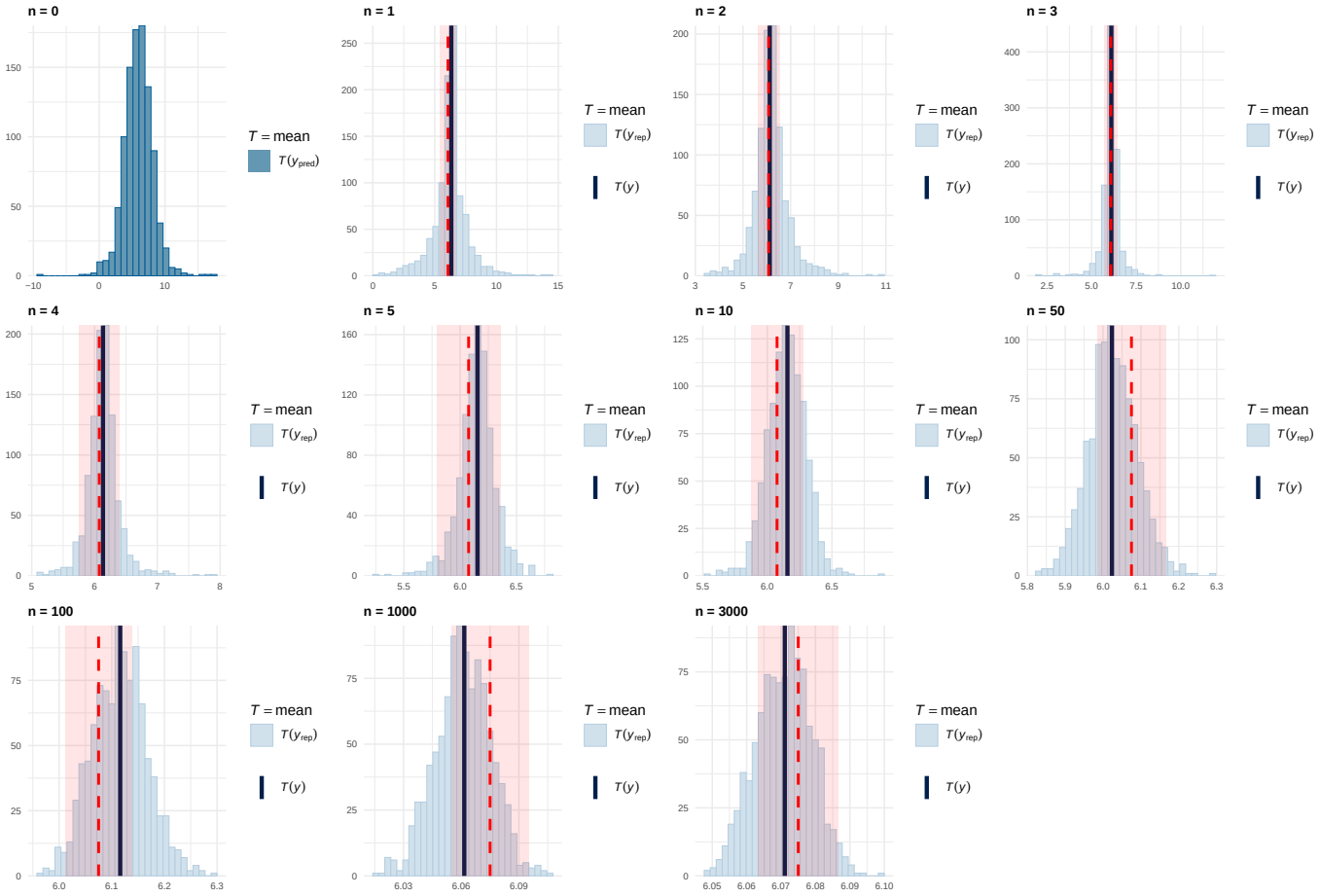

```
=== Convergence Summary ===

# A tibble: 11 x 5
   n_obs median  lower upper  width
   <dbl>  <dbl>  <dbl> <dbl>  <dbl>
 1     0  0.670 0.0310 3.82  3.79
 2     1  0.679 0.0902 2.96  2.87
 3     2  0.498 0.148  1.96  1.82
 4     3  0.329 0.131  1.67  1.54
 5     4  0.259 0.118  0.847 0.730
 6     5  0.210 0.112  0.632 0.521
 7    10  0.292 0.186  0.514 0.328
 8    50  0.316 0.259  0.397 0.138
```

```
 9    100  0.376 0.320  0.442 0.122
10   1000  0.326 0.313  0.343 0.0294
11   3000  0.320 0.313  0.328 0.0152
```

Practical takeaway: With the exponential(1) prior and ~100 observations, the posterior estimate becomes quite accurate. The wide Intercept prior `normal(6, 1.5)` doesn't prevent learning the correct residual SD because they model different aspects of the data.

1.8 Convergence Analysis: Population Mean

Now let's examine how the population mean estimate converges with increasing data:



1.8.1 Understanding These Plots

Each subplot shows a posterior predictive check for the mean statistic:

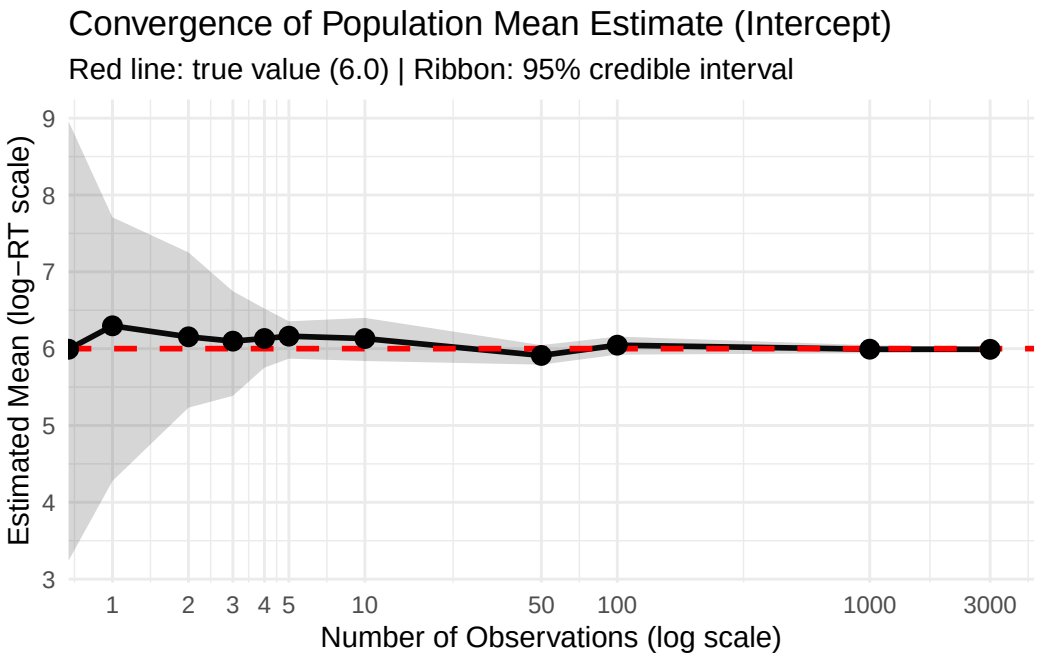
Visual Elements:

- **Light blue histogram (T(y_pred))**: Distribution of mean values from *predicted* datasets
  - Shows all possible mean values the model thinks are plausible
  - Generated by drawing from posterior → simulating datasets → calculating mean of each
  - Width indicates uncertainty about the true mean

- **Dark blue vertical line (T(y))**: Mean of the *actual observed* data
  - The single mean value from your real dataset
  - Should fall within the light blue distribution if model is well-calibrated
- **Red dashed line (6.0)**: True data-generating mean
  - Ground truth used to create the simulated data (log-RT = 6 403ms)
  - Helps verify the model recovers the correct parameter
- **Light red shaded band**: Where T(y) typically falls when sampling from the true model
  - Shows 95% interval for a single observed mean from n draws with =6, =0.3
  - Based on normal distribution: sample mean ~ N(6, 0.3^2/n)
  - SE = 0.3/√n, so band narrows much faster than for SD
  - Dark blue line (T(y)) outside this band → your particular sample was unusual
  - **Key insight**: At n=3000, T(y) 6.0 lands in red band = data consistent with true parameters

Comparison with SD plots:

- **Mean converges faster** than SD (fewer observations needed)
- **Mean estimation** is directly informed by all observations
- **SD estimation** requires enough data to observe variability




```
=== Mean Convergence Summary ===

# A tibble: 11 x 5
   n_obs median lower upper  width
   <dbl>  <dbl> <dbl> <dbl>  <dbl>
 1     0   5.99  3.24  8.96 5.72
 2     1   6.30  4.28  7.71 3.43
 3     2   6.15  5.23  7.25 2.02
 4     3   6.10  5.39  6.75 1.36
 5     4   6.13  5.75  6.52 0.769
 6     5   6.16  5.87  6.36 0.486
```

```
 7    10   6.13  5.84  6.40 0.561
 8    50   5.91  5.79  6.05 0.254
 9   100   6.04  5.92  6.16 0.236
10  1000   5.99  5.94  6.04 0.105
11  3000   5.99  5.96  6.02 0.0546


=== Comparison: Mean vs. SD Convergence ===

Credible interval width at n=10:

  Mean: 0.561

  SD:   0.328

Credible interval width at n=100:

  Mean: 0.236

  SD:   0.122

→ Mean converges faster: CI narrows more quickly with increasing data
```

1.8.2 Key Insight: Why Different Priors Don't Interfere

This analysis demonstrates the original question: **Why does `normal(6, 1.5)` for the Intercept not prevent learning `sigma = 0.3`?**

They model different aspects:

1. **Intercept prior `normal(6, 1.5)`**: Uncertainty about the *population mean*
   - "I think the average log-RT is around 6, but could be anywhere from ~3 to ~9"
   - This is uncertainty ABOUT THE MEAN, not about residual variation
2. **Sigma prior `exponential(1)`**: Uncertainty about *residual variation*
   - "I think observations scatter around their predictions with some SD"
   - This is observation-level noise, independent of the mean

With sufficient data (n > 100):

- Both parameters converge to their true values
- The wide prior on the Intercept (SD=1.5) doesn't "leak into" the sigma estimate
- Each parameter is identified by different aspects of the data

1.9 Summary

1.9.1 Key Diagnostics Checked

☒ **Visual inspection** - Observed data overlaps with posterior predictions
☒ **Mean** - Central tendency captured correctly
☒ **SD** - Spread of data captured correctly
☒ **Extreme values** - Min/max are reasonable
☒ **Predictive intervals** - Coverage is appropriate
☒ **By condition** - Model captures group differences
☒ **Data quantity effect** - Convergence from prior to data-driven estimates

1.9.2 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

1.9.3 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 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 3.10.0

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        processx_3.8.4        inline_0.3.21
 [7] lattice_0.22-6        tzdb_0.4.0            ps_1.8.1
[10] vctrs_0.6.5           tools_4.4.1           generics_0.1.3
[13] stats4_4.4.1          parallel_4.4.1        fansi_1.0.6
[16] cmdstanr_0.9.0        pkgconfig_2.0.3       Matrix_1.7-0
[19] checkmate_2.3.3       RColorBrewer_1.1-3    S7_0.2.0
[22] distributional_0.5.0  RcppParallel_5.1.11-1 lifecycle_1.0.4
[25] compiler_4.4.1        farver_2.1.2          Brobdingnag_1.2-9
[28] tinytex_0.53          codetools_0.2-20      htmltools_0.5.8.1
[31] yaml_2.3.10           pillar_1.9.0          StanHeaders_2.32.10
[34] bridgesampling_1.1-2  abind_1.4-8           nlme_3.1-164
[37] posterior_1.6.1.9000  rstan_2.32.7          tidyselect_1.2.1
[40] digest_0.6.37         mvtnorm_1.3-3         stringi_1.8.4
[43] reshape2_1.4.4        labeling_0.4.3        fastmap_1.2.0
[46] grid_4.4.1            cli_3.6.5             magrittr_2.0.3
[49] loo_2.8.0             pkgbuild_1.4.8        utf8_1.2.4
[52] withr_3.0.2           scales_1.4.0          backports_1.5.0
[55] estimability_1.5.1    timechange_0.3.0      rmarkdown_2.30
[58] matrixStats_1.5.0     emmeans_2.0.0         gridExtra_2.3
[61] hms_1.1.3             coda_0.19-4.1         evaluate_1.0.1
[64] knitr_1.50            rstantools_2.5.0      rlang_1.1.6
[67] xtable_1.8-4          glue_1.8.0            jsonlite_1.8.9
[70] plyr_1.8.9            R6_2.5.1
```