Bruno Nicenboim, Daniel J. Schad, and Shravan Vasishth

Supplementary material: Introduction to Bayesian Data Analysis for Cognitive Science

This book is dedicated to the cognitive science community.

Contents

A

Regression models with `brms` - Extended

A.1 An efficient function for generating prior predictive distributions in R

As noted in section 3.3, generating prior predictive distributions can be computationally slow if done naively in R with a for-loop: Producing 1000 samples of the prior predictive distribution for our model from section 3.3 results in 361000 predicted values, which takes a few seconds to compute. Although this approach works, it is not optimal for more complex models or larger datasets.

To address this, one could use a more efficient function using the `map2_dfr()` function from the `purrr` package, which yields approximately a 10-fold increase in speed. Although the distributions should be the same with both functions, the specific numbers in the tables won't be, due to the randomness in the process of sampling.

The `purrr` function `map2_dfr()` (which works similarly to the base R function `lapply()` and `Map()`) essentially runs a for-loop, and builds a data frame with the output. It iterates over the values of two vectors (or lists) simultaneously, here, `mu_samples` and `sigma_samples` and, in each iteration, it applies a function to each value of the two vectors, here, `mu` and `sigma`. The output of each function is a data frame (or tibble in this case) with `N_obs` observations which is bound in a larger data frame at the end of the loop. Each of these data frames bound together represents an iteration in the simulation, and we identify the iterations by setting `.id = "iter"`.

Although this method for generating prior predictive distributions is a bit involved, it has an advantage in comparison to the more straightforward use of `predict()` (or `posterior_predict()`, which can also generate prior predictions) together with setting `sample_prior = "only"` in the `brms` model (as we will do in section 3.7.2). Our method of generating prior predictive distributions does not depend on Stan's sampler, which means that no matter the number of iterations in our simulation or how uninformative our priors, there will never be any convergence problems.

```
library(purrr)
# Define the function:
normal_predictive_distribution <- function(mu_samples,
                                           sigma_samples,
                                           N_obs) {
  map2_dfr(mu_samples, sigma_samples, function(mu, sigma) {
    tibble(trialn = seq_len(N_obs),
           t_pred = rnorm(N_obs, mu, sigma))
  }, .id = "iter") %>%
    # .id is always a string and
    # needs to be converted to a number
    mutate(iter = as.numeric(iter))
}
# Test the timing:
tic()
prior_pred <-
  normal_predictive_distribution(mu_samples = mu_samples,
                                 sigma_samples = sigma_samples,
                                 N_obs = N_obs)
toc()
```

```
## 0.364 sec elapsed
```

A.2 Truncated distributions

Any distribution can be truncated. For a continuous distribution, the truncated version of the original distribution will have non-zero probability density values for a continuous subset of the original coverage. To make this more concrete, in the example of section 3.5, the normal distribution has coverage for values between minus infinity to plus infinity, and our truncated version $Normal_+$ has coverage between zero and plus infinity: all negative values have a density of zero. Let's see how we can generalize this to be able to understand any truncation of any continuous distribution. (For the discrete case, we can simply replace the integral with a sum, and replace PDF with PMF).

From the axiomatic definitions of probability, we know that the area below a PDF, $f(x)$, must be equal to one (section 1.1). More formally, this means that the integral of f evaluated as $f(-\infty < X < \infty)$ should be equal to one:

$$\int_{-\infty}^{\infty} f(x)dx = 1 \tag{A.1}$$

But if the distribution is truncated, f is going to be evaluated in some subset of its possible values, $f(a < X < b)$; in the specific case of $Normal_+$, for example, $a = 0$, and $b = \infty$. In the general case, this means that the integral of the PDF evaluated for $a < X < b$ will be lower than one, unless $a = -\infty$ and $b = +\infty$.

$$\int_{a}^{b} f(x)dx < 1 \tag{A.2}$$

We want to ensure that we build a new PDF for the truncated distribution so that even though it has less coverage than the non-truncated version, it still integrates to one. To achieve this, we divide the "unnormalized" PDF by the total area of $f(a < X < b)$ (recall the discussion surrounding Equation (1.17)):

$$f_{[a,b]}(x) = \frac{f(x)}{\int_a^b f(x)dx} \tag{A.3}$$

The denominator of the previous equation is the difference between the CDF evaluated at $X = b$ and the CDF evaluated at $X = a$; this can be written as $F(b) - F(a)$:

$$f_{[a,b]}(x) = \frac{f(x)}{F(b) - F(a)} \tag{A.4}$$

For the specific case where $f(x)$ is $Normal(x|0, \sigma)$ and we want the PDF of $Normal_+(x|0, \sigma)$, the bounds will be $a = 0$ and $b = \infty$.

$$Normal_+(x|0, \sigma) = \frac{Normal(x|0, \sigma)}{1/2} \tag{A.5}$$

Because $F(X = b = \infty) = 1$ and $F(X = a = 0) = 1/2$.

You can verify this in R (this is valid for any value of `sd`).

```
dnorm(1, mean = 0) * 2 == dtnorm(1, mean = 0, a = 0)
```

```
## [1] TRUE
```

Unless the truncation of the normal distribution is symmetrical, the mean μ of the truncated normal does not coincide with the mean of the parent (untruncated) normal distribution; call this mean of the parent distribution $\hat{\mu}$. For any type of truncation, the standard deviation of the truncated distribution σ

does not coincide with the standard deviation of the parent distribution; call this latter standard deviation $\hat{\sigma}$. Confusingly enough, the family of truncated functions `*tnorm()` keeps the names of the arguments of the family of functions `*norm()`: `mean` and `sd`. So, when defining a truncated normal distribution like `dtnorm(mean = 300, sd = 100, a = 0, b = Inf)`, the `mean` and `sd` refer to the mean $\hat{\mu}$ and standard deviation $\hat{\sigma}$ of the untruncated parent distribution.

Sometimes one needs to model observed data as coming from a truncated normal distribution. An example would be a vector of observed standard deviations; perhaps one wants to use these estimates to work out a truncated normal prior. In order to derive such an empirically motivated prior, we have to work out what mean and standard deviation we need to use in a truncated normal distribution. We could compute the mean and standard deviation from the observed vector of standard deviations, and then use the procedure shown below to work out the mean and standard deviation that we would need to put into the truncated normal distribution. This approach is used in online chapter E, section E.1.4 for working out a prior based on standard deviation estimates from existing data.

The mean and standard deviation of the parent distribution of a truncated normal ($\hat{\mu}$ and $\hat{\sigma}$) with boundaries a and b, given the mean μ and standard deviation σ of the truncated normal, are computed as follows (Johnson, Kotz, and Balakrishnan 1995). $\phi(X)$ is the PDF of the standard normal (i.e., $Normal(\mu = 0, \sigma = 1)$) evaluated at X, and $\Phi(X)$ is the CDF of the standard normal evaluated at X.

First, define two terms α and β for convenience:

$$\alpha = (a - \hat{\mu})/\hat{\sigma} \qquad \beta = (b - \hat{\mu})/\hat{\sigma} \tag{A.6}$$

Then, the mean μ of the truncated distribution can be computed as follows based on the parameters of the parent distribution:

$$\mu = \hat{\mu} - \hat{\sigma}\frac{\phi(\beta) - \phi(\alpha)}{\Phi(\beta) - \Phi(\alpha)} \tag{A.7}$$

The variance σ^2 of the truncated distribution is:

$$\sigma^2 = \hat{\sigma}^2 \times \left(1 - \frac{\beta\phi(\alpha) - \alpha\phi(\beta)}{\Phi(\beta) - \Phi(\alpha)} - \left(\frac{\phi(\alpha) - \phi(\beta)}{\Phi(\beta) - \Phi(\alpha)}\right)^2\right) \tag{A.8}$$

Equations (A.7) and (A.8) have two variables, so if one is given the values for the truncated distribution μ and σ, one can solve (using algebra) for the mean and standard deviation of the untruncated distribution, $\hat{\mu}$ and $\hat{\sigma}$.

For example, suppose that $a = 0$ and $b = 500$, and that the mean and standard deviation of the untruncated parent distribution is $\hat{\mu} = 300$ and $\hat{\sigma} = 200$. We can simulate such a situation and estimate the mean and standard deviation of the truncated distribution:

```
x <- rtnorm(10000000, mean = 300, sd = 200, a = 0, b = 500)
## the mean and sd of the truncated distributions
## using simulation:
mean(x)
```

```
## [1] 271
```

```
sd(x)
```

```
## [1] 129
```

These simulated values are identical to the values computed using equations (A.7) and (A.8):

```
a <- 0
b <- 500
bar_x <- 300
bar_sigma <- 200
alpha <- (a - bar_x) / bar_sigma
beta <- (b - bar_x) / bar_sigma
term1 <- ((dnorm(beta) - dnorm(alpha)) /
            (pnorm(beta) - pnorm(alpha)))
term2 <- ((beta * dnorm(beta) - alpha * dnorm(alpha)) /
            (pnorm(beta) - pnorm(alpha)))
## the mean and sd of the truncated distribution
## computed analytically:
(mu <- bar_x - bar_sigma * term1)
```

```
## [1] 271
```

```
(sigma <- sqrt(bar_sigma^2 * (1 - term2 - term1^2)))
```

```
## [1] 129
```

The equations for the mean and variance of the truncated distribution (μ and σ) can also be used to work out the mean and variance of the parent

untruncated distribution ($\hat{\mu}$ and $\hat{\sigma}$), if one has estimates for μ and σ (from data).

Suppose that we have observed data with mean $\mu = 271$ and $\sigma = 129$. We want to assume that the data are coming from a truncated normal which has lower bound 0 and upper bound 500. What are the mean and standard deviation of the parent distribution, $\hat{\mu}$ and $\hat{\sigma}$?

To answer this question, first rewrite the equations as follows:

$$\mu - \hat{\mu} + \hat{\sigma}\frac{\phi(\beta) - \phi(\alpha)}{\Phi(\beta) - \Phi(\alpha)} = 0 \tag{A.9}$$

$$\sigma^2 - \hat{\sigma}^2 \times \left(1 - \frac{\beta\phi(\alpha) - \alpha\phi(\beta)}{\Phi(\beta) - \Phi(\alpha)} - \left(\frac{\phi(\alpha) - \phi(\beta)}{\Phi(\beta) - \Phi(\alpha)}\right)^2\right) = 0 \tag{A.10}$$

Next, solve for $\hat{\mu}$ and $\hat{\sigma}$ given the observed mean and the standard deviation of the truncated distribution, and that one knows the boundaries (a, and b).

Define the system of equations according to the specifications of `multiroot()` from the package `rootSolve`: `x` for the unknowns ($\hat{\mu}$ and $\hat{\sigma}$), and `parms` for the known parameters: a, b, and the mean and standard deviation of the truncated normal.

```
eq_system <- function(x, parms) {
  mu_hat <- x[1]
  sigma_hat <- x[2]
  alpha <- (parms["a"] - mu_hat) / sigma_hat
  beta <- (parms["b"] - mu_hat) / sigma_hat
  c(F1 = parms["mu"] - mu_hat + sigma_hat *
    (dnorm(beta) - dnorm(alpha)) / (pnorm(beta) - pnorm(alpha)),
    F2 = parms["sigma"] - sigma_hat *
     sqrt((1 - ((beta) * dnorm(beta) - (alpha) * dnorm(alpha)) /
   (pnorm(beta) - pnorm(alpha)) - ((dnorm(beta) - dnorm(alpha)) /
             (pnorm(beta) - pnorm(alpha)))^2)))
}
```

Solving the two equations using `multiroot()` from the package `rootSolve` gives us the mean and standard deviation $\hat{\mu}$ and $\hat{\sigma}$ of the parent normal distribution. (Notice that `x` is a required parameter of the previous function so that it works with `multiroot()`, however, outside of the function the variable `x` is a vector containing the samples of the truncated normal distribution generated with `rtnorm()`).

```
soln <- multiroot(f = eq_system,
                  start = c(1, 1),
                  parms = c(a = 0,
                            b = 500,
                            mu = mean(x),
                            sigma = sd(x)))
soln$root
```

```
## [1] 300 200
```

The function `compute_meansd_parent()` encapsulates the previous procedure and it is provided in the `bcogsci` package.

A.3 Intercepts in `brms`

When we set up a prior for the intercept in `brms`, we actually set a prior for an intercept assuming that all the predictors are centered. This means that when predictors are not centered (and only then), there is a mismatch between the interpretation of the intercept as returned in the output of `brms` and the interpretation of the intercept with respect to its prior specification. In this case, only the intercept in the output corresponds to the formula in the `brms` call, that is, the intercept in the output corresponds to the non-centered model. However, as we show below, when the intercept is much larger than the effects that we are considering in the formula (what we generally call β), this discrepancy hardly matters.

The reason for this mismatch when our predictors are uncentered is that `brms` increases sampling efficiency by automatically centering all the predictors internally (that is the population-level design matrix X is internally centered around its column means when `brms` fits a model). This did not matter in our previous examples because we centered our predictor (or we had no predictor), but it might matter if we want to have uncentered predictors. In the design we are discussing, a non-centered predictor of load will mean that the intercept, α, has a straightforward interpretation: the α is the mean pupil size when there is no attention load. This is in contrast with the centered version presented before, where the intercept α represents the pupil size for the average load of `2.44` (`c_load` is equal to 0). The difference between the non-centered model (below) and the centered version presented before is depicted in Figure A.1.

Suppose that we are quite sure that the prior values for the no load condition

(i.e., load is non-centered) fall between 400 and 1200 ms. In that case, the following prior could be set for α: $Normal(800, 200)$. In this case, the model becomes:

```
prior_nc <-
  c(prior(normal(800, 200), class = b, coef = Intercept),
    prior(normal(0, 1000), class = sigma),
    prior(normal(0, 100), class = b, coef = load))

fit_pupil_non_centered <- brm(p_size ~ 0 + Intercept + load,
                              data = df_pupil,
                              family = gaussian(),
                              prior = prior_nc)
```

When the predictor is non-centered as shown above, the regular centered intercept is removed by adding `0` to the formula, and by replacing the intercept with the "actual" intercept we want to set priors on with `Intercept`. The word `Intercept` is a reserved word; we cannot name any predictor with this name. This new parameter is also of class `b`, so its prior needs to be defined accordingly. Once we use `0 + Intercept + ..`, the intercept is not calculated with predictors that are automatically centered any more.

The output below shows that, as expected, although the posterior for the intercept has changed noticeably, the posterior for the effect of load remains virtually unchanged.

```
posterior_summary(fit_pupil_non_centered,
                  variable = c("b_Intercept", "b_load"))
```

```
##             Estimate Est.Error   Q2.5 Q97.5
## b_Intercept    622.9      35.5 553.68 694.5
## b_load          32.8      12.1   9.02  56.5
```

Notice the following potential pitfall. A model like the one below will fit a non-centered load predictor, but will assign a prior of $Normal(800, 200)$ to the intercept of a model that assumes a centered predictor, $\alpha_{centered}$, and not the current intercept, α.

```
prior_nc <-
  c(prior(normal(800, 200), class = Intercept),
    prior(normal(0, 1000), class = sigma),
    prior(normal(0, 100), class = b, coef = load))
```

```
fit_pupil_wrong <- brm(p_size ~ 1 + load,
                      data = df_pupil,
                      family = gaussian(),
                      prior = prior_nc)
```

What does it mean to set a prior to $\alpha_{centered}$ in a model that doesn't include $\alpha_{centered}$?

The fitted (expected) values of the non-centered model and the centered one are identical, that is, the values of the response distribution without the residual error are identical for both models:

$$\alpha + load_n \cdot \beta = \alpha_{centered} + (load_n - mean(load)) \cdot \beta \tag{A.11}$$

The left side of Equation (A.11) refers to the expected values based on our current non-centered model, and the right side refers to the expected values based on the centered model. We can re-arrange terms to understand what the effect is of a prior on $\alpha_{centered}$ in our model that doesn't include $\alpha_{centered}$.

$$\begin{aligned} \alpha + load_n \cdot \beta &= \alpha_{centered} + load_n \cdot \beta - mean(load) \cdot \beta \\ \alpha &= \alpha_{centered} - mean(load) \cdot \beta \\ \alpha + mean(load) \cdot \beta &= \alpha_{centered} \end{aligned} \tag{A.12}$$

That means that in the non-centered model, we are actually setting our prior to $\alpha + mean(load) \cdot \beta$. When β is very small (or the means of our predictors are very small because they might be "almost" centered), and the prior for α is very wide, we might hardly notice the difference between setting a prior to $\alpha_{centered}$ or to our actual α in a non-centered model (especially if the likelihood dominates anyway). But it is important to pay attention to what the parameters represent that we are setting priors on.

To sum up, `brms` automatically centers all predictors for posterior estimation, and the prior of the intercept is applied to the centered version of the model during model fitting. However, when the predictors specified in the formula are not centered, then `brms` uses the equations shown before to return in the output the posterior of the intercept for the non-centered predictors.[1]

In our example analyses with `brms` in this book, we will always center our predictors.

[1] These transformations are visible when checking the generated Stan code using `make_stancode()`.

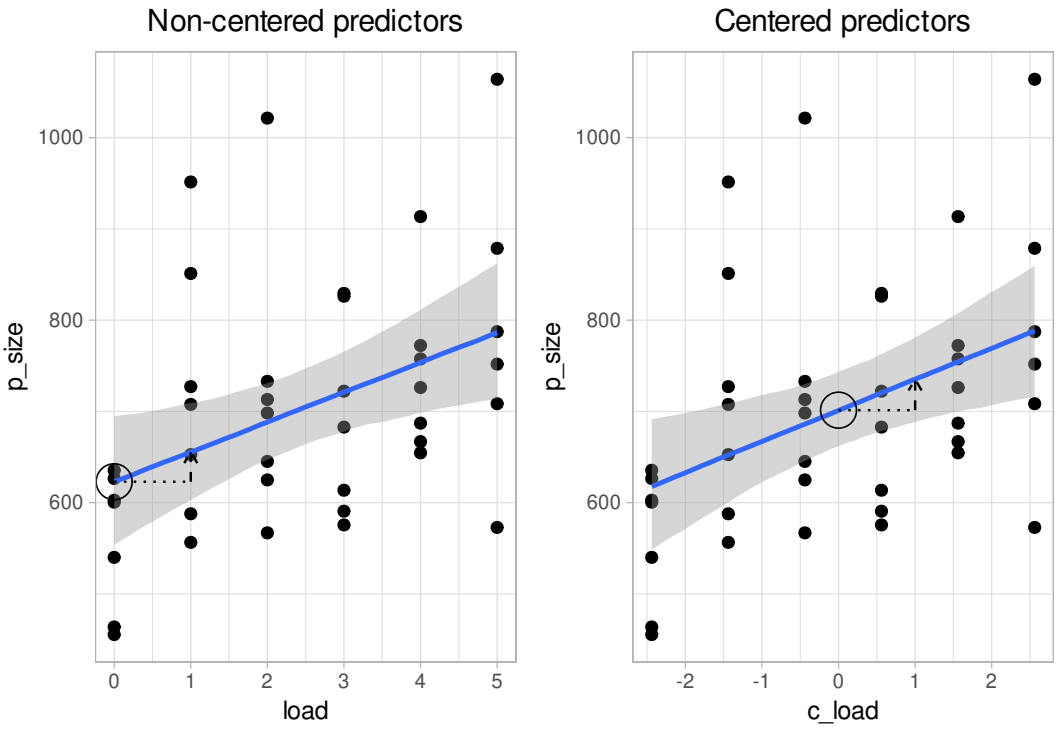

FIGURE A.1: Regression lines for the non-centered and centered linear regressions. The intercept (α) represented by a circle is positioned differently depending on the centering, whereas the slope (β) represented by a vertical dashed line has the same magnitude in both models.

A.4 Understanding the log-normal likelihood

It is important to understand what we are assuming with a log-normal likelihood. Formally, if a random variable Z is normally distributed with mean μ and variance σ^2, then the transformed random variable $Y = \exp(Z)$ is log-normally distributed and has density:

$$LogNormal(y|\mu,\sigma) = f(z) = \frac{1}{\sqrt{2\pi\sigma^2}y}\exp\left(-\frac{(\log(y)-\mu)^2}{2\sigma^2}\right) \tag{A.13}$$

As explained in section 3.7.1, the model from Equation (4.5) is equivalent to the following:

$$\log(t_n) \sim Normal(\alpha + c\_trial_n \cdot \beta, \sigma) \tag{A.14}$$

The family of normal distributions is closed under linear transformations: that

is, if X is normally distributed with mean μ and standard deviation σ, then (for any real numbers a and b), $aX + b$ is also normally distributed, with mean $a\mu + b$ (and standard deviation $\sqrt{a^2\sigma^2} = |a|\sigma$).

This means that, assuming $Z \sim Normal(\alpha, \sigma)$, Equation (A.14) can be re-written as follows:

$$\log(rt_n) = Z + c\_trial_n \cdot \beta \tag{A.15}$$

Exponentiate both sides, and use the property of exponents that $\exp(x + y)$ is equal to $\exp(x) \cdot \exp(y)$; set $Y = \exp(Z)$.

$$\begin{aligned} rt_n &= \exp\left(Z + c\_trial_n \cdot \beta\right) \\ rt_n &= \exp(Z) \cdot \exp\left(c\_trial_n \cdot \beta\right) \\ rt_n &= Y \cdot \exp\left(c\_trial_n \cdot \beta\right) \end{aligned} \tag{A.16}$$

The last equation has two terms being multiplied, the first one, Y, is telling us that we are assuming that finger tapping times are log-normally distributed with a median of $\exp(\alpha)$, the second term, $\exp(c\_trial_n \cdot \beta)$ is telling us that the effect of trial number is multiplicative and grows or decays exponentially with the trial number. This has two important consequences:

1. Different values of the intercept, α, given the same β, will affect the difference in finger tapping or response times for two adjacent trials (compare this with what happens with an additive model, such as when a normal likelihood is used); see Figure A.2. This is because, unlike in the additive case, the intercept doesn't cancel out:

    - Additive case:

$$\begin{aligned} &(\alpha + trial_n \cdot \beta) - (\alpha + trial_{n-1} \cdot \beta) = \\ &= \alpha - \alpha + (trial_n - trial_{n-1}) \cdot \beta \\ &= (trial_n - trial_{n-1}) \cdot \beta \end{aligned} \tag{A.17}$$

    - Multiplicative case:

$$\begin{aligned} &\exp(\alpha) \cdot \exp(trial_n \cdot \beta) - \exp(\alpha) \cdot \exp(trial_{n-1} \cdot \beta) = \\ &= \exp(\alpha)\big(\exp(trial_n \cdot \beta) - \exp(trial_{n-1} \cdot \beta)\big) \\ &\neq \big(\exp(trial_n) - \exp(trial_{n-1})\big) \cdot \exp(\beta) \end{aligned} \tag{A.18}$$

2. As the trial number increases, the same value of β will have a very different impact on the original scale of the dependent variable: Any

(fixed) negative value for β will lead to exponential decay and any (fixed) positive value will lead to exponential growth; see Figure A.3.

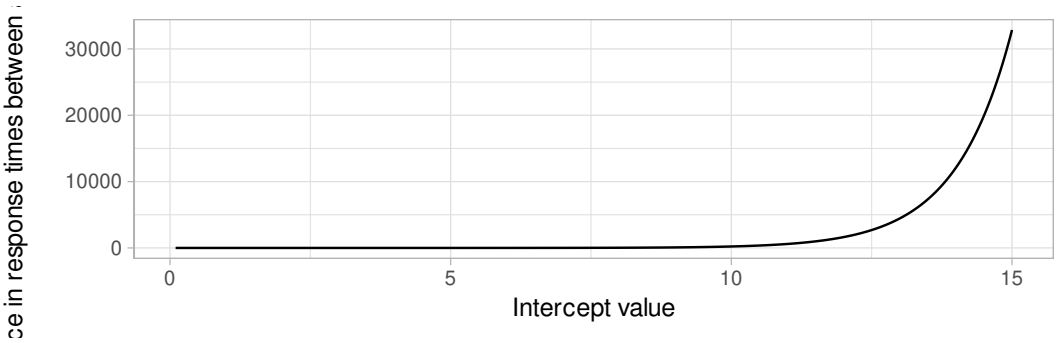

FIGURE A.2: The fitted values of the difference in response time between two adjacent trials, when $\beta = 0.01$ and α lies between 0.1 and 15. The graph shows how changes in the intercept lead to changes in the difference in response times between trials, even if β is fixed.

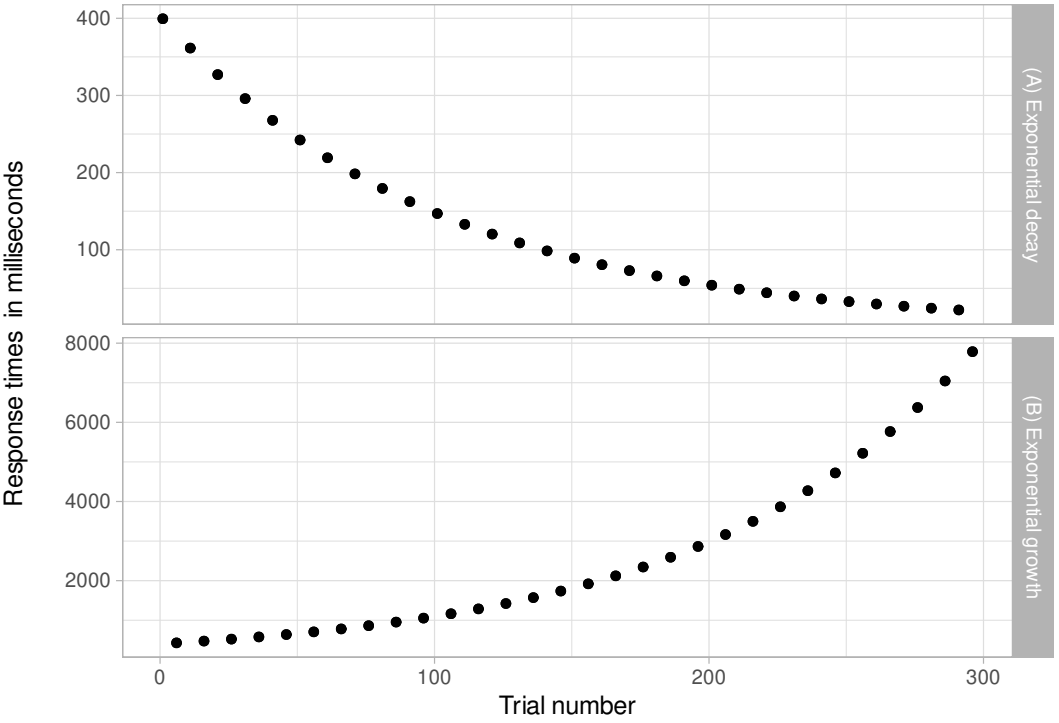

FIGURE A.3: The fitted values of the dependent variable (response times in ms) as a function of trial number, when (A) $\beta = -0.01$, exponential decay, and when (B) $\beta = 0.01$, exponential growth.

Does exponential growth or decay make sense in this particular example? We

need to consider that if they do make sense, they will be an approximation valid for a specific range of values, at some point we will expect a ceiling or a floor effect: response times cannot truly be 0 milliseconds, or take several minutes. However, in our specific model, exponential growth or decay *by trial* is probably a bad approximation: We will predict that our subject will take extremely long (if $\beta > 0$) or extremely short (if $\beta < 0$) time in pressing the space bar in a relatively low number of trials. This doesn't mean that the likelihood is wrong by itself, but it does mean that at least we need to put a cap on the growth or decay of our experimental manipulation. We can do this if the exponential growth or decay is a function of, for example, log-transformed trial numbers:

$$t_n \sim LogNormal(\alpha + c\_\log\_trial_n \cdot \beta, \sigma) \tag{A.19}$$

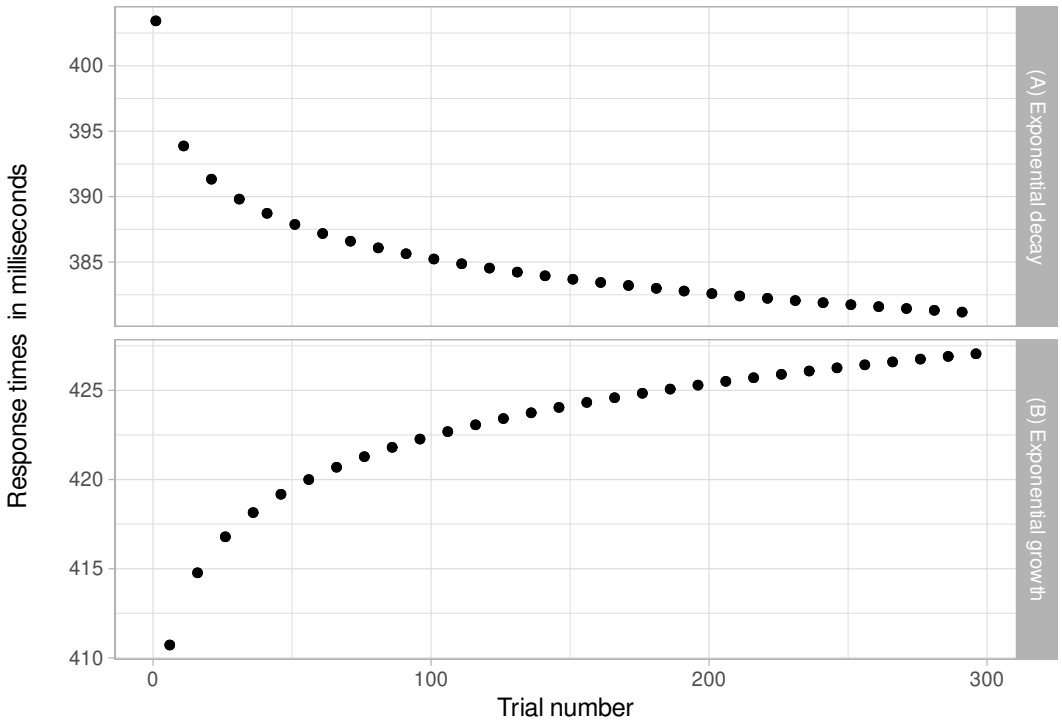

FIGURE A.4: Fitted value of the dependent variable (times in ms) as function of the natural logarithm of the trial number, when (A) $\beta = -0.01$, exponential decay, and when (B) $\beta = .01$, exponential growth.

A.4.1 Log-normal distributions everywhere

The normal distribution is most often assumed to describe the random variation that occurs in the data from many scientific disciplines. However, most

measurements actually show skewed distributions. Limpert, Stahel, and Abbt (2001) discuss the log-normal distribution in scientific disciplines and how diverse type of data, from lengths of latent periods of infectious diseases to distribution of mineral resources in the Earth's crust, including even body height–the quintessential example of a normal distribution–closely fit the log-normal distribution.

Limpert, Stahel, and Abbt (2001) point out that because a random variable that results from multiplying many independent variables has an approximate log-normal distribution, the most basic indicator of the importance of the log-normal distribution may be very general: Chemistry and physics are fundamental in life, and the prevailing operation in the laws of these disciplines is multiplication rather than addition.

Furthermore, at many physiological and anatomical levels in the brain, the distribution of numerous parameters is in fact strongly skewed with a heavy tail, suggesting that skewed (typically log-normal) distributions are fundamental to structural and functional brain organization. This might be explained given that the majority of interactions in highly interconnected systems, especially in biological systems, are multiplicative and synergistic rather than additive (Buzsáki and Mizuseki 2014).

Does the log-normal distribution make sense for response times? It has been long noticed that the log-normal distribution often provides a good fit to response times distributions (Brée 1975; Ulrich and Miller 1994). One advantage of assuming log-normally distributed response times (but, in fact, this is true for many skewed distributions) is that it entails that the standard deviation of the response time distribution will increase with the mean, as has been observed in empirical distributions of response times (Wagenmakers, Grasman, and Molenaar 2005). Interestingly, it turns out that log-normal response times are also easily generated by certain process models. Ulrich and Miller (1993) show, for example, that models in which response times are determined by a series of processes cascading activation from an input level to an output level (usually passing through a number of intervening processing levels along the way) can generate log-normally distributed response times.

A.5 Prior predictive checks in R

The following function is an edited version of the earlier `normal_predictive_distribution` from the online section A.1, which was used

in section 3.3; it has been edited to make it compatible with logistic regression and dependent on set size.

As we did before, our custom function uses the `purrr` function `map2_dfr()`, which runs an efficient for-loop, iterating over two vectors (here `alpha_samples` and `beta_samples`), and builds a data frame with the output.

```
logistic_model_pred <- function(alpha_samples,
                                beta_samples,
                                set_size,
                                N_obs) {
  map2_dfr(alpha_samples, beta_samples,
           function(alpha, beta) {
             tibble(set_size = set_size,
                    # center size:
                    c_set_size = set_size - mean(set_size),
                    # change the likelihood:
                    # Notice the use of a link function
                    # for alpha and beta
                    theta = plogis(alpha + c_set_size * beta),
                    correct_pred = rbern(N_obs, prob = theta))
           },
           .id = "iter") %>%
    # .id is always a string and needs
    # to be converted to a number
    mutate(iter = as.numeric(iter))
}
```

Let's assume 800 observations with 200 observation for each set size:

```
N_obs <- 800
set_size <- rep(c(2, 4, 6, 8), 200)
```

Now, iterate over plausible standard deviations of β with the `purrr` function `map_dfr()`, which iterates over one vector (here `sds_beta`), and also builds a data frame with the output.

```
alpha_samples <- rnorm(1000, 0, 1.5)
sds_beta <- c(1, 0.5, 0.1, 0.01, 0.001)
prior_pred <- map_dfr(sds_beta, function(sd) {
  beta_samples <- rnorm(1000, 0, sd)
```

```
  logistic_model_pred(alpha_samples = alpha_samples,
                      beta_samples = beta_samples,
                      set_size = set_size,
                      N_obs = N_obs) %>%
    mutate(prior_beta_sd = sd)
})
```

Calculate the accuracy for each one of the priors we want to examine, for each iteration, and for each set size.

```
mean_accuracy <-
  prior_pred %>%
  group_by(prior_beta_sd, iter, set_size) %>%
  summarize(accuracy = mean(correct_pred)) %>%
  mutate(prior = paste0("Normal(0, ", prior_beta_sd, ")"))
```

The plot of the accuracy is shown in Figure 4.13 of the book, and repeated here in Figure A.5.

```
mean_accuracy %>%
  ggplot(aes(accuracy)) +
  geom_histogram() +
  facet_grid(set_size ~ prior) +
  scale_x_continuous(breaks = c(0, .5, 1))
```

It's sometimes more useful to look at the predicted differences in accuracy between set sizes. We calculate them as follows, and plot them in Figure 4.14 of the book, repeated here in Figure A.6.

```
diff_accuracy <- mean_accuracy %>%
  arrange(set_size) %>%
  group_by(iter, prior_beta_sd) %>%
  mutate(diff_accuracy = accuracy - lag(accuracy)) %>%
  mutate(diffsize = paste(set_size, "-", lag(set_size))) %>%
  filter(set_size > 2)
```

```
diff_accuracy %>%
  ggplot(aes(diff_accuracy)) +
  geom_histogram() +
```

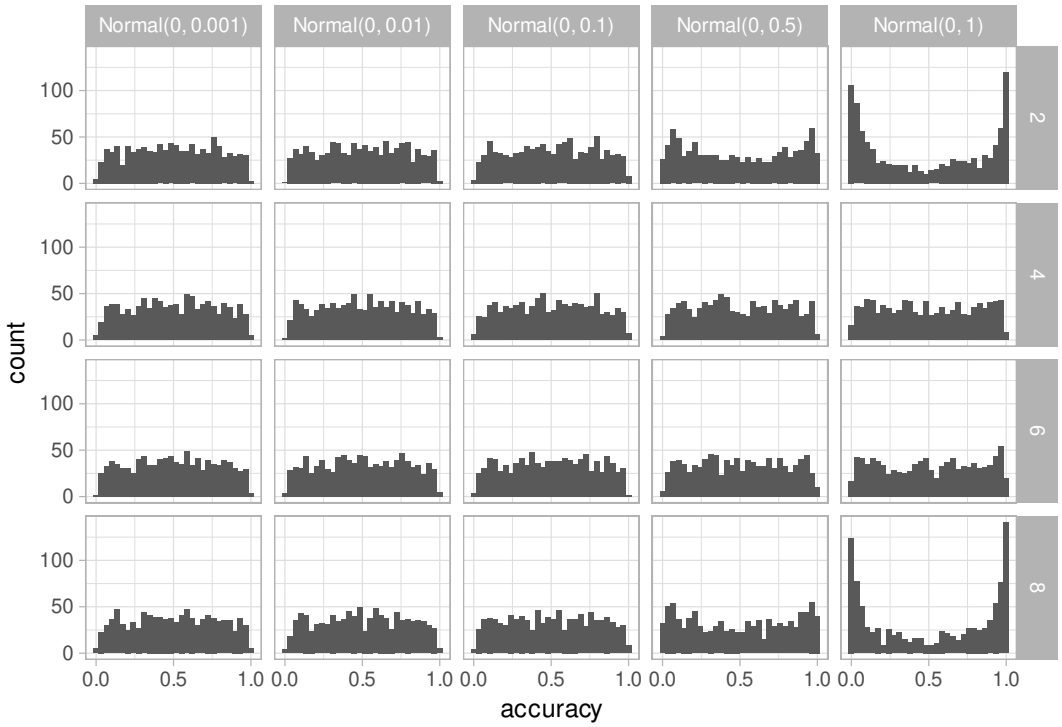

FIGURE A.5: The prior predictive distributions of mean accuracy of the model defined in section 4.3, for different set sizes and different priors for β.

```
  facet_grid(diffsize ~ prior) +
  scale_x_continuous(breaks = c(-.5, 0, .5))
```

A.6 Finitely exchangeable random variables

Formally, we say that the random variables $Y_1, \ldots, Y_N$ are finitely exchangeable if, for any set of particular outcomes of an experiment $y_1, \ldots, y_N$, the probability $p(y_1, \ldots, y_N)$ that we assign to these outcomes is unaffected by permuting the labels given to the variables. In other words, for any permutation $\pi(n)$, where $n = 1, \ldots, N$ (π is a function that takes as input the positive integer n and returns another positive integer; e.g., the function takes a subject indexed as 1, and returns index 3), we can reasonably assume that $p(y_1, \ldots, y_N) = p(y_{\pi(1)}, \ldots, y_{\pi(N)})$. A simple example is a coin tossed twice. Suppose the first coin toss is $Y_1 = 1$, a heads, and the second coin toss is $Y_2 = 0$, a tails. If we are willing to assume that the probability of getting one heads is unaffected by whether it appears in the first or the second toss, i.e.,

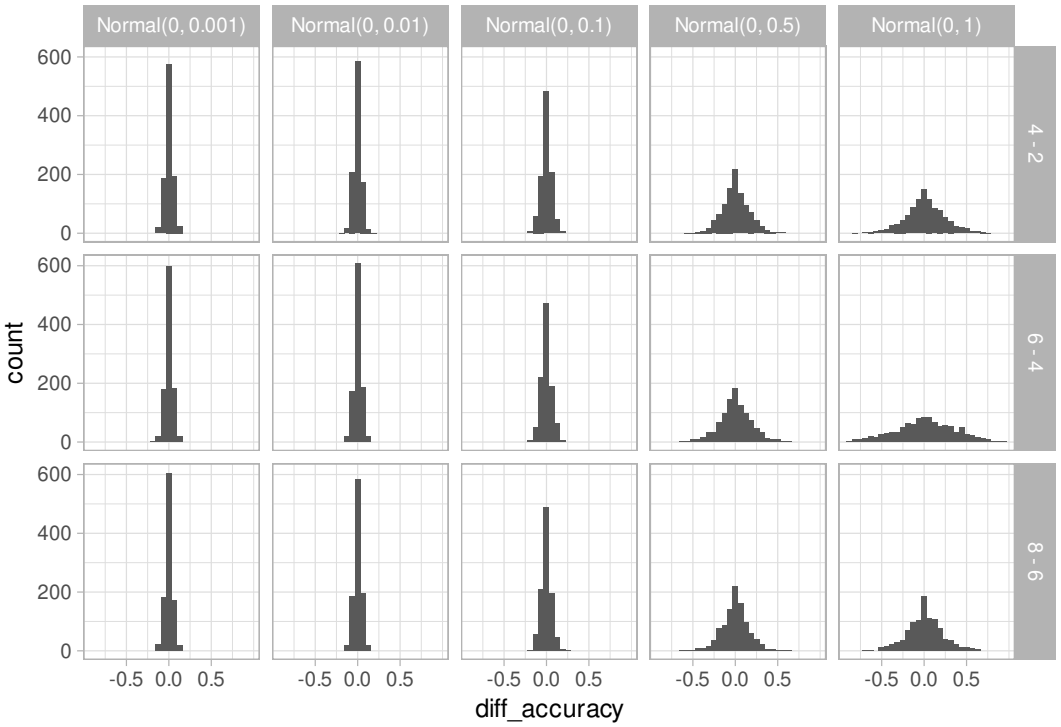

FIGURE A.6: The prior predictive distributions of differences in mean accuracy between set sizes of the model defined in section 4.3 for different priors for β.

$p(Y_1 = 1, Y_2 = 0) = p(Y_1 = 0, Y_2 = 1)$, then we assume that the indices are exchangeable.

Some important connections and differences between exchangeability and the frequentist concept of independent and identically distributed (iid):

- **If the data are exchangeable, they are not necessarily iid**. For example, suppose you have a box with one black ball and two red balls in it. Your task is to repeatedly draw (without replacement) a ball at random. Suppose that in your first draw, you draw one ball and get the black ball. The probability of getting a black ball in the next two draws is now 0. However, if in your first draw you had retrieved a red ball, then there is a non-zero probability of drawing a black ball in the next two draws. The outcome in the first draw affects the probability of subsequent draws–they are not independent. But the sequence of random variables is exchangeable. To see this, consider the following: If a red ball is drawn, count it as a 0, and if a black ball is drawn, then count it as 1. Then, the three possible outcomes and the probabilities are
  - $1, 0, 0;\ P(X_1 = 1, X_2 = 0, X_3 = 0) = \frac{1}{3} \times 1 \times 1 = \frac{1}{3}$

  - $0, 1, 0$ $P(X_1 = 0, X_2 = 1, X_3 = 0) = \frac{2}{3} \times \frac{1}{2} \times 1 = \frac{1}{3}$
  - $0, 0, 1$ $P(X_1 = 0, X_2 = 0, X_3 = 1) = \frac{2}{3} \times \frac{1}{2} \times 1 = \frac{1}{3}$

  The random variables X_1, X_2, X_3 can be permuted and the joint probability distribution (technically, the PMF) is the same in each case.

- **If the data are exchangeable, then they are identically distributed**. For example, in the box containing one black ball and two red balls, suppose we count the draw of a black ball as a 1, and the draw of a red ball as a 0. Then the probability $P(X_1 = 1) = \frac{1}{3}$ and $P(X_1 = 0) = \frac{2}{3}$; this is also true for X_2 and X_3. That is, these random variables are identically distributed.

- **If the data are iid in the standard frequentist sense, then they are exchangeable**. For example, suppose you have $i = 1, \ldots, n$ instances of a random variable X whose PDF is $f(x)$. Suppose also that X_i are iid. The joint PDF (this can be discrete or continuous, i.e., a PMF or PDF) is

$$f_{X_1,\ldots,X_n}(x_1, \ldots, x_n) = f(x_1) \cdot \cdots \cdot f(x_n) \tag{A.20}$$

  Because the terms on the right-hand side can be permuted, the labels can be permuted on any of the x_i. This means that $X_1, \ldots, X_n$ are exchangeable.

A.7 The Matrix Formulation of Hierarchical Models (the Laird-Ware form)

In the book, we generally write linear models as follows; where n refers to the row id in the data frame.

$$y_n \sim Normal(\alpha + \beta \cdot x_n) \tag{A.21}$$

This simple linear model can be re-written as follows:

$$y_n = \alpha + \beta \cdot x_n + \varepsilon_n \tag{A.22}$$

where $\varepsilon_n \sim Normal(0, \sigma)$.

The model does not change if α is multiplied by 1:

$$y_n = \alpha \cdot 1 + \beta \cdot x_n + \varepsilon_n \tag{A.23}$$

The above is actually n linear equations, and can be written compactly in matrix form:

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix} \tag{A.24}$$

Consider this matrix in the above equation:

$$\begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix} \tag{A.25}$$

This matrix is called the model matrix or the design matrix; we will encounter it again in the contrast coding chapters, where it plays a crucial role. If we write the dependent variable y as a $n \times 1$ vector, the above matrix as the matrix X (which has dimensions $n \times 2$), the intercept and slope parameters as a 2×1 matrix ζ, and the residual errors as an $n \times 1$ matrix, we can write the linear model very compactly:

$$y = X\zeta + \varepsilon \tag{A.26}$$

The above matrix formulation of the linear model extends to the hierarchical model very straightforwardly. For example, consider the by-subjects and by-items correlated varying intercept varying slopes model M_{sih} that we saw in section 5.2.5. This model has the following likelihood:

$$\begin{aligned} signal_n \sim Normal(&\alpha + u_{subj[n],1} + w_{item[n],1} \\ &+ c\_cloze_n \cdot (\beta + u_{subj[n],2} + w_{item[n],2}), \sigma) \end{aligned} \tag{A.27}$$

The terms in the location parameter in the normal likelihood can be re-written in matrix form, just like the linear model above. To see this, consider the fact that the location term

$$\alpha + u_{subj[n],1} + w_{item[n],1} + c\_cloze_n \cdot (\beta + u_{subj[n],2} + w_{item[n],2}) \tag{A.28}$$

can be re-written as

$$\begin{aligned} &\alpha \cdot 1 + u_{subj[n],1} \cdot 1 + w_{item[n],1} \cdot 1 + \\ &\qquad \beta \cdot c\_cloze_n + u_{subj[n],2} \cdot c\_cloze_n + w_{item[n],2} \cdot c\_cloze_n \end{aligned} \tag{A.29}$$

The above equation can in turn be written in matrix form as follows. The symbol $\odot$ is the Hadamard product: this is cell-wise multiplication rather than matrix multiplication.[2]

$$\begin{pmatrix} 1 & c\_cloze_1 \\ 1 & c\_cloze_2 \\ \vdots & \vdots \\ 1 & c\_cloze_n \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} + \\ \begin{pmatrix} 1 & c\_cloze_1 \\ 1 & c\_cloze_2 \\ \vdots & \vdots \\ 1 & c\_cloze_n \end{pmatrix} \odot \begin{pmatrix} u_{subj[1],1} & u_{subj[1],2} \\ u_{subj[2],1} & u_{subj[2],2} \\ \vdots & \vdots \\ u_{subj[n],1} & u_{subj[n],2} \end{pmatrix} + \\ \begin{pmatrix} 1 & c\_cloze_1 \\ 1 & c\_cloze_2 \\ \vdots & \vdots \\ 1 & c\_cloze_n \end{pmatrix} \odot \begin{pmatrix} w_{item[1],1} & w_{item[1],2} \\ w_{item[2],1} & w_{item[2],2} \\ \vdots & \vdots \\ w_{item[n],1} & w_{item[n],2} \end{pmatrix} \tag{A.30}$$

In this hierarchical model, there are three model matrices:

- the model matrix associated with the intercept α and the slope β; below, we call this the matrix X.
- the model matrix associated with the by-subject varying intercepts and slopes; call this the matrix Z_u.
- the model matrix associated with the by-item varying intercepts and slopes; call this the matrix Z_w.

The model can now be written very compactly in matrix form by writing these three matrices as follows:

[2] This means that if we have two matrices $A_{i,j}$ and $B_{i,j}$, the Hadamard product produces a matrix $C_{i,j}$ that is the result of multiplying each cell $A_{i,j}$ with $B_{i,j}$, for all row and column ids i, j respectively.

$$
\begin{aligned}
X &= \begin{pmatrix} 1 & c\_cloze_1 \\ 1 & c\_cloze_2 \\ \vdots & \vdots \\ 1 & c\_cloze_n \end{pmatrix} \\
Z_u &= \begin{pmatrix} 1 & c\_cloze_1 \\ 1 & c\_cloze_2 \\ \vdots & \vdots \\ 1 & c\_cloze_n \end{pmatrix} \\
Z_w &= \begin{pmatrix} 1 & c\_cloze_1 \\ 1 & c\_cloze_2 \\ \vdots & \vdots \\ 1 & c\_cloze_n \end{pmatrix}
\end{aligned} \tag{A.31}
$$

The location part of the model M_{sih} can now be written very compactly:

$$X\zeta + Z_u \odot z_u + Z_w \odot z_w \tag{A.32}$$

Here, ζ is a 2×1 matrix containing the intercept α and the slope β, and z_u and z_w are the intercept and slope adjustments by subject and by item:

$$
\begin{aligned}
z_u &= \begin{pmatrix} u_{subj[1],1} & u_{subj[1],2} \\ u_{subj[2],1} & u_{subj[2],2} \\ \vdots & \vdots \\ u_{subj[n],1} & u_{subj[n],2} \end{pmatrix} \\
z_w &= \begin{pmatrix} w_{item[1],1} & w_{item[1],2} \\ w_{item[2],1} & w_{item[2],2} \\ \vdots & \vdots \\ w_{item[n],1} & w_{item[n],2} \end{pmatrix}
\end{aligned} \tag{A.33}
$$

In summary, the hierarchical model has a very general matrix formulation (cf. Laird and Ware 1982):

$$signal = X\zeta + Z_u \odot z_u + Z_w \odot z_w + \varepsilon \tag{A.34}$$

The practical relevance of this matrix formulation is that we can define hierarchical models very compactly and efficiently in Stan by expressing the model in terms of the model matrices (Sorensen, Hohenstein, and Vasishth 2016). As an aside, notice that in the above example, $X = Z_u = Z_w$; but in principle one could have different model matrices for the fixed vs. random effects.

A.8 Treatment contrast with intercept as the grand mean

In chapter 6, we have introduced the treatment contrast, where each contrast compares one condition to a baseline condition. We have discussed that the intercept in the treatment contrast estimates the condition mean for the baseline condition. There are some applications where this behavior may seem suboptimal. This can be the case in experimental designs with multiple factors, where we may want to use centered contrasts (this is discussed in chapter 7). Moreover, the contrast coding of the population-level (or fixed) effects also defines what the group-level (or random) effects assess. If the intercept assesses the grand mean–rather than the baseline condition–in hierarchical models, then the group-level intercepts reflect the grand mean variance, rather than the variance in the baseline condition.

It is possible to design a treatment contrast where the intercept reflects the grand mean (assuming a balanced design; otherwise it's the unweighted grand mean). We implement this using the `hypr` package. The trick is to add the intercept explicitly as a comparison of the average of all four condition means:

```
HcTrGM <- hypr(b0 = ~ (F1 + F2 + F3 + F4) / 4,
               b1 = F2 ~ F1,
               b2 = F3 ~ F1,
               b3 = F4 ~ F1,
               levels = c("F1", "F2", "F3", "F4"))
HcTrGM
```

```
## hypr object containing 4 null hypotheses:
## H0.b0: 0 = (F1 + F2 + F3 + F4)/4  (Intercept)
## H0.b1: 0 = F2 - F1
## H0.b2: 0 = F3 - F1
## H0.b3: 0 = F4 - F1
##
## Call:
## hypr(b0 = ~1/4 * F1 + 1/4 * F2 + 1/4 * F3 + 1/4 * F4, b1 = ~F2 -
##     F1, b2 = ~F3 - F1, b3 = ~F4 - F1, levels = c("F1", "F2",
## "F3", "F4"))
##
## Hypothesis matrix (transposed):
##    b0  b1  b2  b3
## F1 1/4  -1  -1  -1
## F2 1/4   1   0   0
```

```
## F3 1/4   0   1   0
## F4 1/4   0   0   1
##
## Contrast matrix:
##    b0   b1   b2   b3
## F1    1 -1/4 -1/4 -1/4
## F2    1  3/4 -1/4 -1/4
## F3    1 -1/4  3/4 -1/4
## F4    1 -1/4 -1/4  3/4
```

The hypothesis matrix now explicitly codes the intercept as the first column, where all hypothesis weights are equal and sum up to one. This is coding the intercept. The other hypothesis weights are as expected for the treatment contrast. The contrast matrix now looks very different compared to the standard treatment contrast. Next, we fit a model with this adapted treatment contrast. The function `contr.hypothesis` automatically removes the intercept that is encoded in `HcTrGM`, since this is automatically added by `brms`.

```
contrasts(df_contrasts3$F) <- contr.hypothesis(HcTrGM)
fit_TrGM <- brm(DV ~ F,
                data = df_contrasts3,
                family = gaussian(),
                prior = c(prior(normal(20, 50), class = Intercept),
                          prior(normal(0, 50), class = sigma),
                          prior(normal(0, 50), class = b)))
```

```
fixef(fit_TrGM)
```

```
##           Estimate Est.Error   Q2.5 Q97.5
## Intercept    19.94      2.54  14.98  25.0
## Fb1           9.70      6.89  -4.16  23.1
## Fb2          -0.48      6.76 -14.57  13.3
## Fb3          29.45      6.97  15.80  43.2
```

The results show that the coefficients reflect comparisons of each condition $F2$, F3, and F4 to the baseline condition $F1$. The intercept now captures the grand mean across all four conditions of 20.

B

Advanced models with Stan - Extended

B.1 What does `target` do in Stan models?

We can exemplify how `target` works with one hypothetical iteration of the sampler in the model `normal.stan` discussed in section 8.2 and shown below:

```
data {
  int<lower = 1> N;  // Total number of trials
  vector[N] y;  // Score in each trial
}
parameters {
  real mu;
  real<lower = 0> sigma;
}
model {
  // Priors:
  target += normal_lpdf(mu | 0, 20);
  target += lognormal_lpdf(sigma | 3, 1);
  // Likelihood:
  target += normal_lpdf(y | mu, sigma);
}
```

In every iteration where the sampler explores the posterior space, `mu` and `sigma` acquire different values (this is where the Stan algorithm stops the movement of the particle in the Hamiltonian space). Say that in an iteration, `mu = 2.895` and `sigma = 9.002`. Then the following happens in the model block:

1. At the beginning of the iteration, `target` is zero.
2. The transformations that the sampler *automatically* does are taken into account. In our case, although `sigma` is constrained to be positive in our model, inside Stan's sampler it is transformed to an unconstrained space amenable to Hamiltonian Monte Carlo. That is, Stan samples from an auxiliary parameter that ranges from minus infinity to infinity, which is equivalent to `log(sigma)`. This auxiliary parameter is then exponentiated, when it is incorporated into our model.

Because of the mismatch between the constrained parameter space that we defined and the unconstrained space that it is converted to by Stan, an adjustment to the unnormalized posterior is required and added *automatically*. The reasons for this requirement are somewhat complex and will be discussed in section 10. In this particular case, this adjustment (which is the log absolute value of the Jacobian determinant), is equivalent to adding `log(sigma) = 2.197` to `target`.

3. After `target += normal_lpdf(mu | 0, 20);` the log of the density of $Normal(0, 20)$ is evaluated at a given sample of mu (specifically 2.895) and this is added to `target`. In R, this would be `dnorm(x = 2.895, mean = 0, sd = 20, log = TRUE)`, which is equal to `-3.925`. Thus, `target` should be `-3.925 + 2.197 = -1.728`.
4. After `target += lognormal_lpdf(sigma | 3, 1)`, we add the log of the density of $LogNormal(3, 1)$ evaluated at `9.002` to the previous value of the target. In R, this would be `dlnorm(x = 9.002 , mean = 3, sd = 1, log = TRUE)`, which is equal to `-3.438`. Thus, `target` should be updated to `-1.728 + -3.438 = -5.166`.
5. After each iteration of the for-loop in the model block, we add to the target the log density of $Normal(2.895, 9.002)$ evaluated at each of the values of Y. In R, this would be to add `sum(dnorm(Y, 2.895, 9.002, log = TRUE))` (which is equal to `-363.164`) to the current value of `target` `-5.166 + -363.164 = -368.33`.

This means that for the coordinates [mu = 2.895, sigma = 9.002], the height of the unnormalized posterior would be the value `exp(target)` = $\exp(-368.33) = 1.087 \times 10^{-160}$. Incidentally, the value of `target` is returned as `lp__` (log probability) in an object storing a fit model with Stan.

It is possible to expose the value of `target`, by printing `target()` inside a Stan model. The value of `target` after each iteration is named `lp__` in the Stan object. This can be useful for troubleshooting a problematic model.

B.2 Explicitly incrementing the log probability function (`target`) vs. using the sampling or distribution ~ notation

In this book, we specify priors and likelihoods by explicitly incrementing the log-probability function using the following syntax:

```
target +=  pdf_name_lpdf(parameter | ...)
```

However, Stan also allows for specifying priors and likelihood with the so-called sampling or distribution statement notation with the following code.

```
parameter ~ pdf_name(..)
```

Confusingly enough a sampling statement does not perform any actual sampling, and it is meant to be a notational convenience.

The following two lines of code lead to the same behavior in Stan with respect to parameter estimation. There is, nonetheless, an important difference between them.

```
x ~ normal(mu, sigma);
target += normal_lpdf(x | mu, sigma);
```

The important difference is that the sampling notation (the notation with the $\sim$) will *drop normalizing constants*. Consider the following formula that corresponds to the log-transformed PDF of a normal distribution:

$$-log(\sigma) - \frac{log(2\pi)}{2} - \frac{(x-\mu)^2}{2\sigma^2} \tag{B.1}$$

If one uses the sampling notation, Stan will ignore the terms that don't contain parameters, such as $-\frac{log(2\pi)}{2}$. Depending on whether the variable x, the location μ, and the scale σ are data or parameters, Stan will ignore different terms. For example, consider the case of a linear regression. The data y (taking the role of x in the previous equation) is assumed to be normally distributed with a location (μ) and scale (σ) to be estimated. In this case, only $-\frac{log(2\pi)}{2}$ can be dropped, because both $-log(\sigma)$ and $-\frac{(y-\mu)^2}{2\sigma^2}$ contain parameters. Another example where different terms would be dropped is the case of assigning a normal prior distribution to a parameter θ. Here, the location and scale (μ and σ) are data and θ takes the role of x in the previous equation and acts as a parameter. This means that $-log(\sigma)$ is a constant term that can be ignored, but not $-\frac{(\theta-\mu)^2}{2\sigma^2}$ because it contains the parameter θ. Dropping constant terms does not affect parameter estimation because it only affects the unnormalized likelihood in the same way in all the parameter space. To make this more concrete, the whole plot in Figure 8.1 will move up or down by some constant amount, and this won't affect the Hamiltonian dynamics that determine how we sample from the posterior.

The advantage of the sampling notation is that it can be faster (when many terms are ignored), but the disadvantage is that (i) it is not compatible with the calculation of Bayes factor with bridge sampling (see section 13.4 in chapter 13), or the calculation of the log-likelihood for cross-validation (see chapter 14), (ii) it misleads us into thinking that Stan is actually sampling the left term

in the sampling statement, e.g., drawing y from a normal distribution in the previous example, when in fact at each step the log-probability (`target`) is incremented based on the parameter values determined by Hamiltonian dynamics (as explained before), and (iii) it makes it less straightforward to transition to more complex models where the sampling notation cannot be used (as in, for example, mixture models in chapter 17).

If one is not going to use Bayes factor with bridge sampling or cross-validation, the same speed advantage of the sampling notation can also be achieved by incrementing the log-probability with log-unnormalized probability density or mass functions (functions ending with `_lupdf` or `_lupmf`). The previous example would be translated into the following:

```
target += normal_lupdf(y | mu, sigma);
```

B.3 An alternative R interface to Stan: `cmdstanr`

At the time of writing this, there are two major nuisances with `rstan`, (i) the R code interfaces directly with C++ creating installation problems in many systems, (ii) `rstan` releases lag behind Stan language releases considerably preventing the user from taking advantage of the latest features of Stan. The package `cmdstanr` (https://mc-stan.org/cmdstanr/) is a lightweight interface to Stan for R that solves these problems. The downside (at the moment of writing this) is that, being lightweight, some functionality of `rstan` is lost, such as looking up functions with `lookup()`, as well as using the fitted model with the `bridgesampling` package to generate Bayes factors. Furthermore, the package `cmdstanr` is currently under development and the application programming interface (API) might still change. However, the user interested in an easy (and painless) installation and the latest features of Stan might find it useful.

Once `cmdstanr` is installed, we can use it as follows:

First create a new `CmdStanModel` object from a file containing a Stan program using `cmdstan_model()`

```
normal <- system.file("stan_models",
                      "normal.stan",
                      package = "bcogsci")
normal_mod <- cmdstan_model(normal)
```

The object `normal_mod` is an R6 reference object (https://r6.r-lib.org/). This class of object behaves similarly to objects in object oriented programming

languages, such as python. Methods are accessed using `$` (rather than `.` as in python).

To sample, use the `$sample()` method. The data argument accepts a list (as we used in `stan()` from `rstan`). However, many of the arguments of `$sample` have different names than the ones used in `stan()` from the `rstan` package:

```
lst_score_data <- list(y = y, N = length(y))
fit_normal_cmd <- normal_mod$sample(data = lst_score_data,
                                    seed = 123,
                                    chains = 4,
                                    parallel_chains = 4,
                                    iter_warmup = 1000,
                                    iter_sampling = 1000)
```

To show the posterior summary, access the method `$summary()` of the object `fit_normal_cmd`.

```
fit_normal_cmd$summary()
```

Access the samples of `fit_normal_cmd` using `$draws()`.

```
fit_normal_cmd$draws(variables = "mu")
```

The vignette of https://mc-stan.org/cmdstanr/ shows more use cases, and how the samples can be transformed into other formats (data frame, matrix, etc.) together with the package `posterior` (https://mc-stan.org/cmdstanr/).

B.4 Matrix, vector, or array in Stan?

Stan contains three basic linear algebra types, `vector`, `row_vector`, and `matrix`. But Stan also allows for building arrays of any dimension from any type of element (integer, real, etc.). This means that there are several ways to define one-dimensional N-sized containers of real numbers,

```
array[N] real a;
vector[N] a;
row_vector[N] a;
```

as well as, two-dimensional N1×N2-sized containers of real numbers:

```
array[N1, N2] real m;
matrix[N1, N2] m;
array[N1] vector[N2] b;
array[N1] row_vector[N2] b;
```

These distinctions affect either what we can do with these variables, or the speed of our model, and sometimes are interchangeable. Matrix algebra is only defined for (row) vectors and matrices, that is we cannot multiply arrays. The following line requires all the one-dimensional containers (`p_size` and `c_load`) to be defined as vectors of the same size (or all as row vectors):

```
p_size = alpha + c_load * beta;
```

Many "vectorized" operations are also valid for arrays, that is, `normal_lpdf`, accepts (row) vectors (as we did in our code) or arrays as in the next example. There is of course no point in converting a vector to an array as follows, but this shows that Stan allows both type of one-dimensional containers.

```
array[N] real mu = to_array_1d(alpha + c_load * beta);
target += normal_lpdf(p_size | mu, sigma);
```

By contrast, the outcome of "vectorized" pseudorandom number generator (`_rng`) functions can only be stored in an array. The following example shows the only way to vectorize this type of function:

```
array[N] real p_size_pred = normal_rng(alpha + c_load * beta,
                                       sigma);
```

Alternatively, one can always use a for-loop, and it won't matter if `p_size_pred` is an array or a vector:

```
vector[N] p_size_pred;
for(n in 1:N)
    p_size_pred[n] = normal_rng(alpha + c_load[n] * beta, sigma);
```

See also Stan's user's guide section on matrices, vector, and arrays (Stan Development Team 2024, Chapter 18 of the User's guide).

B.5 A simple non-centered parameterization

Stan's sampler can explore the parameter space more easily if its step size is appropriate for all the parameters. This is achieved when there are no strong dependencies between parameters. In section 9.1.2, we fit an uncorrelated varying intercept and slopes model with Stan, and we assume the following.

$$\mathbf{u}_2 \sim Normal(0, \tau_{u_2}) \tag{B.2}$$

where $\mathbf{u}_2$ is the column vector of $u_{i,2}$'s. The index i refers to the subject id.

We can transform a vector v into z-scores as follows.

$$\mathbf{z}_v = \frac{\mathbf{v} - mean(\mathbf{v})}{SD(\mathbf{v})} \tag{B.3}$$

Transforming the parameter u_2 into z-scores amounts to centering it, so we can call this a centered parameterization.

$$\mathbf{z}_{u_2} = \frac{\mathbf{u}_2 - 0}{\tau_{u_2}} \tag{B.4}$$

where

$$\mathbf{z}_{u_2} \sim Normal(0, 1) \tag{B.5}$$

Now $\mathbf{z}_{u_2}$ is easier to sample because it doesn't depend on other parameters (in particular, it is no longer conditional on τ) and its scale is 1. Once we have sampled this centered parameter, we can derive the actual parameter we care about by carrying out the inverse operation, which is called a non-centered parameterization:

$$\mathbf{u}_2 = \mathbf{z}_{u_2} \cdot \tau_{u_2} \tag{B.6}$$

A question that might be raised here is whether using a non-centered parameterization is always a good idea. Betancourt and Girolami (2015) point out that the extremeness of the dependency depends on the amount of data, and the efficacy of the parameterization depends on the strength of the data (on how informative the data is). When there is enough data, this parameterization is unnecessary and it may be more efficient to use the centered parameterization. However, cases where there is enough data to render this parameterization useless might also be cases where the partial pooling of the hierarchical models may not be needed in the first place. Although data from conventional lab experiments in psychology, psycholinguistics, and related areas seem to benefit from the non-centered parameterization, the jury is still out for larger data sets with thousands of subjects from crowdsourcing websites.

B.6 Cholesky factorization for reparameterizing hierarchical models with correlations between adjustments to different parameters

First, some definitions that we will need below. A matrix is square if the number of rows and columns is identical. A square matrix $\boldsymbol{A}$ is symmetric if $\boldsymbol{A}^T = \boldsymbol{A}$, i.e., if transposing the matrix gives the matrix back. Suppose that $\boldsymbol{A}$ is a known matrix with real numbers. If $\boldsymbol{x}$ is a vector of variables with length p (a $p \times 1$ matrix), then $x^T Ax$ is called a quadratic form in x ($x^T Ax$ will be a scalar, 1×1). If $x^T Ax > 0$ for all x, then $\boldsymbol{A}$ is a positive definite matrix. If $x^T Ax \geq 0$ for all $x \neq 0$, then $\boldsymbol{A}$ is positive semi-definite.

We encountered correlation matrices first in section 1.6.3. A correlation matrix is always symmetric, has ones along the diagonal, and real values ranging between -1 and 1 on the off-diagonals. Given a correlation matrix $\mathbf{R_u}$, we can decompose it into a lower triangular matrix $\mathbf{L_u}$ such that $\mathbf{L_u L_u}^T = \mathbf{R_u}$. The matrix $\mathbf{L_u}$ is called the Cholesky factor of $\mathbf{R_u}$. Intuitively, you can think of $\mathbf{L_u}$ as the matrix equivalent of the square root of $\mathbf{R_u}$. More details on the Cholesky factorization can be found in Gentle (2007).

$$\mathbf{L_u} = \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix} \tag{B.7}$$

For a model without a correlation between adjustments for the intercept and slope as the one in section 9.1.2, we assumed that adjustments u_1 and u_2 were generated by two independent normal distributions. But in section 9.1.3, we want to allow the possibility that the adjustments can have a non-zero correlation. We can use the Cholesky factorization to generate correlated random variables in the following way.

1. Generate uncorrelated vectors, z_{u_1} and z_{u_2}, for each vector of adjustments u_1 and u_2, as sampled from $Normal(0, 1)$:

$$\begin{aligned} z_{u_1} &\sim Normal(0, 1) \\ z_{u_2} &\sim Normal(0, 1) \end{aligned}$$

2. By multiplying the Cholesky factor with the z's, generate a matrix that contains two row vectors of correlated variables (with standard deviation 1).

$$\mathbf{L_u} \cdot \mathbf{z_u} = \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} z_{u_{1,subj=1}} & z_{u_{1,subj=2}} & ... & z_{u_{1,subj=N_{subj}}} \\ z_{u_{2,subj=1}} & z_{u_{2,subj=2}} & ... & z_{u_{2,subj=N_{subj}}} \end{pmatrix}$$

$$= \begin{pmatrix} L_{11} \cdot z_{u_{1,1}} + 0 \cdot z_{u_{2,1}} & ... & L_{11} \cdot z_{u_{1,N_{subj}}} + 0 \cdot z_{u_{2,1}} \\ L_{21} \cdot z_{u_{1,1}} + L_{22} \cdot z_{u_{2,1}} & ... & L_{11} \cdot z_{u_{1,N_{subj}}} + L_{22} \cdot z_{u_{2,N_{subj}}} \end{pmatrix}$$

A very informal explanation of why this works is that we are making the variable that corresponds to the slope to be a function of a scaled version of the intercept.

3. The last step is to scale the previous matrix to the desired standard deviation. We define the diagonalized matrix $diag\_matrix(\tau_u)$ as before:

$$\begin{pmatrix} \tau_{u_1} & 0 \\ 0 & \tau_{u_2} \end{pmatrix}$$

Then pre-multiply it by the correlated variables with standard deviation 1 from before:

$$\mathbf{u} = diag\_matrix(\tau_u) \cdot \mathbf{L_u} \cdot \mathbf{z_u} =$$

$$\begin{pmatrix} \tau_{u_1} & 0 \\ 0 & \tau_{u_2} \end{pmatrix} \begin{pmatrix} L_{11} \cdot z_{u_{1,1}} & ... \\ L_{21} \cdot z_{u_{1,1}} + L_{22} \cdot z_{u_{2,1}} & ... \end{pmatrix}$$

$$\begin{pmatrix} \tau_{u_1} \cdot L_{11} \cdot z_{u_{1,1}} & \tau_{u_1} \cdot L_{11} \cdot z_{u_{1,2}} & ... \\ \tau_{u_2} \cdot (L_{21} \cdot z_{u_{1,1}} + L_{22} \cdot z_{u_{2,1}}) & \tau_{u_2} \cdot (L_{21} \cdot z_{u_{1,2}} + L_{22} \cdot z_{u_{2,2}}) & ... \end{pmatrix}$$

It might be helpful to see how one would implement this in R:

Let's assume a correlation of 0.8.

```
rho_u <- 0.8
# Correlation matrix
(R_u <- matrix(c(1, rho_u, rho_u, 1), ncol = 2))
```

```
##      [,1] [,2]
## [1,]  1.0  0.8
## [2,]  0.8  1.0
```

```
# Cholesky factor:
# (Transpose it so that it looks the same as in Stan)
(L_u <- t(chol(R_u)))
```

```
##      [,1] [,2]
## [1,]  1.0  0.0
## [2,]  0.8  0.6
```

```
# Verify that we recover R_u,
# Recall that %*% indicates matrix multiplication
L_u %*% t(L_u)
```

```
##      [,1] [,2]
## [1,]  1.0  0.8
## [2,]  0.8  1.0
```

1. Generate uncorrelated z from a standard normal distribution assuming only 10 subjects.

```
N_subj <- 10
(z_u1 <- rnorm(N_subj, 0, 1))
```

```
##  [1]  0.6366 -0.4838  0.5169  0.3690 -0.2154  0.0653 -0.0341  2.1285
##  [9] -0.7413 -1.0960
```

```
(z_u2 <- rnorm(N_subj, 0, 1))
```

```
##  [1]  0.0378  0.3105  0.4365 -0.4584 -1.0633  1.2632 -0.3497 -0.8655
##  [9] -0.2363 -0.1972
```

2. Create matrix of correlated parameters.

First, create a matrix with the uncorrelated parameters:

```
# matrix z_u
(z_u <- matrix(c(z_u1, z_u2), ncol = N_subj, byrow = TRUE))
```

```
##        [,1]   [,2]  [,3]   [,4]   [,5]   [,6]    [,7]   [,8]   [,9]
## [1,] 0.6366 -0.484 0.517  0.369 -0.215 0.0653 -0.0341  2.128 -0.741
```

```
## [2,] 0.0378  0.310 0.437 -0.458 -1.063 1.2632 -0.3497 -0.866 -0.236
##       [,10]
## [1,] -1.096
## [2,] -0.197
```

Then, generate correlated parameters by pre-multiplying the $\mathbf{z}_u$ matrix with $\mathbf{L}_u$.

```
L_u %*% z_u
```

```
##       [,1]   [,2]  [,3]   [,4]   [,5]   [,6]    [,7] [,8]   [,9]
## [1,] 0.637 -0.484 0.517 0.3690 -0.215 0.0653 -0.0341 2.13 -0.741
## [2,] 0.532 -0.201 0.675 0.0202 -0.810 0.8101 -0.2370 1.18 -0.735
##       [,10]
## [1,] -1.096
## [2,] -0.995
```

3. Use the following diagonal matrix to scale the z_u.

```
tau_u1 <- 0.2
tau_u2 <- 0.01
(diag_matrix_tau <- diag(c(tau_u1,  tau_u2)))
```

```
##      [,1] [,2]
## [1,]  0.2 0.00
## [2,]  0.0 0.01
```

4. Finally, generate the adjustments for each subject u:

```
(u <- diag_matrix_tau %*% L_u %*% z_u)
```

```
##         [,1]     [,2]    [,3]     [,4]    [,5]   [,6]
## [1,] 0.12731 -0.09676 0.10337 0.073793 -0.0431 0.0131
## [2,] 0.00532 -0.00201 0.00675 0.000202 -0.0081 0.0081
##          [,7]   [,8]     [,9]    [,10]
## [1,] -0.00681 0.4257 -0.14827 -0.21920
## [2,] -0.00237 0.0118 -0.00735 -0.00995
```

```
# The rows are correlated, approximately 0.8
cor(u[1, ], u[2, ])
```

```
## [1] 0.85
```

```
# The variance components can be recovered as well:
sd(u[1, ])
```

```
## [1] 0.179
```

```
sd(u[2, ])
```

```
## [1] 0.00751
```

B.7 Different rank visualizations and the SBC package.

Implementing the simulation-based calibration algorithm "by hand", as we did in section 10.2, introduces a new source of potential errors. Fortunately, the R package SBC (Kim et al. 2024) provides tools to validate a Stan model (or any sampling algorithm) by allowing us to run simulation-based calibrations easily. The package is in active development at the moment[1] and can be installed with the following command.

```
remotes::install_github("hyunjimoon/SBC")
```

One of the main advantages of this package is that it provides several ways to visualize the results of the simulation-based calibration procedure; see https://hyunjimoon.github.io/SBC/. Figure B.1 shows rank histograms produced by SBC of a correct model and several different incorrect models. An alternative to rank histograms is to use an empirical cumulative distribution function (ECDF)-based method, as proposed by Säilynoja, Bürkner, and Vehtari (2022). The idea behind this method is that if the ranks produced by the simulation-based calibration algorithm are uniform the ECDF of the ranks should be close to the CDF of a uniform distribution. Figure B.2 shows the difference between the ECDF of the ranks and the CDF of a uniform distribution together with 95% confidence bands (this is the default in the SBC package) for a correct model and different incorrect ones.

[1] Even though the package is already fully functional, function names and arguments might change by the time this book is published.

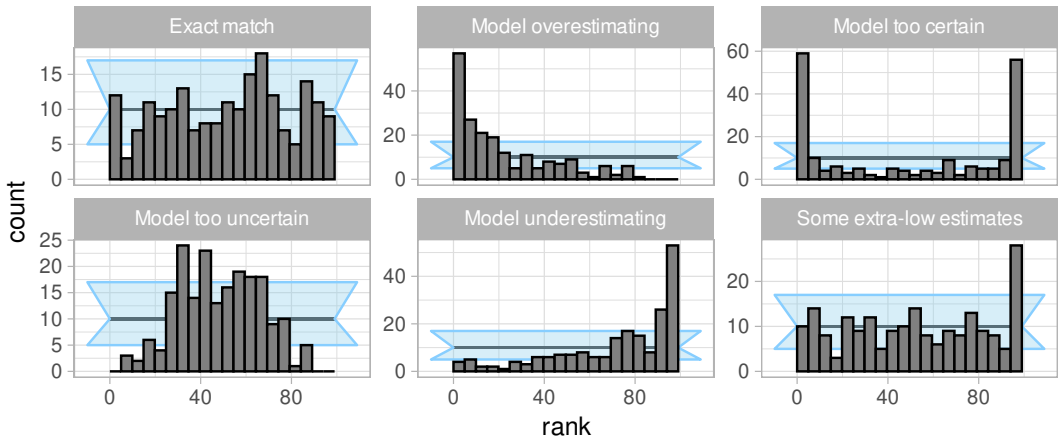

FIGURE B.1: Rank histograms produced by the R package SBC showing the outcome one would expect for a correct model and for several different incorrect ones together with 95% confidence bands (these are the default bands in the package).

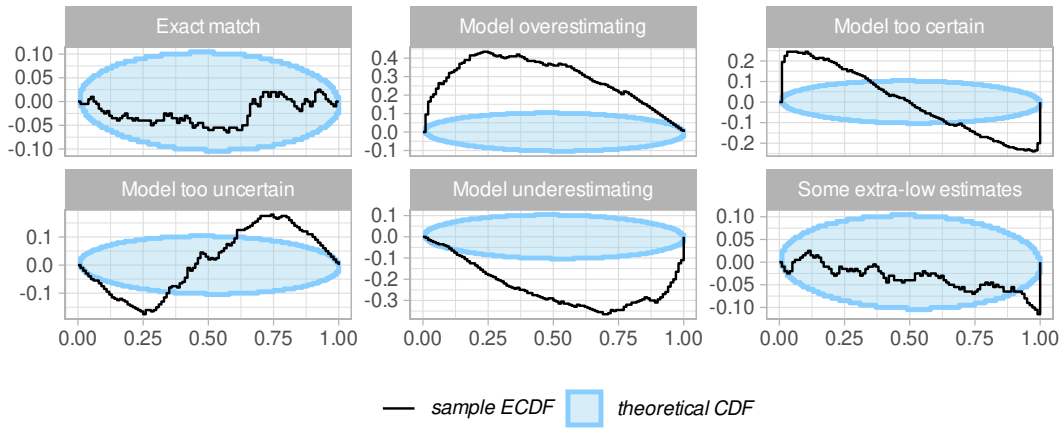

FIGURE B.2: Difference between the perfectly uniform CDF and empirical cumulative distribution function (ECDF) of the ranks produced by the SBC R package together with 95% confidence bands. The figure shows the outcome one would expect for a correct model and for several different incorrect ones.

C

Evidence synthesis and measurements with error - Extended

C.1 What happens if we set `sigma = TRUE` in `resp_se()` function in `brms`?

If we modify the meta-analysis `brms` model in section 11.1.1.1 by setting `sigma = TRUE` in the `resp_se()` function, we won't be able to get estimates for ζ_n. This is because these estimates will be handled implicitly. The model presented formally in Equation (11.3), repeated here as (C.1) is equivalent to the one in Equation (C.2). A critical difference is that ζ_n does not appear any more.

$$\begin{aligned} \text{effect}_n &\sim Normal(\zeta_n, SE_n) \\ \zeta_n &\sim Normal(\zeta, \tau) \\ \zeta &\sim Normal(0, 100) \\ \tau &\sim Normal_+(0, 100) \end{aligned} \tag{C.1}$$

$$\begin{aligned} \text{effect}_n &\sim Normal(\zeta, \sqrt{\tau^2 + SE_n^2}) \\ \zeta &\sim Normal(0, 100) \\ \tau &\sim Normal_+(0, 100) \end{aligned} \tag{C.2}$$

where $n = 1, \ldots, N_{studies}$

This works because of the following property of normally distributed random variables:

If X and Y are two independent random variables, and

$$\begin{aligned} X &\sim Normal(\mu_X, \sigma_X) \\ Y &\sim Normal(\mu_Y, \sigma_Y) \end{aligned} \tag{C.3}$$

then, Z, the sum of these two random variables is:

$$Z = X + Y \tag{C.4}$$

The distribution of Z has the following form:

$$Z \sim Normal\left(\mu_X + \mu_Y, \sqrt{\sigma_X^2 + \sigma_Y^2}\right) \tag{C.5}$$

In our case, let

$$\begin{aligned} U_n &\sim Normal(0, SE_n) \\ \zeta_n &\sim Normal(\zeta, \tau) \end{aligned} \tag{C.6}$$

Analogous to Equations (C.4) and (C.5), effect$_n$ can be expressed as a sum of two independent random variables:

$$\text{effect}_n = U_n + \zeta_n \tag{C.7}$$

The distribution of effect$_n$ will be

$$\text{effect}_n \sim Normal\left(\zeta, \sqrt{SE^2 + \tau^2}\right) \tag{C.8}$$

We can fit this in `brms` as follows. In this model specification, one should not include the `+ (1 | study_id)`, and the prior for τ should now be specified for `sigma`.

```
priors2 <- c(prior(normal(0, 100), class = Intercept),
             prior(normal(0, 100), class = sigma))
fit_sbi_sigma <- brm(effect | resp_se(`SE`, sigma = TRUE) ~ 1,
                     data = df_sbi,
                     prior = priors2,
                     control = list(adapt_delta = .99,
                                    max_treedepth = 10))
```

There are slight differences with `fit_sbi` from section 11.1.1.1 due to the different parameterization and the sampling process, but the results are very similar:

```
posterior_summary(fit_sbi_sigma,
                  variable = c("b_Intercept", "sigma"))
```

```
##             Estimate Est.Error  Q2.5 Q97.5
## b_Intercept     13.1      6.43 2.359  28.3
## sigma           11.6      7.99 0.504  30.1
```

Compare this with the original model:

```
fit_sbi <- brm(effect | resp_se(`SE`, sigma = FALSE) ~
                 1 + (1 | study_id),
               data = df_sbi,
               prior = priors,
               control = list(adapt_delta = .99, max_treedepth = 10))
```

```
posterior_summary(fit_sbi,
                  variable = c("b_Intercept", "sigma"))
```

```
##             Estimate Est.Error Q2.5 Q97.5
## b_Intercept     13.3      6.23 2.62  27.1
## sigma            0.0      0.00 0.00   0.0
```

If we are not interested in the underlying effects in each study, this parameterization of the meta-analysis can be faster and more robust (i.e., it has less potential convergence issues). A major drawback is that we can no longer display a forest plot as we do in Figure 11.1.

D

Model comparison - Extended

D.1 Credible intervals should not be used to reject a null hypothesis

Researchers often incorrectly use credible intervals for null hypothesis testing, that is, to test whether a parameter β is zero or not. A common approach is to check whether zero is included in the 95% credible interval for the parameter β; if it is, then the null hypothesis that the effect is zero is accepted; and if zero is outside the interval, then the null is rejected. For example, in a tutorial paper that two of the authors of this book wrote (Nicenboim and Vasishth 2016), we incorrectly suggest that the credible interval can be used to reject the hypothesis that the β is zero. This is generally not the correct approach. The problem with this approach is that it is a heuristic that will work in some cases and might be misleading in others (for an example, see Vasishth et al. 2022). Unfortunately, when they will work or not is in fact not well-defined.

Why is the credible-interval approach only a heuristic? One line of (generally incorrect) reasoning that justifies looking at the overlap between credible intervals and zero is based on the fact that the most likely values of β lie within 95% credible interval.[1] This entails that if zero is outside the interval, it must have a low probability density. This is true, but it's meaningless: Regardless of where zero lies (or any point value), zero will have a probability mass of exactly zero since we are dealing with a continuous distribution. The lack of overlap doesn't tell us how much posterior probability the null model has.

A partial solution could be to look at a probability interval close to zero rather than zero (e.g., an interval of, say, -2 to 2 ms in a response time experiment), so that we obtain a non-zero probability mass. While the lack of overlap would be slightly more informative, excluding a small interval can be problematic when the prior probability mass of that interval is very small to begin with (as was the case with the regularizing priors we assigned to our parameters).

[1] This is also strictly true only in a highest density interval (HDI), this is a credible interval where all the points within the interval have a higher probability density than points outside the interval. However, when posterior distributions are symmetrical, these intervals are virtually identical to the equal-tail intervals we use in this book.

Rouder, Haaf, and Vandekerckhove (2018) show that if prior probability mass is added to the point value zero using a *spike-and-slab* prior (or if probability mass is added to the small interval close to zero if one considers that equivalent to the null model), looking at whether zero is in the 95% credible interval is analogous to the Bayes factor. Unfortunately, the *spike-and-slab* prior cannot be incorporated in Stan, because it relies on a discrete parameter. However, other programming tools (like PyMC3, JAGS, or Turing) can be used if such a prior needs to be fit; see the further readings at the end of the chapter.

Rather than looking at the overlap of the 95% credible interval, we might be tempted to conclude that there is evidence for an effect because the probability that a parameter is positive is high, that is $P(\beta > 0) >> 0.5$. However, the same logic from the previous paragraph renders this meaningless. Given that the probability mass of a point value, $P(\beta = 0)$, is zero, what we can conclude from $P(\beta > 0) >> 0.5$ is that β is very likely to be positive rather than negative, but we can't make any assertions about whether β is exactly zero.

As we saw, the main problem with these heuristics is that they ignore that the null model is a separate hypothesis. In many situations, the null hypothesis may not be of interest, and it might be perfectly fine to base our conclusions on credible intervals or $P(\beta > 0)$. The problem arises when these heuristics are used to provide evidence in favor or against the null hypothesis. If one wants to argue about the evidence in favor of or against a null hypothesis, Bayes factors or cross-validation will be needed. These are discussed in the next two chapters.

How can credible intervals be used sensibly? The region of practical equivalence (ROPE) approach (Spiegelhalter, Freedman, and Parmar 1994; Freedman, Lowe, and Macaskill 1984; and, more recently, Kruschke and Liddell 2018; Kruschke 2014) is a reasonable alternative to hypothesis testing and arguing for or against a null. This approach is related to the spike-and-slab discussion above. In the ROPE approach, one can define a range of values for a target parameter that is predicted before the data are seen. Of course, there has to be a principled justification for choosing this range a priori; an example of a principled justification would be the prior predictions of a computational model. Then, the overlap (or lack thereof) between this predicted range and the observed credible interval can be used to infer whether one has estimates consistent (or partly consistent) with the predicted range. Here, we are not ruling out any null hypothesis, and we are not using the credible interval to make a decision like "the null hypothesis is true/false."

There is one situation where credible intervals could arguably be used to carry out a null hypothesis test. When priors are flat, credible intervals can show frequentist properties, making it reasonable to check whether zero falls within

the credible interval. For example, Newall et al. (2023) use credible intervals as confidence intervals after calibration. They explicitly verify that 5% of the 95% credible intervals exclude zero when no effect exists. When using such an approach, a verification step would be necessary. We don't discuss this approach any further because our aim in this part of the book is not to derive frequentist statistics from Bayesian analysis, but to use Bayesian methods for obtaining posterior probabilities and Bayes factors, focusing on Bayesian hypothesis testing.

D.2 The likelihood ratio vs the Bayes factor

The likelihood ratio test is a very similar, but frequentist, approach to model comparison and hypothesis testing, which also compares the likelihood for the data given two different models. We show this here to highlight the similarities and differences between frequentist and Bayesian hypothesis testing. In contrast to the Bayes factor, the likelihood ratio test depends on the "best" (i.e., the maximum likelihood) estimate for the model parameter(s), that is, the model parameter θ occurs on the right side of the semi-colon in the equation for each likelihood. (An aside: we do not use a conditional statement, i.e., the vertical bar, when talking about likelihood in the frequentist context; instead, we use a semi-colon. This is because the statement $f(y \mid \theta)$ is a conditional statement, implying that θ has a probability density function associated with it; in the frequentist framework, parameters cannot have a pdf associated with them, they are assumed to have fixed, point values.)

$$LikRat = \frac{P(\boldsymbol{y}; \hat{\boldsymbol{\Theta}}_{\mathbf{1}}, \mathscr{M}_1)}{P(\boldsymbol{y}; \hat{\boldsymbol{\Theta}}_{\mathbf{2}}, \mathscr{M}_2)} \tag{D.1}$$

That means that in the likelihood ratio test, each model is tested on its ability to explain the data using this "best" estimate for the model parameter (here, the maximum likelihood estimate $\hat{\theta}$). That is, the likelihood ratio test reduces the full range of possible parameter values to a point value, leading to overfitting the model to the maximum likelihood estimate (MLE). If the MLE badly misestimates the true value of the parameter (point value), due to Type M error (Gelman and Carlin 2014), we could end up with a "significant" effect that is just a consequence of this misestimation (it will not be consistently replicable; see Vasishth et al. 2018 for an example). By contrast, the Bayes factor involves range hypotheses, which are implemented via integrals over the model parameter; that is, it uses marginal likelihoods that are averaged across all possible prior values of the model parameter(s). Thus, if, due to

Type M error, the best point estimate (the MLE) for the model parameter(s) is not very representative of the possible values for the model parameter(s), then Bayes factors will be superior to the frequentist likelihood ratio test (see exercise G.13.2). An additional difference, of course, is that Bayes factors rely on priors for estimating each model's parameter(s), whereas the frequentist likelihood ratio test does not (and cannot) consider priors in the estimation of the best-fitting model parameter(s). As we show in this chapter, this has far-reaching consequences for Bayes factor-based model comparison.

D.3 Approximation of the (expected) log predictive density of a model without integration

To compare models based on their predictive accuracy, we often use the expected log predictive density ($elpd$), which evaluates how well a model's predictions align with likely future data. In previous sections, we introduced the idea that the $elpd$ can be calculated by integrating over all possible future data, weighting predictions by their likelihood under the true data-generating process. However, because the true data-generating distribution, p_t, is unknown, we instead use the observed data distribution as a proxy. This allows us to approximate the $elpd$ by summing the posterior predictive density of our observed data points, assuming they reflect the distribution of future data.

As an example, imagine that there are N observations in an experiment. Suppose also that the true generative process (which is normally always unknown to us) is a Beta distribution:

$$p_t(y) = Beta(y|1, 3) \tag{D.2}$$

Set N and observe some simulated data y:

```
N <- 10000
y_data <- rbeta(N, 1, 3)
head(y_data)
```

```
## [1] 0.5239 0.0487 0.3031 0.0625 0.4269 0.0669
```

Let's say that we fit the Bayesian model $\mathscr{M}_1$, and somehow, after getting the posterior distribution, we are able to derive the analytical form of its posterior predictive distribution for the model:

$$p(y_{pred}|y, \mathscr{M}_1) = Beta(y_{pred}|2, 2) \tag{D.3}$$

This distribution will tell us how likely different future observations will be, and it also entails that our future observations will be bounded by 0 and 1. (Any observation outside this range will have a probability density of zero).

Imagine that we could know the true distribution of the data, p_t, which is conveniently close to our posterior predictive distribution. This means that Equation (14.4), repeated below, is simple enough, and we know all its terms:

$$elpd = u(\mathscr{M}_1) = \int_{y_{pred}} p_t(y_{pred}) \log p(y_{pred} \mid y, \mathscr{M}_1)\, dy_{pred} \tag{D.4}$$

We can compute this quantity in R. Notice that we don't introduce the data at any point. However, the data had to be used when `p`, the posterior predictive distribution, was derived; we skipped that step here.

```
# True distribution:
p_t <- function(y) dbeta(y, 1, 3)
# Predictive distribution:
p <- function(y) dbeta(y, 2, 2)
# Integration:
integrand <- function(y) p_t(y) * log(p(y))
integrate(f = integrand, lower = 0, upper = 1)$value
```

```
## [1] -0.375
```

Because we will never know `p_t`, this integral can be approximated using the data, `y_data`. It is possible to approximate the integration without any reference to `p_t`; see Equation (14.5):

```
1/N * sum(log(p(y_data)))
```

```
## [1] -0.358
```

The main problem with this approach is that we are using `y_data` twice, once to derive `p`, the predictive posterior distribution, and once for the approximation of *elpd*. We'll see that cross-validation approaches rely on deriving the posterior predictive distribution with part of the data, and estimating the approximation to *elpd* with unseen data. (Don't worry that we don't know the analytical form of the posterior predictive distribution: we saw that we could generate samples from that distribution based on the distribution we use as the likelihood and our posterior samples.)

D.4 The cross-validation algorithm for the expected log predictive density of a model

Here we spell out the Bayesian cross-validation algorithm in detail:

1. Split the data pseudo-randomly into K held-out or validation sets D_k, (where $k = 1, \ldots, K$) that are a fraction of the original data, and K training sets, D_{-k}. The length of the held-out data vector D_k is approximately $1/K$-th the size of the full data set. It is common to use $K = 10$ for K-fold-CV. For LOO-CV, K should be set to the number of observations.

2. Fit K models using each of the K training sets, and obtain posterior distributions $p_{-k}(\Theta) = p(\Theta \mid D_{-k})$, where Θ is the vector of model parameters.

3. Each posterior distribution $p(\Theta \mid D_{-k})$ is used to compute the predictive accuracy (calculated as $\widehat{elpd}$) for each held-out data-point y_n in the vector D_k:

$$\widehat{elpd}_n = \log p(y_n \mid D_{-k}) \text{ with } y_n \in D_k \tag{D.5}$$

Given that the posterior distribution $p(\Theta \mid D_{-k})$ is summarized by S samples, the log predictive density for each data point y_n in a data vector D_k can be approximated as follows:

$$\widehat{elpd}_n = \log\left(\frac{1}{S}\sum_{s=1}^{S} p(y_n \mid \Theta^{k,s})\right) \tag{D.6}$$

where $\Theta^{k,s}$ corresponds to the sample s of the posterior of the model fit to the training set D_k.

5. We obtain the $elpd_{kfold}$ (or $elpd_{loo}$) for all the held-out data points by summing up the $\widehat{elpd}_n$:

$$elpd_{kfold} = \sum_{n=1}^{N} \widehat{elpd}_n \tag{D.7}$$

The standard deviation of the sampling distribution (the standard error) can

be computed by multiplying the standard deviation (or square root of variance) of the N components by $\sqrt{N}$. Letting $\widehat{ELPD}$ be the vector $\widehat{elpd}_1, \ldots, \widehat{elpd}_N$, the standard error is computed as follows:

$$se(\widehat{elpd}) = \sqrt{N\, Var(\widehat{ELPD})} \tag{D.8}$$

The difference between the $elpd_{kfold}$ of two competing models, $\mathscr{M}_1$ and $\mathscr{M}_2$, is a measure of relative predictive performance. The standard error of their difference can be computed using the formula discussed in Vehtari, Gelman, and Gabry (2017):

$$se(\widehat{elpd}_{\mathscr{M}1} - \widehat{elpd}_{\mathscr{M}2}) = \sqrt{N\, Var(\widehat{ELPD}_{\mathscr{M}1} - \widehat{ELPD}_{\mathscr{M}2})} \tag{D.9}$$

E

The Art and Science of Prior Elicitation

Nothing strikes fear into the heart of the newcomer to Bayesian methods more than the idea of specifying priors for the parameters in a model. On the face of it, this concern seems like a valid one; how can one know what the plausible parameter values are in a model before one has even seen the data? In reality, this worry is purely a consequence of the way we are normally taught to carry out data analysis, especially in areas like psychology and linguistics. Model fitting is considered to be a black-box activity, with the primary concern being whether the effect of interest is "significant" or "non-significant." As a consequence of the training that we receive in areas like psychology and (psycho)linguistics, we are taught to focus on one thing (the p-value) and we learn to ignore the estimates (and the uncertainty of those estimates) that we obtain from the model; it becomes irrelevant whether the effect of interest has a mean value of 500 ms (in a reading study, say) or 10 ms; all that matters is whether it is a significant effect or not. In fact, the way many scientists summarize the literature in their field is by classifying studies into two bins: significant and non-significant. There are obvious problems with this classification method; for example, $p = 0.051$ might be counted as "marginally" significant, but $p = 0.049$ is never counted as marginally non-significant. But there will usually not be any important difference between these two borderline values. Real-life examples of such a binary classification approach are seen in Phillips, Wagers, and Lau (2011) and Hammerly, Staub, and Dillon (2019). Because the focus is on significance, we never develop a sense of what the estimates of an effect are likely to be in a future study. This is why, when faced with a prior-distribution specification problem, we are misled into feeling like we know nothing about the quantitative estimates relating to a problem we are studying.

Prior specification has a lot in common with something that physicists call a Fermi problem. As Von Baeyer (1988) describes it: "A Fermi problem has a characteristic profile: Upon first hearing it, one doesn't have even the remotest notion what the answer might be. And one feels certain that too little information exists to find a solution. Yet, when the problem is broken down into subproblems, each one answerable without the help of experts or reference books, an estimate can be made …". Fermi problems in the physics

context are situations where one needs ballpark (approximate) estimates of physical quantities in order to proceed with a calculation. The name comes from a physicist, Enrico Fermi; he developed the ability to carry out fairly accurate back-of-the-envelope calculations when working out approximate numerical values needed for a particular computation. Von Baeyer (1988) puts it well: "Prudent physicists—those who want to avoid false leads and dead ends—operate according to a long-standing principle: Never start a lengthy calculation until you know the range of values within which the answer is likely to fall (and, equally important, the range within which the answer is unlikely to fall)." As in physics, so in data analysis: as Bayesians, we need to acquire the ability to work out plausible ranges of values for parameters. This is a learnable skill, and improves with practice. With time and practice, we can learn to become prudent data analysts.

As Spiegelhalter, Abrams, and Myles (2004) point out, there is no one "correct" prior distribution. One consequence of this fact is that a good Bayesian analysis always takes a range of prior specifications into account; this is called a sensitivity analysis. We have already seen examples of this, but more examples will be provided in this and later chapters.

Prior specification requires the estimation of probabilities. Human beings are not good at estimating probabilities, because they are susceptible to several kinds of biases (Kadane and Wolfson 1998; Spiegelhalter, Abrams, and Myles 2004). We list the most important ones that are relevant to cognitive science applications:

- Availability bias: Events that are more salient to the researcher are given higher probability, and events that are less salient are given lower probability.
- Adjustment and anchoring bias: The initial assessment of the probability of an event can influence one's subsequent judgements. An example is credible intervals: a researcher's estimate of the credible interval will tend to be influenced by their initial assessment.
- Overconfidence: When eliciting credible intervals from oneself, there is a tendency to specify too tight an interval.
- Hindsight bias: If one relies on the data to come up with a prior for the analysis of that very same data set, one's assessment is likely to be biased.

Although training can improve the natural tendency to be biased in these different ways, one must recognize that bias is inevitable when eliciting priors, either from oneself or from other experts; it follows that one should always define "a community of priors" (Kass and Greenhouse 1989): one should consider the effect of informed as well as skeptical or agnostic (uninformative) priors on the posterior distribution of interest. Incidentally, bias is not unique to Bayesian statistics; the same problems arise in frequentist data analysis. Even

in frequentist analyses, the researcher always interprets the data in the light of their prior beliefs; the data never really "speak for themselves." For example, the researcher might remove "outliers" based on a belief that certain values are implausible; or the researcher will choose a particular likelihood based on their belief about the underlying generative process. All these are subjective decisions made by the researcher, and can dramatically impact the outcome of the analyses.

The great advantage that Bayesian methods have is that they allow us to formally take a range of (competing) prior beliefs into account when interpreting the data. We illustrate this point in the present chapter.

E.1 Eliciting priors from oneself for a self-paced reading study: An example

In section 3.5, we have already encountered a sensitivity analysis; there, several priors were used to investigate how the posterior is affected. Here is another example of a sensitivity analysis; the problem here is how to elicit priors from oneself for a particular research problem.

E.1.1 An example: English relative clauses

We will work out priors from first principles for a commonly-used experiment design in psycholinguistics. As an example, consider English subject vs. object relative clause processing differences. Relative clauses are sentences like (1a) and (1b):

(1a) The *reporter* [who the photographer *sent* to the editor] was hoping for a good story. (ORC)

(1b) The *reporter* [who *sent* the photographer to the editor] was hoping for a good story. (SRC)

Sentence (1a) is an object relative clause (ORC): the noun *reporter* is modified by a relative clause (demarcated in square brackets), and the noun *reporter* is the object of the verb *sent*. Sentence (1b) is a subject relative clause (SRC): the noun *reporter* is modified by a relative clause (demarcated in square brackets), but this time the noun *reporter* is the subject of the verb *sent*. Many theories in sentence processing predict that the reading time at the verb *sent* will be shorter in English subject vs. object relatives; one explanation is that the dependency distance between *reporter* and *sent* is shorter in subject vs. object relatives (Grodner and Gibson 2005).

The experimental method we consider here is self-paced reading.[1] The self-paced reading method is commonly used in psycholinguistics as a cheaper and faster substitute to eyetracking during reading. The subject is seated in front of a computer screen and is initially shown a series of broken lines that mask words from a complete sentence. The subject then unmasks the first word (or phrase) by pressing the space bar. Upon pressing the space bar again, the second word/phrase is unmasked and the first word/phrase is masked again; see Figure E.1. The time in milliseconds that elapses between these two space-bar presses counts as the reading time for the first word/phrase. In this way, the reading time for each successive word/phrase in the sentence is recorded. Usually, at the end of each trial, the subject is also asked a yes/no question about the sentence. This is intended to ensure that the subject is adequately attending to the meaning of the sentence.

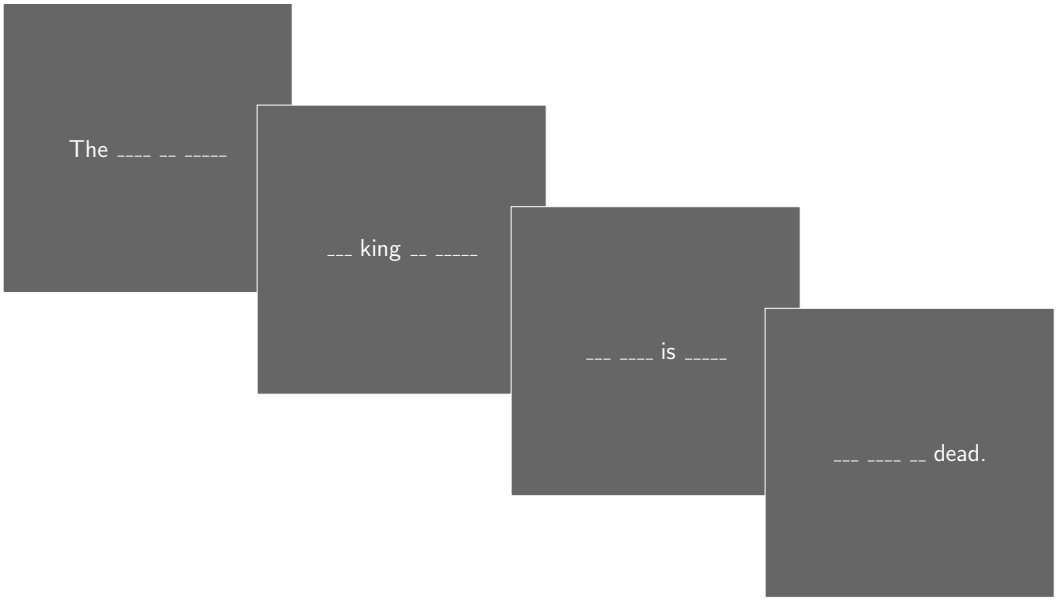

FIGURE E.1: A moving window self-paced reading task for the sentence "The king is dead." Words are unmasked one by one after each press of the space bar.

A classic example of self-paced reading data appeared in online exercise G.5.2. A hierarchical model that we could fit to such data would be the following. In chapter 5, we showed that for reading-time data, the log-normal likelihood is generally a better choice than a normal likelihood. In the present chapter, in order to make it easier for the reader to get started with thinking about priors, we use the normal likelihood instead of the log-normal. In real-life data

[1] This discussion reuses text from Vasishth et al. (2022).

analysis, the normal likelihood would be a very poor choice for reading-time data.

The model below has varying intercepts and varying slopes for subjects and for items, but assumes no correlation between the varying intercepts and slopes. The correlation is removed in order to compare the posteriors to the estimates from the corresponding frequentist `lme4` model. In the model shown below, we use "default" priors that the `brm` function assumes for all the parameters. We are only using default priors here as a starting point; in practice, we will **never** use default priors for a reported analysis. In the model output below, for brevity we will only display the summary of the posterior distribution for the slope parameter, which represents the difference between the two condition means.

```
data("df_gg05_rc")
df_gg05_rc <- df_gg05_rc %>%
  mutate(c_cond = if_else(condition == "objgap", 1 / 2, -1 / 2))
fit_gg05 <- brm(RT ~ c_cond + (c_cond || subj) + (c_cond || item),
                data = df_gg05_rc,
                control = list(adapt_delta = 0.99))
```

```
(default_b <- posterior_summary(fit_gg05,
                                variable = "b_c_cond"))
```

```
##          Estimate Est.Error Q2.5 Q97.5
## b_c_cond      103      36.6 31.2   176
```

The estimates from this model are remarkably similar to those from a frequentist linear mixed model (Bates, Mächler, et al. 2015):

```
fit_lmer <- lmer(RT ~ c_cond + (1 + c_cond || subj) +
                   (1 + c_cond || item), df_gg05_rc)
b <- summary(fit_lmer)$coefficients["c_cond", "Estimate"]
SE <- summary(fit_lmer)$coefficients["c_cond", "Std. Error"]
## estimate of the slope and
## lower and upper bounds of the 95% CI:
(lmer_b <- c(b, b - (2 * SE), b + (2 * SE)))
```

```
## [1] 102.3  29.9 174.7
```

The similarity between the estimates from the Bayesian and frequentist models is due to the fact that default priors, being relatively uninformative, don't

TABLE E.1: Estimates of the mean difference (with 95% confidence/credible intervals) between two conditions in a hierarchical model of English relative clause data from Grodner and Gibson, 2005, using (a) the frequentist hierarchical model, (b) a Bayesian model using default priors from the brm function, and (c) a Bayesian model with uniform priors.

model	mean	lower	upper
Frequentist	102	30	175
Default prior	103	31	176
Uniform	102	29	180

influence the posterior much. This leads to the likelihood dominating in determining the posteriors. In general, such uninformative priors on the parameters will show a similar lack of influence on the posterior (Spiegelhalter, Abrams, and Myles 2004). We can quickly establish this in the above example by using another uninformative prior:

```
unif_prior <- c(prior(uniform(-2000, 2000), class = Intercept,
                      lb = -2000, ub = 2000),
                prior(uniform(-2000, 2000), class = b,
                      lb = -2000, ub = 2000),
                prior(normal(0, 500), class = sd),
                prior(normal(0, 500), class = sigma))
fit_gg05_unif <- brm(RT ~ c_cond + (c_cond || subj) + (c_cond || item),
                     prior = unif_prior,
                     data = df_gg05_rc)
```

```
(uniform_b <- posterior_summary(fit_gg05_unif,
                                variable = c("b_c_cond")))
```

```
##          Estimate Est.Error Q2.5 Q97.5
## b_c_cond      102      38.4 29.3   180
```

As shown in Table E.1, the means of the posteriors from this versus the other two model estimates shown above all look very similar.

It is tempting for the newcomer to Bayesian statistics to conclude from Table E.1 that default priors used in `brms`, or uniform priors, are good enough for fitting models. This conclusion would in general be incorrect. There are many reasons why a sensitivity analysis–which includes regularizing, relatively informative priors–is necessary in Bayesian modeling. First, relatively informative,

regularizing priors must be considered in many cases to avoid convergence problems (an example is finite mixture models, presented in chapter 17). In fact, in many cases the frequentist model fit in `lme4` will return estimates–such as ±1 correlation estimates between varying intercepts and varying slopes–that are actually represent convergence failures (Bates, Kliegl, et al. 2015; Matuschek et al. 2017). In Bayesian models, unless we use regularizing priors that are at least mildly informative, we will generally face similar convergence problems. Second, when computing Bayes factors, a sensitivity analysis using increasingly informative priors is vital; see chapter 13 for extensive discussion of this point. Third, one of the greatest advantages of Bayesian models is that one can formally take into account conflicting or competing prior beliefs in the model, by eliciting informative priors from competing experts. Although such a use of informative priors is still rare in cognitive science, it can be of great value when trying to interpret a statistical analysis.

Given the importance of regularizing and informative priors, we consider next some informative priors that we could use in the given model. We unpack the process by which we could work these priors out from existing information in the literature.

Initially, when trying to work out some alternative priors for some parameters of interest, we might think that we know absolutely nothing about the seven parameters in this model. But, as in Fermi problems, we actually know more than we realize.

Let's think about the parameters in the relative clause example one by one. For ease of exposition, we begin by writing out the model in mathematical form. n is the row id in the data-frame. The variable `c_cond` is a sum-coded (± 0.5) predictor.

$$RT_n \sim Normal(\alpha + u_{subj[n],1} + w_{item[n],1} + c\_cond_n \cdot (\beta + u_{subj[n],2} + w_{item[n],2}), \sigma) \quad \text{(E.1)}$$

where

$$\begin{aligned} u_1 &\sim Normal(0, \tau_{u_1}) \\ u_2 &\sim Normal(0, \tau_{u_2}) \\ w_1 &\sim Normal(0, \tau_{w_1}) \\ w_2 &\sim Normal(0, \tau_{w_2}) \end{aligned} \quad \text{(E.2)}$$

The parameters that we need to define priors for are the following: $\alpha, \beta, \tau_{u_1}, \tau_{u_2}, \tau_{w_1}, \tau_{w_2}, \sigma$.

E.1.2 Eliciting a prior for the intercept

We will proceed from first principles. Let's begin with the intercept, α; under the sum-contrast coding used here, it represents the grand mean reading time in the data set.

Ask yourself: What is the absolute minimum possible reading time? The answer is 0 ms; reading time cannot be negative. You have already eliminated half the real-number line as impossible values! Thus, one cannot really say that one knows *nothing* about the plausible values of mean reading times. Having eliminated half the real-number line, now ask yourself: what is a reasonable upper bound on reading time for an English ditransitive verb? Even after taking into account variations in word length and frequency, one minute (60 seconds) seems like too long; even 30 seconds seems unreasonably long to spend on a single word. As a first attempt at an approximation, somewhere between 2500 and 3000 ms might constitute a reasonable upper bound, with 3000 ms being less likely than 2500 ms.

Now consider what an approximate average reading time for a verb might be. One can arrive at such a ballpark number by asking oneself how fast one can read an abstract that has, say, 500 words in it. Suppose that we estimate that we can read 500 words in 120 seconds (two minutes). Then, 120/500 = 0.24 seconds is the time we would spend per word on average; this is 240 ms per word. Maybe two minutes for 500 words was too optimistic? Let's adjust the mean to 300 ms, instead of 240 ms. Such intuition-based judgments can be a valuable starting point for an analysis, as Fermi showed repeatedly in his work (Von Baeyer 1988). If one is uncomfortable consulting one's intuition about average reading times, or even as a sanity check to independently validate one's own intuitions, one can look up a review article on reading that gives empirical estimates (e.g., Rayner 1998).

One could express the above guesses as a normal distribution truncated at 0 ms on the ms scale, with mean 300 ms and standard deviation 1000 ms. An essential step in such an estimation procedure is to plot one's assumed prior distribution graphically to see if it seems reasonable: Figure E.2 shows a graphical summary of this prior.

Once we plot the prior, one might conclude that the prior distribution is a bit too widely spread out to represent mean reading time per word. But for estimating the posterior distribution, it will rarely be harmful to allow a broader range of values than we strictly consider plausible (the situation is different when it comes to Bayes factors analyses, as we will see later—there, widely spread out priors for a parameter of interest can have a dramatic impact on the Bayes factor test for whether that parameter is zero or not).

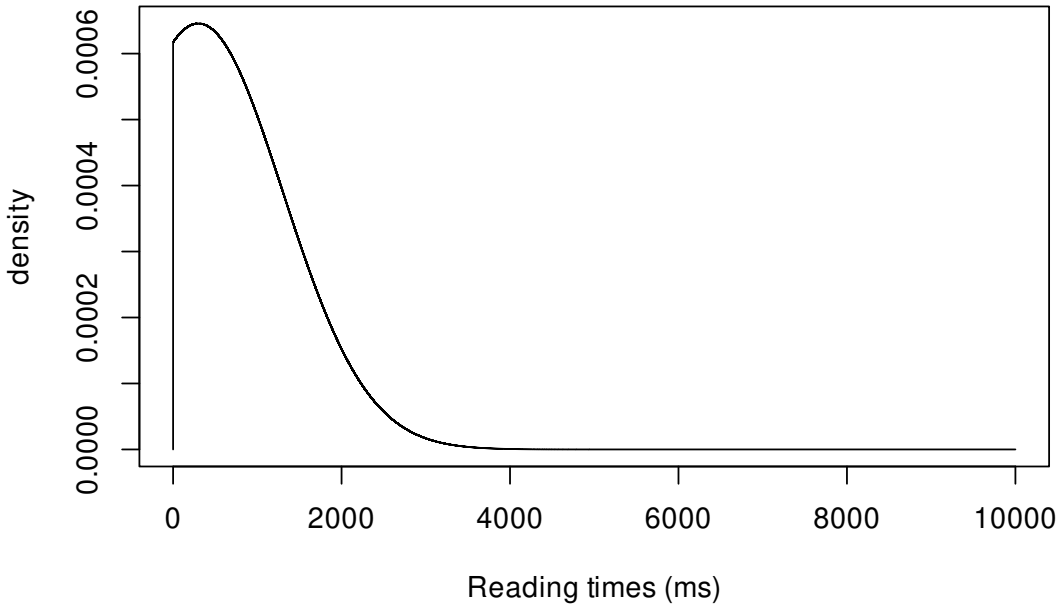

FIGURE E.2: A truncated normal distribution representing a prior distribution on mean reading times.

Another way to obtain a better feel for what plausible distributions of word reading times might be to just plot some existing data from published work. Figure E.3 shows the distribution of mean reading times from ten published studies.

Although our truncated normal distribution, $Normal_+(300, 1000)$, seems like a pretty wild guess, it actually is not terribly unreasonable given what we observe in these ten published self-paced reading studies. As shown in Figure E.3, the distribution of mean reading times in these different self-paced reading studies from different languages (English, Persian, Dutch, Hindi, German, Spanish) fall within the prior distribution. The means range from a minimum value of 464 ms and a maximum value of 751 ms. These values easily lie within the 95% credible interval for a $Normal_+(300, 1000)$: [40, 2458] ms. These 10 studies are not about relative clauses; but that doesn't matter, because we are just trying to come up with a prior distribution on average reading times for a word. We just want an approximate idea of the range of plausible mean reading times.

The above prior specification for the intercept can (and must!) be evaluated in the context of the model using prior predictive checks. We have already encountered prior predictive checks in previous chapters; we revisit them in detail in the online chapter F. In the above data set on English relative clauses, one could check what the prior on the intercept implies in terms of the data

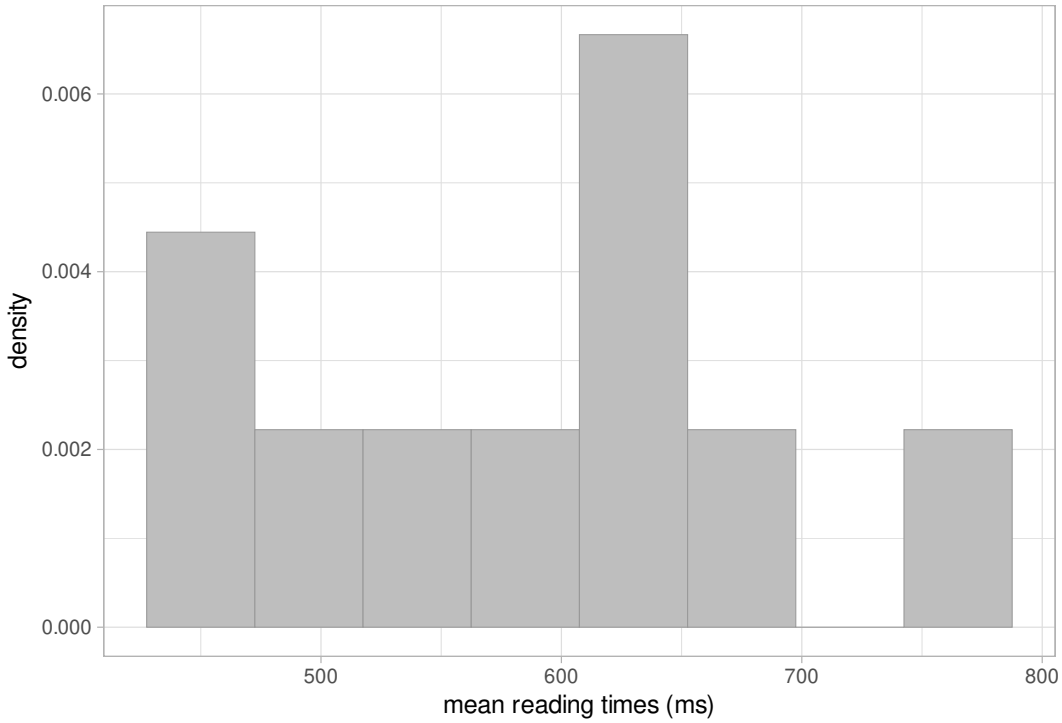

FIGURE E.3: The distribution of mean reading times from ten self-paced reading studies.

generated by the model (see chapter 5 for examples). As stressed repeatedly throughout this book, sensitivity analysis is an integral component of Bayesian methodology. A sensitivity analysis should be used to work out what the impact is of a range of priors on the posterior distribution.

E.1.3 Eliciting a prior for the slope

Having come up with some potential priors for the intercept, consider next the prior specification for the effect of relative clause type on reading time; this is the slope β in the model above. Recall that `c_cond` is ± 0.5 sum coded.

Theory suggests (see Grodner and Gibson 2005 for a review) that subject relatives in English should be easier to process than object relatives, at the relative clause verb. This means that a priori, we expect the difference between object and subject relatives to be positive in sign. What would be a reasonable mean for this effect? We can look at previous research to obtain some ballpark estimates.

For example, Just and Carpenter (1992) carried out a self-paced reading study on English subject and object relatives, and their Figure 2 (p. 130) shows that the difference between the two relative clause types at the relative clause verb

ranges from about 10 ms to 100 ms (depending on working memory capacity differences in different groups of subjects). This is already a good starting point, but we can look at some other published data to gain more confidence about the approximate difference between the conditions.

For example, Reali and Christiansen (2007) investigated subject and object relatives in four self-paced reading studies; in their design, the noun phrase inside the relative clause was always a pronoun, and they carried out analyses on the verb plus pronoun, not just the verb as in Grodner and Gibson (2005). We can still use the estimates from this study, because including a pronoun like "I", "you", or "they" in a verb region is not going to increase reading times dramatically. The hypothesis in Reali and Christiansen (2007) was that because object relatives containing a pronoun occur more frequently in corpora than subject relatives with a pronoun, the relative clause verb should be processed faster in object relatives than subject relatives (this is the opposite to the prediction for the reading times at the relative clause verb discussed in Grodner and Gibson 2005). The authors report comparisons for the pronoun and relative clause verb taken together (i.e., pronoun+verb in object relatives and verb+pronoun in subject relatives). In experiment 1, they report a -57 ms difference between object and subject relatives, with a 95% confidence interval ranging from -104 to -10 ms. In a second experiment, they report a difference of -53.5 ms with a 95% confidence interval ranging from -79 to -28 ms; in a third experiment, the difference was -32 ms $[-48, -16]$; and in a fourth experiment, -43 ms $[-84, -2]$. This range of values gives us a good ballpark estimate of the magnitude of the effect.

Yet another study involved English relative clauses is by Fedorenko, Gibson, and Rohde (2006). In this self-paced reading study, Fedorenko and colleagues compared reading times within the entire relative clause phrase (the relative pronoun and the noun+verb sequence inside the relative clause). Their data show that object relatives are harder to process than subject relatives; the difference in means is 460 ms, with a confidence interval $[299, 621]$ ms. This difference is much larger than in the studies mentioned above, but this is because of the long region of interest considered—it is well-known that the longer the reading/response time, the larger the standard deviation and therefore the larger the potential difference between means (Wagenmakers and Brown 2007).

One can also look at adjacent, related phenomena in sentence processing to get a feel for what the relative clause processing time difference should be. Research on similarity-based interference is closely related to relative clause processing differences: in both types of phenomenon, the assumption is that

intervening nouns can increase processing difficulty. So let's look at some reading studies on similarity-based interference.

In a recent study (Jäger, Engelmann, and Vasishth 2017), we investigated the estimates from about 80 reading studies on interference. Interference here refers to the difficulty experienced by the comprehender during sentence comprehension when they need to retrieve a particular word from working memory, but other words with similar features hinder retrieval. The meta-analysis in Jäger, Engelmann, and Vasishth (2017) suggests that the effect sizes for interference effects range from at most -50 to 50 ms, depending on the phenomenon (some kinds of interference cause speed-ups, others cause slow-downs; see the discussion in Engelmann, Jäger, and Vasishth 2020).

Given that the Grodner and Gibson (2005) design can be seen as falling within the broader class of interference effects (Lewis and Vasishth 2005; Vasishth et al. 2019; Vasishth and Engelmann 2022), it is reasonable to choose informative priors that reflect this observed range of interference effects in the literature.

The above discussion gives us some empirical basis for assuming that the object minus subject relative clause difference in the Grodner and Gibson (2005) study on English could range from 10 to at most 100 ms or so. Although we expect the effect to be positive, perhaps we don't want to pre-judge this before we see the data. For this reason, we could decide on a $Normal(0, 50)$ prior on the slope parameter in the model. This prior, which implies that we are 95% certain that the range of values lies between -100 and $+100$ ms. This prior is specifically for the millisecond scale, and specifically for the case where the critical region is one word (the relative clause verb in English).

In this particular example, it makes sense to assume that large effects like 100 ms are unlikely; this is so even if we do occasionally see estimates that are even higher than 100 ms in published data. For example, in Gordon, Hendrick, and Johnson (2001), their experiments 1-4 have very large OR-SR differences at the relative clause verb: 450 ms, 250 ms, 500 ms, and 200 ms, respectively, with an approximate SE of 50 ms. The number of subjects in the four experiments were 44, 48, 48, and 68, respectively. Given the other estimates mentioned above, we would be unwilling to take such large effects seriously because a major reason for observing overly large estimates in a one-word region of interest would be publication bias coupled with Type M error (Gelman and Carlin 2014). Published studies in psycholinguistics are often underpowered, which leads to exaggerated estimates being published (Type M error). Because big-news effects are encouraged in major journals, overestimates tend to get published

preferentially.[2] In recent work (Vasishth et al. 2022), we have shown that, if we repeatedly simulate data assuming a typical subject sample size of 42 and an effect size of approximately 50 ms, the probability of obtaining a Bayes factor (see chapter 13) larger than 10 in favor of the effect (i.e., a Bayes factor that indicates overwhelming evidence for the effect) will be relatively low at 64%. In that particular simulation study, we found that some 210 subjects would be needed to have an 80% probability of obtaining a Bayes factor in favor of the effect that is larger than 10.

Of course, if our experiment is designed so that the critical region constitutes several words, as in the Fedorenko, Gibson, and Rohde (2006) study, then one would have to choose a prior with a larger mean and standard deviation.

Box E.1. ***The scale of the parameter must be taken into account when eliciting a prior***

A related, important issue to consider when defining priors is the scale in which the parameter is defined. For example, if we were analyzing the Grodner and Gibson (2005) experiment using the log-normal likelihood, then the intercept and slope are on the log millisecond scale. A uniform prior on the intercept and slope parameter imply rather strange priors on the millisecond scale. For example, suppose we assume that the intercept *on the log ms scale* has priors $Normal_+(300, 100)$ and the slope has a prior $Normal(0, 50)$. In the millisecond scale, the priors on the intercept and slope imply a very broad range of reading time differences between the two conditions, ranging from a very large negative value to a very large positive value, which obviously makes little sense:

```
intercept <- rtnorm(100000, mean = 300, sd = 100, a = 0)
slope <- rnorm(100000, mean = 0, sd = 50)
effect <- exp(intercept + slope / 2) -
  exp(intercept - slope / 2)
quantile(effect, prob = c(0.025, 0.975))
```

```
##       2.5%      97.5%
## -2.38e+211  9.69e+210
```

In this connection, it may be useful to revisit the discussion in section

[2] See Vasishth et al. (2013);Jäger et al. (2020);Nicenboim et al. (2018);Vasishth et al. (2018) for detailed discussion of this point in the context of psycholinguistics.

4.3.2, where we discussed the effect of prior specification on the log-odds scale and what that implies on the probability scale.

Box E.2. ***Cromwell's rule***

A frequently asked question from newcomers to Bayes is: what if I define a too restricted prior? Wouldn't that bias the posterior distribution? This concern is also raised quite often by critics of Bayesian methods. The key point here is that a good Bayesian analysis always involves a sensitivity analysis, and also includes prior and posterior predictive checks under different priors. One should reject the priors that make no sense in the particular research problem we are working on, or which unreasonably bias the posterior. As one gains experience with Bayesian modeling, these concerns will recede as we come to understand how useful and important priors are for interpreting the data. The online chapter F elaborates on developing a sensible workflow for understanding and interpreting the results of a Bayesian analysis.

As an extreme example of an overly specific prior, if one were to define a $Normal(0, 10)$ prior for the α and/or β parameters on the millisecond scale for the Grodner and Gibson (2005) example above; that would definitely bias the posterior for the parameters. Let's check this. Try running this code (the output of the code is suppressed here to conserve space). In this model, the correlation between the varying intercepts and varying slopes for subjects and for items are not included; this is only done in order to keep the model simple.

```
restrictive_priors <-
  c(prior(normal(0, 10), class = Intercept),
    prior(normal(0, 10), class = b),
    prior(normal(0, 500), class = sd),
    prior(normal(0, 500), class = sigma))

fit_restrictive <- brm(RT ~ c_cond + (c_cond || subj) +
                         (c_cond || item),
                       prior = restrictive_priors,
                     # Increase the iterations to avoid warnings
                       iter = 4000,
                       df_gg05_rc)
```

```
summary(fit_restrictive)
```

If you run the above code, you will see that the overly specific (and extremely unreasonable) priors on the intercept and slope will dominate in determining the posterior; such priors obviously make no sense. If there is ever any doubt about the implications of a prior, prior and posterior predictive checks should be used to investigate the implications.

Here, an important Bayesian principle is Cromwell's rule (Lindley 1991; Jackman 2009): we should generally allow for some uncertainty in our priors. A prior like $Normal(0, 10)$ or $Normal_{+}(0, 10)$ on the millisecond scale is clearly overly restrictive given what we've established about plausible values of the relative clause effect from existing data. A more reasonable but still quite tight prior would be $Normal(0, 50)$. In the spirit of Cromwell's rule, just to be conservative, we can consider allowing (in a sensitivity analysis) larger possible effect sizes by adopting a prior such as $Normal(0, 75)$, and we allow the effect to be negative, even if theory suggests otherwise.

Although there are no fixed rules for deciding on a prior, a sensitivity analysis will quickly establish whether the prior or priors chosen are biasing the posterior. One critical thing to remember related to Cromwell's rule is that if we categorically rule out a range of values a priori for a parameter by giving that range a probability of 0, the posterior will also never include that range of values, no matter what the data show. For example, in the Reali and Christiansen (2007) experiments, if we had used a truncated prior like $Normal_{+}(0, 50)$, the posterior can never show the observed negative sign on the effects as reported in the paper. As a general rule, therefore, one should allow the effect to vary in both directions, positive and negative. Sometimes unidirectional priors are justified; in those cases, it is of course legitimate to use them. An example is the prior on standard deviations (which cannot be negative). Another example is using directional priors in when carrying out Bayes factors analyses (e.g., Ly et al. 2019).

E.1.4 Eliciting priors for the variance components

Having defined the priors for the intercept and the slope, we are left with prior specifications for the variance component parameters. At least in psycholinguistics, the residual standard deviation is usually the largest source of variance; the by-subject intercepts' standard deviation is usually the next-largest value, and if experimental items are designed to have minimal variance,

then these are usually the smallest components. Here again, we can look at some previous data to get a sense of what the priors should look like.

For example, we could use the estimates for the variance components from existing studies. Figure E.4 shows the empirical distributions from 10 published studies. There are four classes of variance component: the subject and item intercept standard deviations, the standard deviations of slopes, and the standard deviations of the residuals. In each case, we can compute the estimated means and standard deviations of each type of variance component, and then use these to define normal distributions truncated at 0. The empirically estimated distributions of the variance components are shown in Figure E.4. The estimated means and standard deviations of each type of variance component are as follows:

- Subject intercept SDs: estimated mean: 165, estimated standard deviation (sd): 55.
- Item intercept SDs: mean: 49, sd: 52.
- Slope SDs: mean 39, sd: 58.
- Residual SDs: mean: 392, sd: 140.

The largest standard deviations are those estimated for the by-subject intercept adjustment and the residual term, so these are the ones we will focus on. In order to be conservative, for the standard deviations of the group factors (subject and item), we take the larger values (the by-subject intercept's standard deviations) as a guide when designing priors for all the standard deviations of the by-subject and by-item adjustments. For the residual standard deviation, we use the above estimates.

We can now use the equations (A.7) and (A.8) shown in online section A.2 to work out the means and standard deviations of a corresponding truncated normal distribution. As an example, we could assume a prior distribution truncated at 0 from below, and at 1000 ms from above. That is, $a = 0$, and $b = 1000$.

We can write a function that takes the estimated means and standard deviations, and returns the mean and standard deviation of the corresponding truncated distribution (see online A.2). The function `compute_meansd_parent()` provided in `bcogsci` accomplishes this.

The largest variance component among the group-level effects (that is, all variance components other than the residual standard deviation) is the by-subjects intercept. One can compute the mean and standard deviation of of the truncated distribution that would generate the observed mean and standard deviation of the item-level estimates:

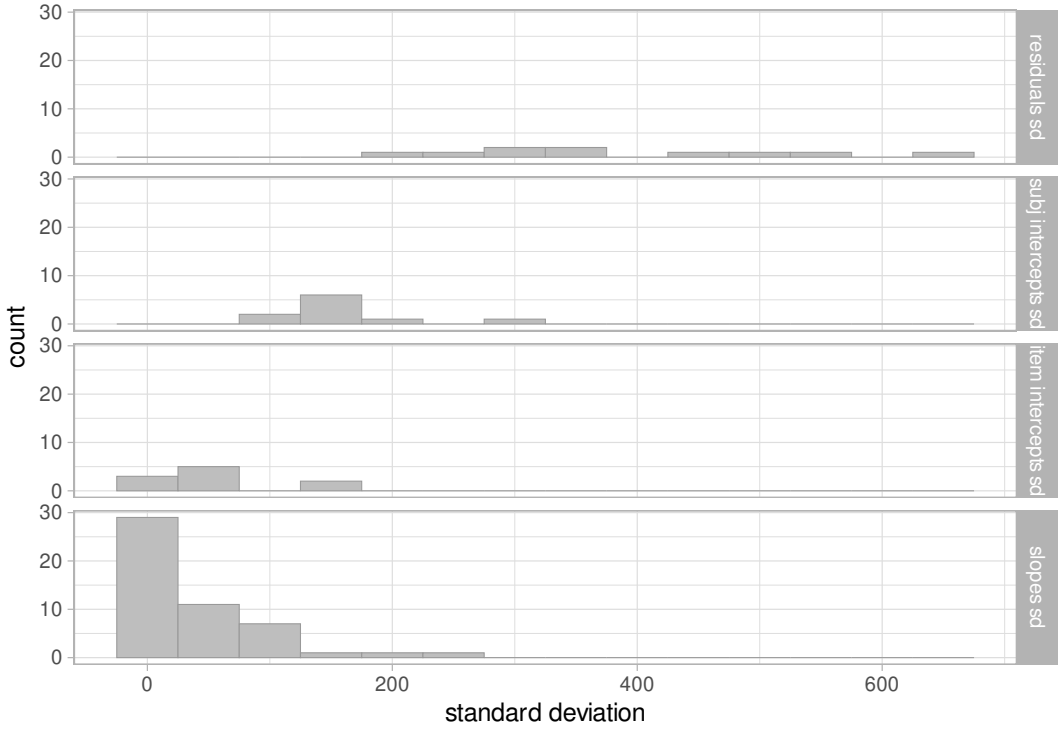

FIGURE E.4: Histograms of empirical distributions of the different variance components from ten publishes studies. The y-axis shows counts rather than density in order to make it clear that we are working with only a few data sets.

```
# Subject intercept SDs:
compute_meansd_parent(mean_trunc = 165,
                      sd_trunc = 55,
                      a = 0,
                      b = 1000)
```

```
## $location
## [1] 165
##
## $scale
## [1] 55.4
```

The corresponding truncated distribution is shown in Figure E.5.

The prior shown in Figure E.5 looks a bit too restrictive; it could well happen that in a future study the by-subject intercept standard deviation is closer

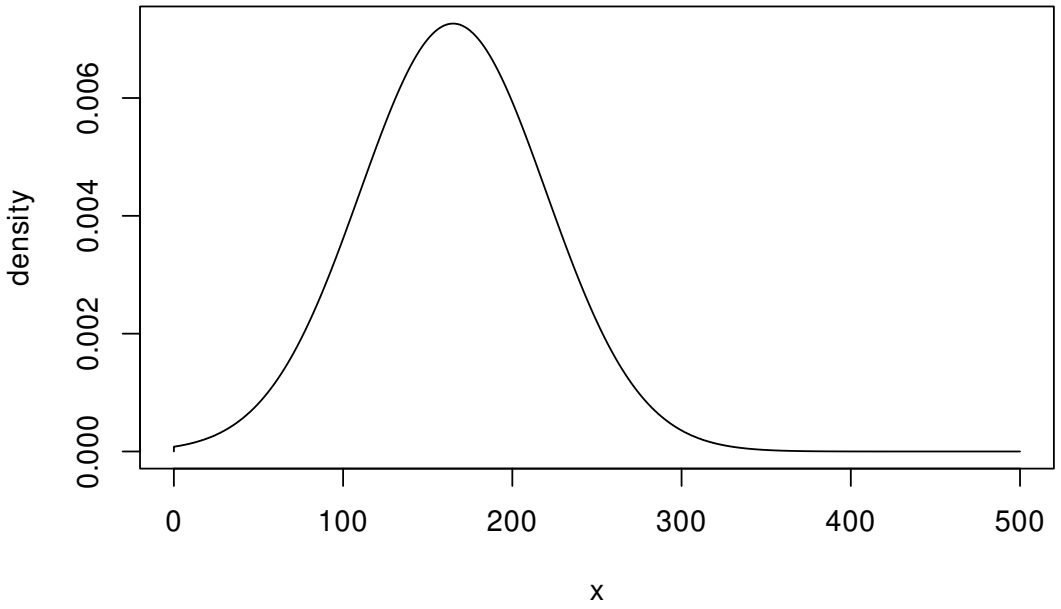

FIGURE E.5: A truncated normal distribution (with location 165 and scale 55) representing an empirically derived prior distribution for the parameter for the by-subjects intercept adjustment in a hierarchical model.

to 500 ms. Taking Cromwell's rule into account, one could widen the scale parameter of the truncated normal to, say 200. The result is Figure E.6.

Figure E.6 does not look too unreasonable as an informative prior for this variance component. This prior will also serve us well for all the other group-level effects (the random intercept for items, and the random slopes for subject and item), which will have smaller values.

Finally, the prior for the residual standard deviation is going to have to allow a broader range of wider values:

```
compute_meansd_parent(mean_trunc = 392,
                      sd_trunc = 140,
                      a = 0,
                      b = 1000)
```

```
## $location
## [1] 391
##
## $scale
## [1] 142
```

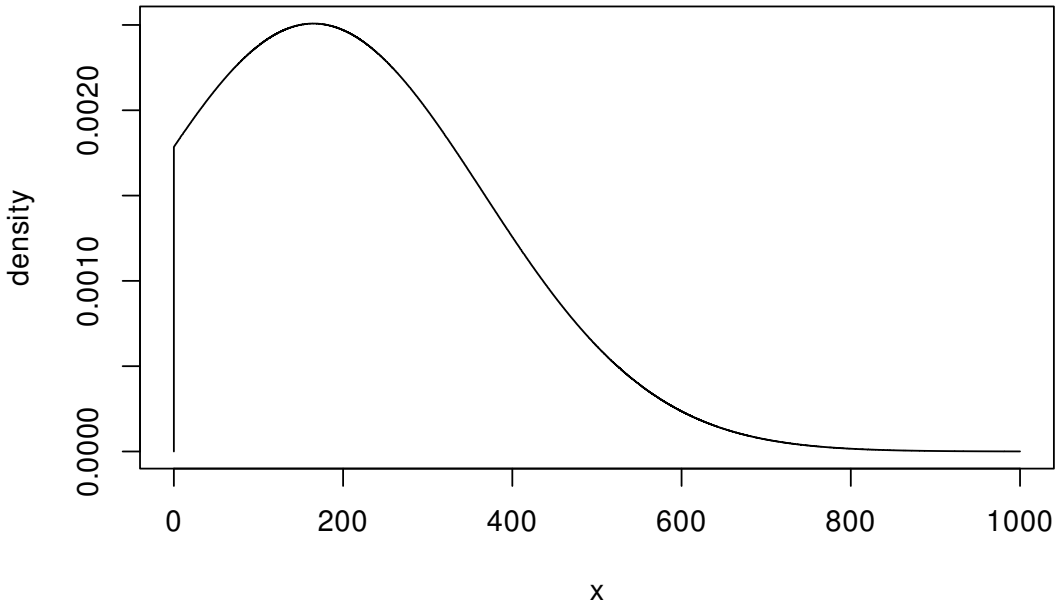

FIGURE E.6: A truncated normal distribution (with location 165 and scale 200) representing an empirically derived prior distribution for the parameter for the by-subjects intercept adjustment in a hierarchical model taking Cromwell's rule into account.

Figure E.7 shows a plausible informative prior derived from the empirical estimates.

We stress again that Cromwell's rule should generally be kept in mind—it's usually better to have a little bit more uncertainty than warranted than too tight a prior. An overly tight prior will ensure that the posterior is entirely driven by the prior. Again, prior predictive checks should be an integral part of the process of establishing a sensible set of priors for the variance components. This point about prior predictive checks are elaborated on with examples in the online chapter F.

We now apply the relatively informative priors we came up with above to analyze the Grodner and Gibson (2005) data. Applying Cromwell's rule, we allow for a bit more uncertainty than our existing empirical data suggest.

Specifically, we could choose the following informative priors for the Grodner and Gibson (2005) data:

- The intercept: $\alpha \sim Normal(500, 100)$
- The slope: $\beta \sim Normal(50, 50)$
- Standard deviation of the adjustments to the intercept : $\tau_{u_1}, \tau_{w_1} \sim Normal_+(165, 200)$

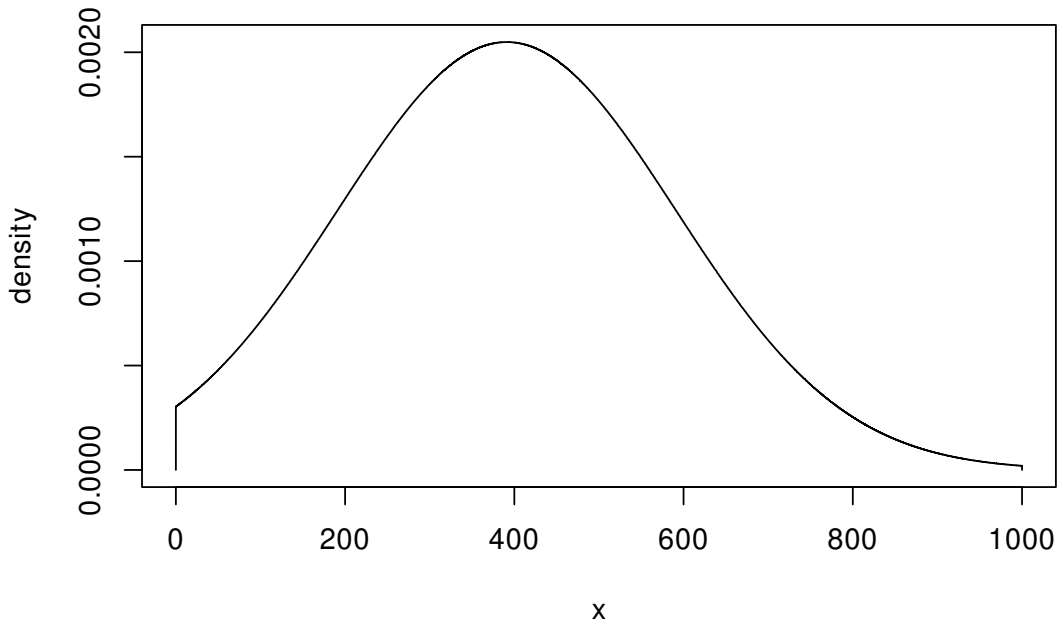

FIGURE E.7: A truncated normal distribution representing an empirically derived prior distribution for the parameter for the residual standard deviation in a hierarchical model.

- Standard deviation of the adjustments to the slopes : $\tau_{u_2}, \tau_{w_2} \sim Normal_+(39, 58)$
- The residual standard deviation: $\sigma \sim Normal_+(391, 200)$

The first step is to check whether the prior predictive distribution makes sense. Figure E.8 shows that the prior predictive distributions are not too implausible, although they could be improved further. One big problem is the normal distribution assumed in the model; a log-normal distribution captures the shape of the distribution of the Grodner and Gibson (2005) data better than a normal distribution. The discrepancy between the Grodner and Gibson (2005) data and our prior predictive distribution implies that we might be using the wrong likelihood. Another problem is that the reading times in the prior predictive distribution can be negative—this is also a consequence of our using the wrong likelihood. As an exercise, fit a model with a log-normal likelihood and informative priors based on previous data. When using a log-normal likelihood, the prior for the slope parameter obviously has to be on the log scale. Therefore, we will need to define an informative prior on the log scale for the slope parameter. For example, consider the following prior on the slope: $Normal(0.12, 0.04)$. Here is how to interpret this on the millisecond scale: Assuming a mean reading time of 6 log ms, this prior roughly corresponds to an effect size on the millisecond scale that has a 95% credible interval ranging from 16 ms to 81

ms. Review section 3.7.2 if you have forgotten how this transformation was done.

For now, because our running example uses a normal likelihood on reading times in milliseconds, we can retain these priors.

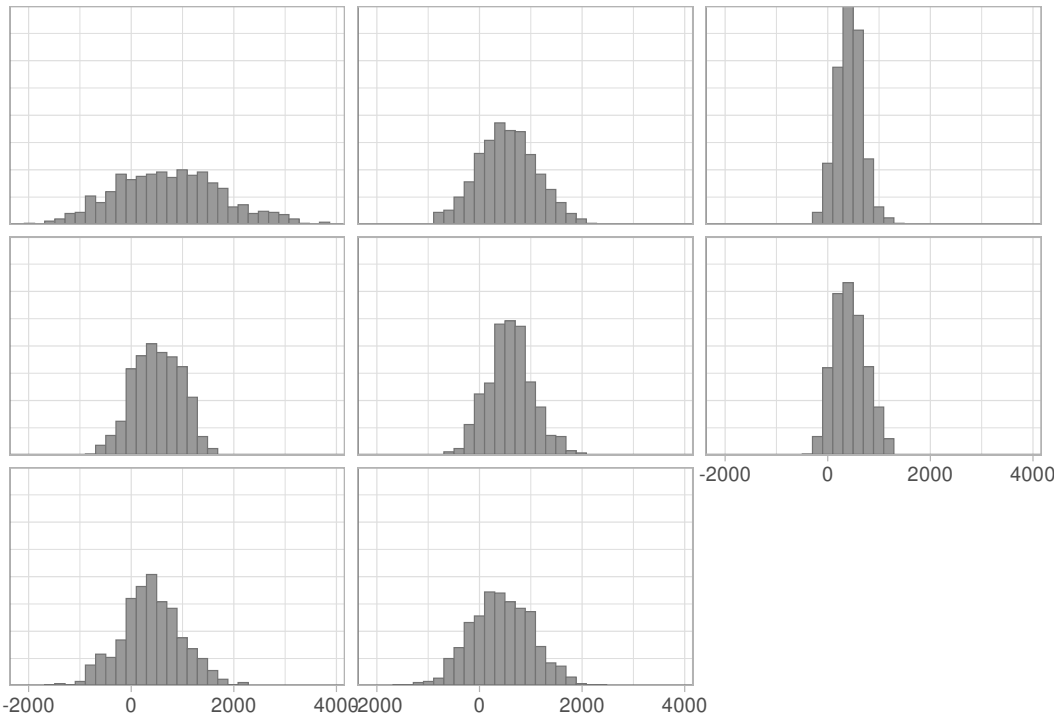

FIGURE E.8: Prior predictive distributions from the model (using a normal likelihood) to be used for the Grodner and Gibson data analysis. The panels show eight prior predictive distributions.

The sensitivity analysis could then be displayed, showing the posteriors under different prior settings. Figures E.9 and E.10 show the posteriors under two distinct sets of priors.

What can one do if one doesn't know absolutely anything about one's research problem? An example is the power posing data that we encountered in Chapter 4, in an exercise in section G.4. Here, we investigated the change in testosterone levels after the subject was either asked to adopt a high power pose or a low power pose (a between-subjects design). Not being experts in this domain, we may find ourselves stumped for priors. In such a situation, it could be defensible to use uninformative priors like $Cauchy(0, 2.5)$, at least initially. However, as discussed in a later chapter, if one is committed to doing a Bayes factor analysis, then we are obliged to think carefully about plausible a priori values of the effect. This would require consulting one or more experts

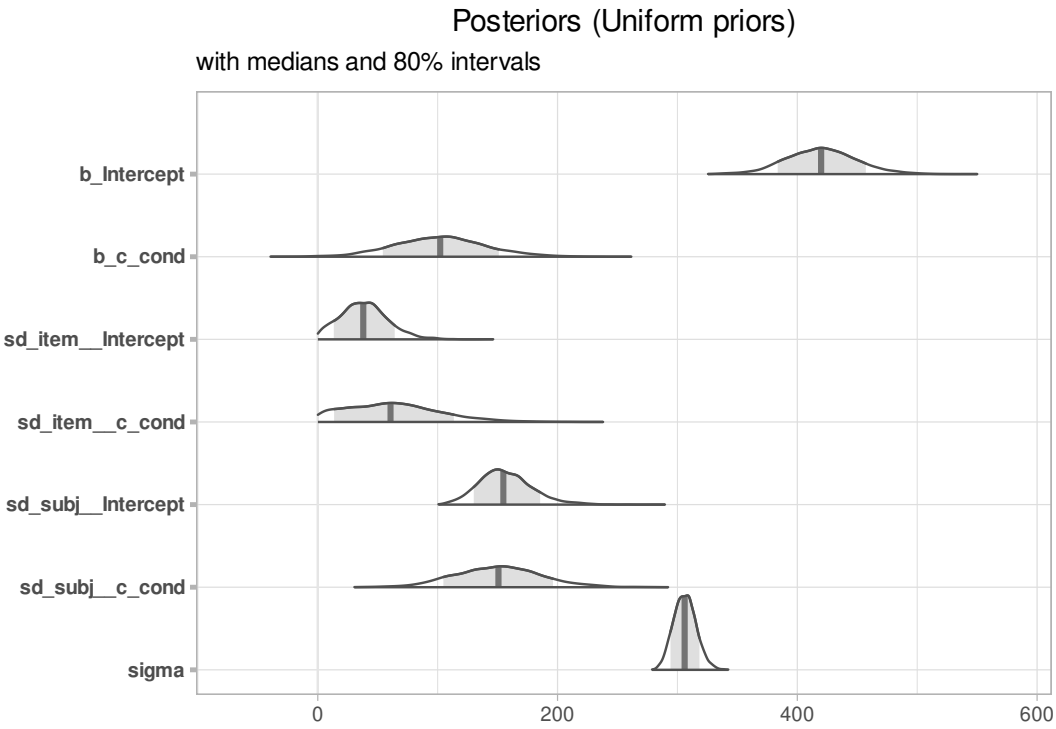

FIGURE E.9: Posterior distributions of parameters for the English relative clause data, using uniform priors ($Uniform(0, 2000)$) on the intercept and slope.

or reading the literature on the topic to obtain ballpark estimates. An exercise at the end of this chapter will elaborate on this idea. We turn next to the topic of eliciting priors from experts.

E.2 Eliciting priors from experts

It can happen that one is working on a research problem where either our own prior knowledge is lacking, or we need to incorporate a range of competing prior beliefs into the analysis. In such situations, it becomes important to elicit priors from experts other than oneself. Although informal elicitation can be a perfectly legitimate approach, there does exist a well-developed methodology for systematically eliciting priors in Bayesian statistics (O'Hagan et al. 2006).

The particular method developed by O'Hagan and colleagues comes with an R package called SHELF, which stands for the Sheffield Elicitation Framework; the method was developed by statisticians at the University of Sheffield, UK. At the time of writing this, SHELF is available from http://www.tonyohagan.co.uk/shelf/. This framework comes with a detailed set of instructions and a

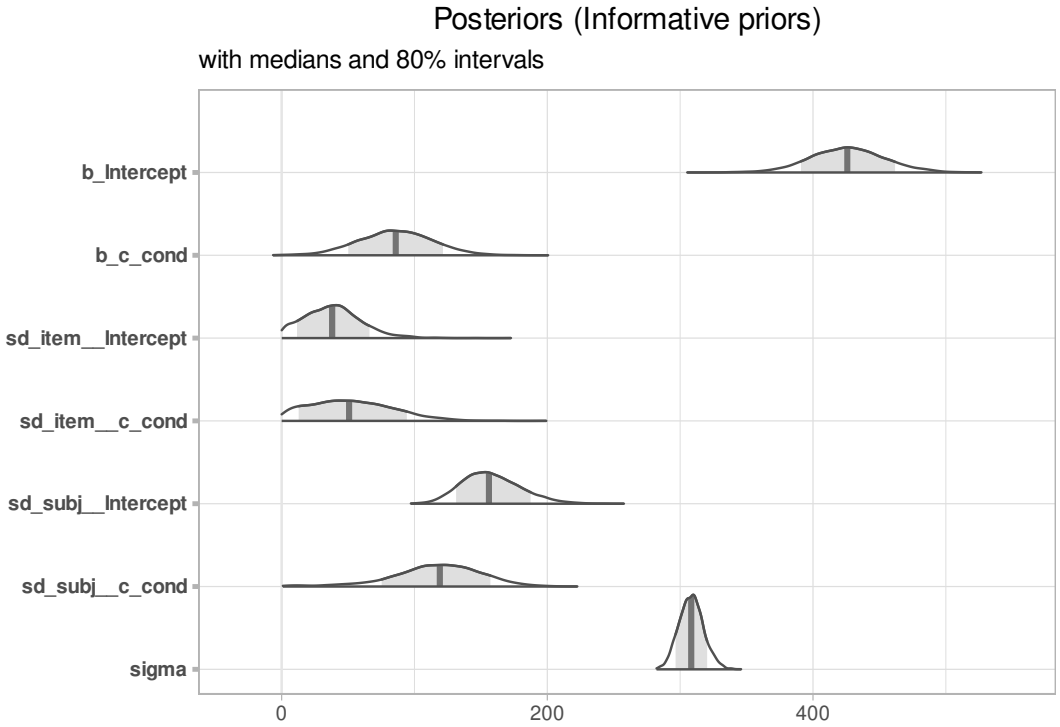

FIGURE E.10: Posterior distributions of parameters for the English relative clause data, using relatively informative priors on the intercept and slope.

fixed procedure for eliciting distributions. It also provides detailed guidance on documenting the elicitation process, thereby allowing a full record of the elicitation process to be created. Creating such a record is important because the elicitation procedure needs to be transparent to a third party reading the final report on the data analysis.

The SHELF procedure works as follows. There is a facilitator and an expert (or a group of experts; we will consider the single expert case here, but one can easily extend the approach to multiple experts).

- A pre-elicitation form is filled out by the facilitator in consultation with the expert. This form sets the stage for the elicitation exercise and records some background information, such as the nature of the expertise of the assessor.
- Then, an elicitation method is chosen. Simple methods are the most effective. One good approach is the quartile method. The expert first decides on a lower and upper limit of possible values for the quantity to be estimated. Because the lower and upper bounds are elicited before the median, this minimizes the effects of the "anchoring and adjustment heuristic" (O'Hagan et al. 2006), whereby experts tend to anchor their subsequent estimates of quartiles based on their first judgement of the median. Following this, a

median value is decided on, and lower and upper quartiles are elicited. The SHELF package has functions to display these quartiles graphically, allowing the expert to adjust them at this stage if necessary. It is important for the expert to confirm that, in their judgement, the four partitioned regions that result have equal probability.

- The elicited distribution is then displayed as a density plot (several choices of probability density functions are available, but we will usually use the normal or the truncated normal in this chapter); this graphical summary serves to give feedback to the expert. The parameters of the distribution are also displayed. Once the expert agrees to the final density, the parameters can be considered the expert's judgement regarding the prior distribution of the bias. One can consult multiple experts and either combine their judgements into one prior, or consider each expert's prior separately in a sensitivity analysis.

When eliciting priors from more than one expert, one can elicit the priors separately and then use the priors separately in a sensitivity analysis. This approach takes each individual expert's opinion in interpreting the data and can be a valuable sensitivity analysis (for an example from psycholinguistics, see the discussion surrounding Table 2.2 on p. 47 in Vasishth and Engelmann 2022). Alternatively, one can pool the priors together (see Spiegelhalter, Abrams, and Myles 2004 for discussion) and create a single consensus prior; this would amount to an average of the differing opinions about prior distributions. A third approach is to elicit a consensus prior by bringing all the experts together and eliciting a prior from the group in a single setting. Of course, these approaches are not mutually exclusive. One of the hallmark properties of Bayesian analysis is that the posterior distribution of the parameter of interest can be investigated in light of differing prior beliefs and the data (and of course the model). Box E.3 illustrates a simple elicitation procedure involving two experts; the example is adapted from the SHELF package's vignette.

Box E.3. ***Example: prior elicitiation using SHELF***

An example of prior elicitation using SHELF is shown below. This example is adapted from the SHELF vignette.

Suppose that two experts are consulted separately. The question asked of the experts is what they think that a probability parameter X has as plausible values. The parameter X can be seen as a percentage; so, it ranges from 0 to 100.

Step 1: Elicit quartiles and median from each expert.

- Expert A states that $P(X < 30) = 0.25, P(X < 40) = 0.5, P(X < 50) = 0.75$.
- Expert B states that $P(X < 20) = 0.25, P(X < 25) = 0.5, P(X < 35) = 0.75$.

Step 2: Fit the implied distributions for each expert's judgements and plot the distributions, along with a pooled distribution (the linear pool in the figure) using the `plotfit()` function from the library `SHELF`.

```
elicited <- matrix(c(30, 20, 0.25,
                     40, 25, 0.5,
                     50, 35, 0.75),
                   nrow = 3, ncol = 3, byrow = TRUE)
dist_2expr <- fitdist(vals = elicited[, 1:2],
                      probs = elicited[, 3],
                      lower = 0, upper = 100)
plotfit(dist_2expr, lp = TRUE, returnPlot = TRUE) +
  scale_color_grey() +
  theme_light()
```

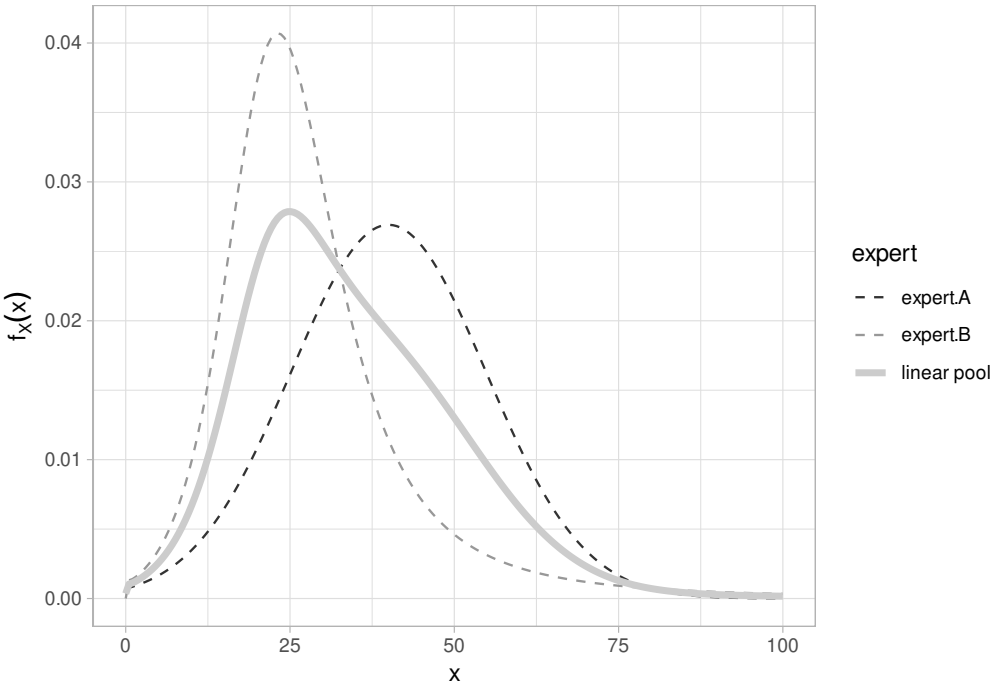

FIGURE E.11: Visualizing priors elicited from two experts for a parameter X representing a percentage ranging from 0 to 100.

Step 3: Then bring the two experts together and elicit a consensus distribution.

Suppose that the experts agree that $P(X < 25) = 0.25, P(X < 30) = 0.5, P(X < 40) = 0.75$. The consensus distribution is then:

```
elicited <- matrix(c(25, 0.25,
                     30, 0.5,
                     40, 0.75),
                   nrow = 3, ncol = 2, byrow = TRUE)
dist_cons <- fitdist(vals = elicited[,1],
                     probs = elicited[,2],
                     lower = 0, upper = 100)
plotfit(dist_cons, ql = 0.05, qu = 0.95, returnPlot = TRUE) +
  theme_light()
```

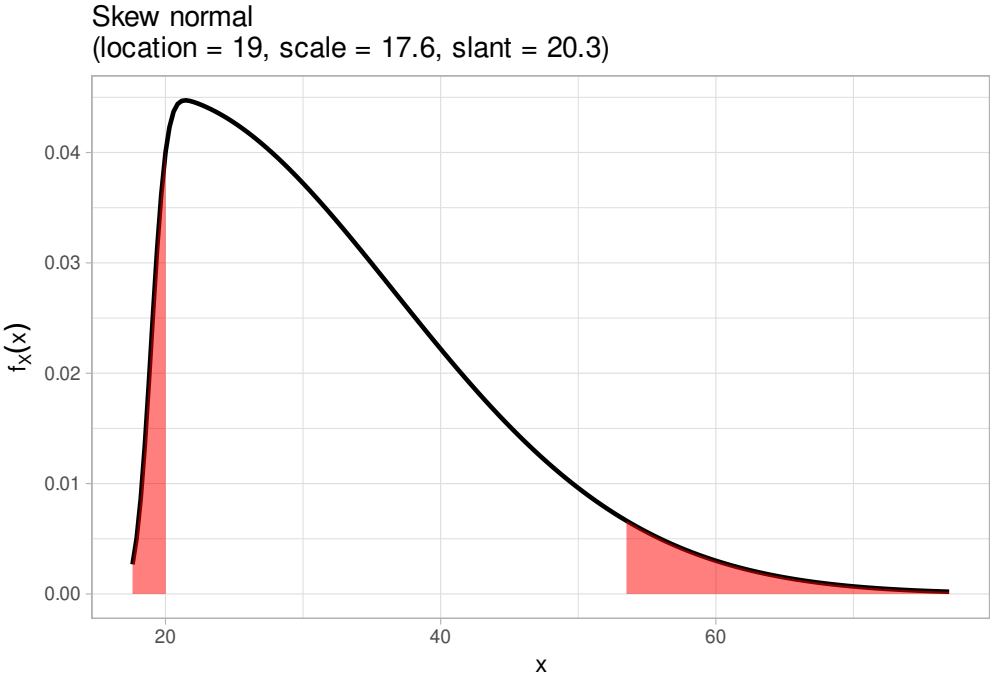

FIGURE E.12: Visualizing a consensus prior from two experts for a parameter X representing a percentage ranging from 0 to 100.

Step 4: Give feedback to the experts by showing them the 5th and 95th percentiles, and check that these bounds match their beliefs. If not, then repeat the above steps.

```
feedback(dist_cons, quantiles = c(0.05, 0.95))
```

```
## $fitted.quantiles
##      normal     t skewnormal gamma lognormal logt beta hist
## 0.05   12.5  7.49       20.0  16.2      17.3 14.8 15.2    5
## 0.95   50.4 55.10       53.5  53.2      55.3 64.1 51.2   88
##      mirrorgamma mirrorlognormal mirrorlogt
## 0.05        10.5            9.18       2.08
## 0.95        49.1           48.60      52.10
##
## $fitted.probabilities
##    elicited normal     t skewnormal gamma lognormal  logt
## 25      0.25  0.288 0.289       0.268 0.279     0.274 0.275
## 30      0.50  0.451 0.453       0.469 0.461     0.466 0.469
## 40      0.75  0.772 0.774       0.767 0.769     0.767 0.768
```

```
##     beta hist mirrorgamma mirrorlognormal mirrorlogt
## 25 0.283 0.25       0.292           0.295      0.296
## 30 0.456 0.50       0.446           0.444      0.447
## 40 0.770 0.75       0.772           0.773      0.775
```

E.3 Deriving priors from meta-analyses

Meta-analysis has been used widely in clinical research (Higgins and Green 2008; Sutton et al. 2012; DerSimonian and Laird 1986; Normand 1999) but, at least at the time of writing this book, it has been used relatively rarely in (psycho)linguistics. Random-effects meta-analysis (discussed in a later chapter in detail) is an especially useful tool in cognitive science.

Meta-analysis is not a magic bullet; this is because of publication bias—usually only (supposedly) newsworthy results are published, leading to a skewed picture of the effects. As a consequence, meta-analysis will probably always lead to biased estimates. Nevertheless, meta-analytic estimates can still tell us something about what we know so far from published studies, if only that the studies are too noisy to be interpretable. Thus, despite this limitation of meta-analytic estimates, some prior information is better than no information. As long as one remains aware of the limitations of meta-analysis, one can still use them effectively to study one's research questions.

We begin with observed effects y_n (e.g., estimated difference between two conditions) and their estimated standard errors (SEs); the SEs serve as an indication of the precision of the estimate, with larger SEs implying a low-precision estimate. Once we have collected the observed estimates (e.g., from published studies), we can define an assumed underlying generative process whereby each study $n = 1, \dots, N$ has an unknown true mean ζ_n:

$$y_n \sim Normal(\zeta_n, SE_n)$$

A further assumption is that each unknown true mean ζ_n in each study is generated from a distribution that has some true overall mean ζ, and standard deviation τ. The standard deviation τ reflects between-study variation, which could be due to different subjects being used in each study, different lab protocols, different methods, different languages being studied, etc.

$$\zeta_n \sim Normal(\zeta, \tau)$$

This kind of meta-analysis is actually the familiar hierarchical model we have already encountered in chapter 5. As in hierarchical models, hyperpriors have to be defined for ζ and τ. A useful application of this kind of meta-analysis

TABLE E.2: The difference between object and subject relative clause reading times (effect), along with their standard errors (SE), from different published reading studies on Chinese relative clauses. The data from Gibson et al 2013 will be removed in the meta-analysis, as we will use the posterior from the meta-analysis as an informative prior for analyzing the data from that study.

study.id	study	y	se
1	Hsiao et al 03	50.0	25.0
2	Gibson et al 13	−120.0	48.0
3	Vas. et al 13, E1	148.5	50.9
4	Vas. et al 13, E2	82.6	41.2
5	Vas. et al 13, E3	−109.4	54.8
6	Jaeg. et al 15, E1	55.6	65.1
7	Jaeg. et al 15, E2	81.9	36.3
8	Wu 09	50.0	23.0
9	Qiao et al 11, E1	−70.0	42.0
10	Qiao et al 11, E2	6.2	19.9
11	Lin & Garn. 11, E1	−100.0	30.0
12	Chen et al 08	75.0	35.5
13	C Lin & Bev. 06	100.0	80.0

is to derive a posterior distribution for ζ based on the available evidence; this posterior can be used (e.g., with a normal approximation) as a prior for a future study.

A simple example is the published data on Chinese relative clauses; the data are adapted from Vasishth (2015). Table E.2 shows the mean difference between object and subject relatives in Chinese, along with the standard error, that was derived from published reading studies on Chinese relatives.

Suppose that we want to do a new study investigating the difference between object and subject relative clauses, and suppose that in the sensitivity analysis, one of the priors we want is an empirically justifiable informative prior. Of course, the sensitivity analysis will also contain uninformative priors; we have seen examples of such priors in the previous chapters.

We can derive an empirically justified prior by conducting a group-level effects meta-analysis. We postpone discussion to chapter 11 of how exactly to fit such a model; here, we simply report the posterior distribution of the overall effect ζ based on the prior data, ignoring the details of model fitting.

First, the data from one study (Gibson and Wu 2013) is removed below, be-

cause the estimates from all the other studies will be used to derive a prior for that study.

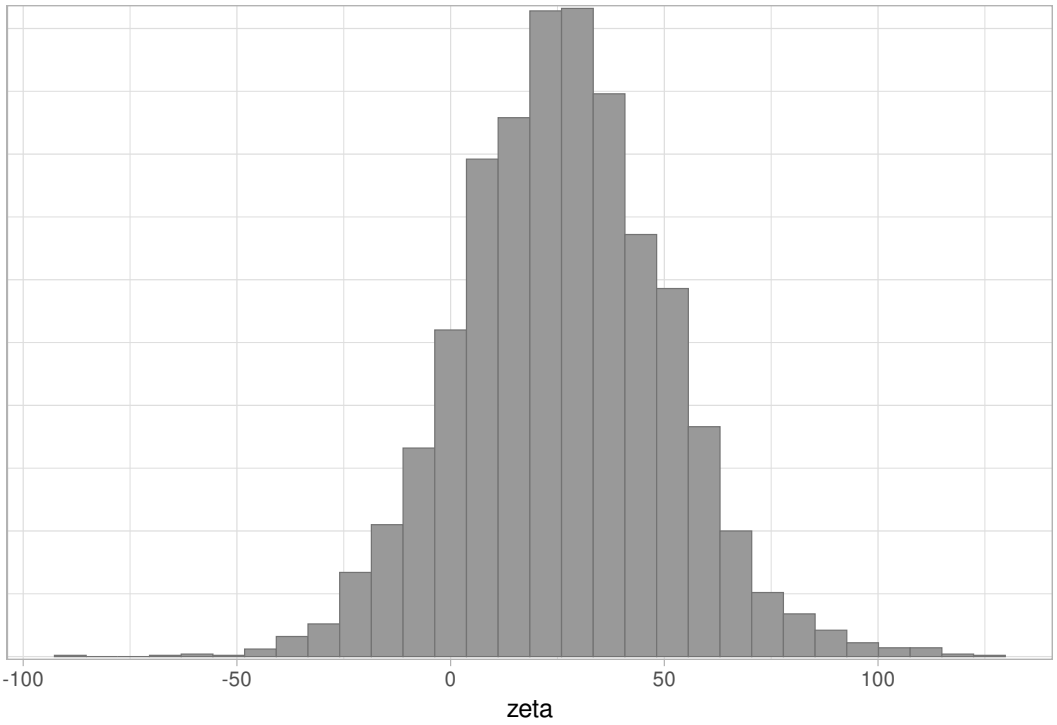

FIGURE E.13: The posterior distribution of the difference between object and subject relative clause processing in Chinese relative clause data, computed from a random-effects meta-analysis using published Chinese relative clause data from reading studies.

The posterior distribution of ζ is shown in Figure E.13. What we can derive from this posterior distribution of ζ is a normal approximation that represents what we know so far about Chinese relatives, based on the available data. The key here is the word "available"; almost certainly there exist studies that were inconclusive and were therefore never published. The published record is always biased because of the nature of the publication game in science (only supposedly newsworthy results get published).

The mean of the posterior is 26 ms, and the width of the 95% credible intervals is $75 - (-20) = 95$ ms. Since the 95% credible interval has a width that is approximately four times the standard deviation (assuming a normal distribution), we can work out the standard deviation by dividing the width by four: 23.75. Given these estimates, we could use a normal distribution with mean 26 and standard deviation 23.75 as an informative prior in a sensitivity analysis.

As an example, we will analyze the data set from Gibson and Wu (2013) that was not part of the above meta-analysis; recall that, for the above meta-analysis, the estimate from the Gibson and Wu (2013) study that was shown in Table E.2 has been removed. The meta-analysis posterior will then be used as an informative prior. First, load the data and sum-code the predictor:

```
data("df_gibsonwu")
df_gibsonwu <- df_gibsonwu %>%
  mutate(c_cond = if_else(type == "obj-ext", 1 / 2, -1 / 2))
```

Because we will now use a log-normal likelihood for the reading time data, we need to work out what the meta-analysis posterior of ζ corresponds to on the log scale. The grand mean reading time of the Gibson and Wu (2013) data on the log scale is 6.1. In order to arrive at approximately the mean difference of 26 ms, the log-scale value of the mean difference can be worked out as follows.

If we know the grand mean reading time $\hat{\alpha}$ on the log scale, then we can find out what the slope $\hat{\beta}$ needs to be such that the difference between the two conditions is approximately 26 ms.[3] In other words, we need to find $\hat{\beta}$ in the equation:

$$exp(\hat{\alpha} + \hat{\beta}/2) - exp(\hat{\alpha} - \hat{\beta}/2) = 26$$

One can find this estimate by trial and error, or solve the equation analytically. Similarly, if the lower and upper bounds of the effect estimate from the meta-analysis are known on the ms scale, we can figure out the lower and upper bounds of $\hat{\beta}$, call them $\hat{\beta}_{lower}$ and $\hat{\beta}_{upper}$:

$$exp(\hat{\alpha} + \hat{\beta}_{lower}/2) - exp(\hat{\alpha} - \hat{\beta}_{lower}/2) = -20$$

$$exp(\hat{\alpha} + \hat{\beta}_{upper}/2) - exp(\hat{\alpha} - \hat{\beta}_{upper}/2) = 75$$

Once we have estimates of $\hat{\beta}_{lower}$ and $\hat{\beta}_{upper}$, we can figure out the standard deviation estimate of the effect by computing the interval $\hat{\beta}_{upper} - \hat{\beta}_{lower}$ and dividing it by 4 (because the 95% credible interval will span four times the standard deviation). Thus, the end-result of our calculation is a mean and a standard deviation (on the log scale) of a normal distribution, which we can use as a relatively informative prior, informed by the meta-analysis, for the future study.

[3] There is no one-to-one correspondence between getting values from normal likelihoods and then use them in a log-normal likelihood, and this is just an approximated value.

```
(int <- mean(log(df_gibsonwu$rt)))
```

```
## [1] 6.06
```

```
## the effect size on the log ms scale:
b <- 0.058
## the slope on the log scale:
exp(int + b / 2) - exp(int - b / 2)
```

```
## [1] 24.9
```

```
## the lower bound on the log scale:
lower <- -0.052
exp(int + lower / 2) - exp(int - lower / 2)
```

```
## [1] -22.3
```

```
upper <- 0.17
exp(int + upper / 2) - exp(int - upper / 2)
```

```
## [1] 73
```

```
## the interval divided by 4:
(SE <- round( (upper - lower ) / 4, 3 ) )
```

```
## [1] 0.056
```

As always, we will do a sensitivity analysis, using uninformative priors on the slope parameter ($Normal(0, 1)$), as well as the meta-analysis prior ($Normal(0.058, 0.056)$).

```
## uninformative priors on the parameters of interest
## and on the variance components:
fit_gibsonwu_log <-
  brm(rt ~ c_cond +
        (c_cond | subj) +
        (c_cond | item),
      family = lognormal(),
      prior = c(prior(normal(6, 1.5), class = Intercept),
                prior(normal(0, 1), class = b),
```

TABLE E.3: A summary of the posteriors under a relatively uninformative prior and an informative prior based on a meta-analysis, for the Chinese relative clause data from Gibson and Wu, 2013.

Priors	Mean	Lower	Upper
Normal(0, 1)	−0.07	−0.18	0.04
Normal(0.058, 0.056)	−0.01	−0.08	0.08

```
                prior(normal(0, 1), class = sigma),
                prior(normal(0, 1), class = sd),
                prior(lkj(2), class = cor)),
      data = df_gibsonwu)
## meta-analysis priors:
fit_gibsonwu_ma <-
  brm(rt ~ c_cond +
        (c_cond | subj) +
        (c_cond | item),
      family = lognormal(),
      prior = c(
        prior(normal(6, 1.5), class = Intercept),
        prior(normal(0.058, 0.056), class = b),
        prior(normal(0, 1), class = sigma),
        prior(normal(0, 1), class = sd),
        prior(lkj(2), class = cor)),
      data = df_gibsonwu)
```

A summary of the posteriors (means and 95% credible intervals) under the *Normal*(0, 1) and the meta-analysis prior is shown in Table E.3. In this particular case, the posteriors are influenced by the two different priors. The differences between the two posteriors are small, but these differences could in principle lead to different outcomes (and conclusions) in a Bayes factor analysis.

E.4 Using previous experiments' posteriors as priors for a new study

In a situation where we are attempting to replicate a previous study's results, we can derive an informative prior for the analysis of the replication attempt

TABLE E.4: A summary of the posteriors under an uninformative prior ($Normal(0, 1)$), a prior based on previous data, and a meta-analysis prior, for data from a replication attempt of Gibson and Wu, 2013.

Priors	Mean	Lower	Upper
Normal(0,1)	−0.08	−0.20	0.05
Normal(-0.07,0.2)	−0.08	−0.20	0.04
Normal(0.041, 0.2)	−0.07	−0.19	0.05

by figuring out a prior based on the previous study's posterior distribution. In the previous chapter, we encountered this in one of the exercises: Given data on Chinese relatives (Gibson and Wu 2013), we want to replicate the effect with a new data set that has the same design but different subjects. The data from the replication attempt is from Vasishth et al. (2013).

The first data set from Gibson and Wu (2013) was analyzed in the previous section using uninformative priors. We can extract the mean and standard deviation of the posterior, and use that to derive an informative prior for the replication attempt.

Now, for the replication study, we can use this posterior (with a normal approximation), if we want to build on what we learned from the original Gibson and Wu (2013) study. As usual, we will do a sensitivity analysis: one model is fit with an uninformative prior on the parameter of interest, $Normal(0, 1)$, as we did in the preceding section; and another model will be fit with the informative directional prior $Normal(-0.073, 0.04)$. For good measure, we can also include a model with a prior derived from the meta-analysis in the preceding section (the posterior of the ζ parameter).

Table E.4 summarizes the different posteriors under the three prior specification. Again, in this case, the differences in the posteriors are small, but in a Bayes factor analysis, the outcomes under these different priors could be different.

This particular analysis shows that under all the priors considered, the mean of the effect estimate is in the predicted negative direction; this implies that on average the object minus subject relative clause reading time difference is such that object relatives are read faster. However, the credible intervals span a wide range under all the priors; this is a consequence of the relatively small sample size in the study considered here.

E.5 Summary

Working out appropriate priors for one's research problem is essentially a Fermi problem. One can use several different strategies for working out priors: introspection, a literature review, computing statistics from existing data, conducting a meta-analysis, using posteriors from existing data as priors for a new, closely related study, or formally eliciting priors from domain experts using a pre-defined prior-elicitation protocol. If a prior specification is too vague, this can lead to slow convergence or convergence problems, and could lead to biased Bayes factors (biased towards the null hypothesis); and if a prior is too informative, this can also bias the posterior. This inherent potential for bias in prior specification should be formally investigated using sensitivity analyses (with a collection of uninformative, skeptical, and informative priors of various types), and prior and posterior predictive checks. Although prior specification seems like a daunting task to the beginning student of Bayes, with time and experience one can develop a very well-informed set of priors for one's research problems.

E.6 Further reading

For interesting (and amusing) examples of Fermi solutions to questions, see `https://what-if.xkcd.com/84/`. Two important books, Mahajan (2010) and Mahajan (2014), unpack the art of approximation in mathematics and other disciplines; the approach presented in these books is closely related to the art of Fermi-style approximation. Levy (2021) is an important book that develops the analytical skill needed to figure out what your "tacit knowledge" about a particular problem is. Tetlock and Gardner (2015) explains how experts deploy existing knowledge to derive probabilistic predictions (predictions that come with a certain amount of uncertainty) about real-world problems—this skill is closely related to prior (self-)elicitation. An excellent presentation of prior elicitation is in O'Hagan et al. (2006). Useful discussions about priors are provided in Lunn et al. (2012); Spiegelhalter, Abrams, and Myles (2004); Gelman, Simpson, and Betancourt (2017); and Simpson et al. (2017). The Stan website also includes some guidelines: Prior distributions for `rstanarm` models in `https://mc-stan.org/rstanarm/articles/priors.html`; and prior choice recommendations in `https://github.com/stan-dev/stan/wiki/Prior-Choice-Recommendations`. Browne and Draper (2006) and Gelman (2006) discuss prior specifications in hierarchical models.

F

Workflow

Although modern Bayesian analysis tools (such as `brms`) greatly facilitate Bayesian computations, model specification is still (as it should be) the responsibility of the user. Chapter 3 is one of the earlier chapters where some of the steps required to arrive at a robust and useful analysis were described. In this chapter, these ideas are brought together to spell out a principled approach to developing a workflow. This chapter is an abbreviated version a recent introduction of a principled Bayesian workflow to cognitive science (Schad, Betancourt, and Vasishth 2019). For a revised published version of the 2019 paper, see Schad, Betancourt, and Vasishth (2020).

A lot of research has been done recently in order to create tools that guarantee reliable Bayesian data analyses (see, for example, Gabry et al. 2019; Talts et al. 2018). The development of a principled Bayesian workflow for performing a probabilistic analysis is one of the most recent outcomes of this research (Betancourt 2018; Schad, Betancourt, and Vasishth 2019). This process leaves space for future advancements in methodology and offers a logical first set of steps to take for a robust analysis. Parts of this workflow can, in principle, be applied to any type of data analysis, whether frequentist or Bayesian, whether sampling-based or based on analytic procedures.

In this chapter, we discuss some aspects of the principled Bayesian workflow. Certain components of this workflow are particularly useful when working with advanced or non-standard models.

When fitting a model, it is important to ask several questions and perform various checks to validate a probabilistic model. Before delving into the details of this discussion, we first examine the process of model building and how different traditions have led to different approaches to these questions.

F.1 Building a model

An effective approach to model building is to begin with a minimal model that captures only the phenomenon of interest, without incorporating much

other structure in the data. For instance, this could be a linear model that includes only the factor or covariate of primary interest. This model is then subjected to a series of checks, which are described in detail in the following sections. If the model passes all of these checks and does not exhibit any signs of inadequacy, it can be applied to the problem at hand with confidence, knowing that it provides reasonably robust inferences for the scientific question that needs to be answered. However, if the model fails one or more of these checks, the model may need to be improved; in addition, even the scientific question may need to be changed. For example, in a repeated measures data set, we may use a sample of 30 subjects with the aim to estimate the correlation parameter between by-group adjustments (their random effects correlation). If the analysis shows that the sample is not large enough to reliably estimate the correlation term, the sample size may need to be increased, or the plan to investigate any correlation may need to be abandoned.

To guide and inform model development, initially an aspirational model $\mathscr{M}_A$ is specified. This model is an idea that encompasses every aspect of the phenomenon and the measurement procedure, as if time, money, subjects, computational and mathematical tools, and other resources were all infinite. It accounts for all systematic effects that might influence the measurement process, such as influences of time or heterogeneity across individuals. By using this model as a starting point, random walks in model space can be avoided during the development of the model. The model has to capture both the latent phenomenon of interest and also the environment and experiment used to probe it.

The initial model $\mathscr{M}_1$ is designed to incorporate only the phenomenon of core scientific interest, without including any additional aspects or structures relevant for modeling or measurement. This is in contrast to the aspirational model $\mathscr{M}_A$, which includes all possible details (of course, within reason) of the phenomenon and measurement process. If the initial model turns out to be inadequate, then the aspirational model guides model development. In case the expanded model still shows problems in model checking, then model development is continued with another cycle of development.

In the following sections, prior and posterior predictive checks are discussed briefly, because they provide a foundation for a principled approach to model expansion. Critically, model development is best built up via *expansion*. In the case that an expanded model turns out to not be a better description of the data, it's always possible to go back to a previous, simpler, version of the model.

An alternative strategy for model fitting is proposed by some researchers, whereby the model contains all group-level variance components (e.g., by-

participant and by-items) that are allowed by the experimental design, as well as a full variance-covariance matrix for all group-level parameters. A commonly used name for this model, especially in psychology and psycholinguistics, is the "maximal" model (e.g., Barr et al. 2013). However, this model can be seen as "maximal" only in the framework of a linear model. For example, section 5.2.6 treated distributional models, which are already beyond the scope of the maximal models in the sense of Barr et al. (2013). Nevertheless, a maximal model can provide an alternative starting point for the principled Bayesian workflow. Here, model expansion is not the focus. Instead, if the maximal model approach is taken, the workflow that we discuss here can be useful for specifying priors encoding domain expertise, and to ensure model adequacy. In the principled Bayesian workflow, it may be reasonable for some steps (especially computationally demanding steps) to be executed only one time for a given series of related studies (with similar designs) or only if models are coded in Stan.

The term "maximal" in a maximal model as used by Barr et al. (2013) refers to the maximal specification of the variance components within the parameters of the linear regression approximation. Models constrained by the linear regression structure are unable to account for factors like measurement error, dynamic changes in processes over time, or selection bias in the data. Crucially, the aspirational model–a representation of the actual data-generating process–is not the "maximal" model.

Last but not least, occasionally the outcomes of the Bayesian workflow will demonstrate that the data or experimental design used is insufficient to address the specific scientific question at hand. In this instance, either the level of ambition must be lowered or fresh data must be gathered, maybe using an alternative experimental design that is more sensitive to the relevant phenomenon.

Pre-registration of experimental analyses prior to data collection is a significant development in open science practices (Chambers 2019). Online resources like AsPredicted (`https://aspredicted.org/`) and the Open Science Foundation (`https://osf.io/`) can be used for this (but see Szollosi et al. 2020). What details should or can be recorded during the Bayesian workflow's preregistration? The population- and group-level effects (also referred to as fixed and random effects) and contrast coding (Schad et al. 2020) should be described if the maximal model in the sense of Barr et al. (2013) is going to be used for analysis. Rigid preregistration is meaningless in the context of incremental model building unless one knows precisely what the model is, as any subsequent inference will be limited, if not useless, if the model isn't appropriate for the data at hand. Preregistration's deeper problem is that a model cannot be

validated until the phenomenon *and* experiment are thoroughly understood. Defining the initial and aspirational models as well as the incremental strategy used to probe the initial model in order to move it closer to the aspirational model is one feasible option. It can be helpful to preregister the initial model and the principles one plans to use in model selection; this is a helpful step even though it can be challenging, or indeed impossible, to fully define the aspirational model. The incremental model building strategy towards the aspirational model may be seen as lying at the boundary between confirmatory and exploratory, and becomes more confirmatory the more clearly the aspirational model can be spelled out a priori (Lee et al. 2019).

F.2 Principled questions to ask on a model

What qualities are key for a useful probabilistic model? A first quality is consistency with domain expertise. Furthermore, in order to effectively address scientific inquiries, a probabilistic model must possess sufficient richness to accurately represent the structure of the actual data generation process. Two additional requirements must be met when developing very complex or non-standard models (which we will touch on briefly in this chapter): the model must allow accurate posterior approximation and it must capture enough of the experimental design to provide meaningful insights into our research questions.

In order to meet these requirements for our probabilistic model, what can we do? We will go over the several analysis steps and questions to ask.

We will first examine whether our model is consistent with our domain expertise using prior predictive checks. Additionally, posterior predictive checks evaluate whether the model adequately captures the relevant structure of the actual data-generating process for the given data set. We will also touch on two more, computationally costly steps that are part of the principled workflow and can be used, for example, when coding complex or non-standard models: this includes examining (a) model sensitivity, and (b) the question of whether we can recover model parameters with the provided design, including checks of computational faithfulness, by examining whether posterior estimation is accurate.

F.2.1 Checking whether assumptions are consistent with domain expertise: Prior predictive checks

When investigating the model, it is crucial to first determine whether the model and the prior parameter distributions are in line with domain knowledge. Prior distributions may be chosen on the basis of previous studies or practicality. It is frequently challenging to determine which prior distributions to use for complex models, as well as the effects that distributions of prior model parameters have on the a priori expected data. One workable solution is to simulate artificial data from the model using prior distributions, and then verify if the simulated data make sense and align with domain knowledge. When compared to directly evaluating prior distributions in complex models, this (simulation-based) method is frequently far simpler to judge.

To put this strategy into action, take the following actions:

1. Using the prior $p(\boldsymbol{\Theta})$, draw a parameter set $\boldsymbol{\Theta}_{pred}$ from it via random sampling: $\boldsymbol{\Theta}_{pred} \sim p(\boldsymbol{\Theta})$
2. Based on this parameter set $\boldsymbol{\Theta}_{pred}$, simulate artificial data $\boldsymbol{y}_{pred}$ from the model: $\boldsymbol{y}_{pred} \sim p(\boldsymbol{y} \mid \boldsymbol{\Theta}_{pred})$

It is helpful to compute summary statistics of the simulated data $t(\boldsymbol{y}_{pred})$ in order to evaluate whether previous model predictions are consistent with domain expertise. Histograms can be used to display the distribution of these summary statistics (see Figure F.1). This may rapidly show whether the data falls within an expected range or whether a sizable number of extreme data points are expected a priori. Extreme values, for instance, might be reading times less than 50 ms or more than 2000 ms in a study utilizing self-paced reading times. While reading times longer than 2000 ms for a word are not impossible, they are unlikely and largely at odds with domain knowledge. Reading research has demonstrated that a tiny percentage of observations may truly take extreme values. However, if we find a substantial number of extreme data points in the hypothetical data and if these are at odds with domain expertise, then the model or the priors should be changed to produce hypothetical data that falls within the range of acceptable values.

Selecting quality summary statistics is more art than science. However, the choice of summary statistics will be important since they offer important indicators of the information that we want the model to take into account. As a result, they ought to be carefully selected and created in accordance with our expectations regarding the actual process of data generation as well as the kinds of structures and effects we expect the data will display. One can use summary statistics to criticize the model as well. For example, one can

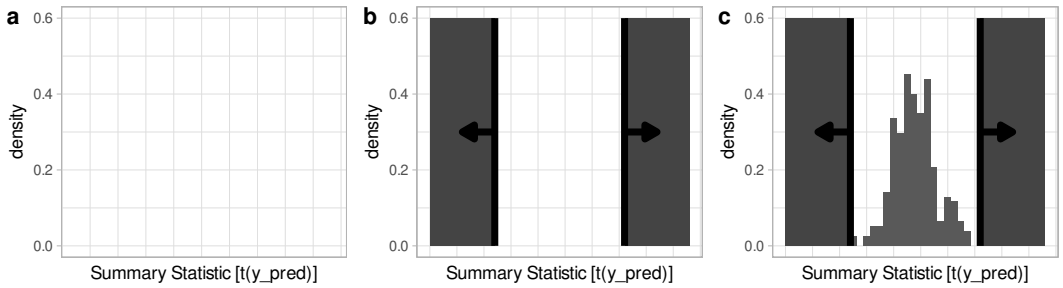

FIGURE F.1: Prior predictive checks. a) In a first step, define a summary statistic that one wants to investigate. b) Second, define extremity thresholds (shaded areas), beyond which one does not expect a lot of data to be observed. c) Third, simulate prior model predictions for the data (histogram) and compare them with the extreme values (shaded areas).

formalize criticism of an analysis into a summary statistic that one believes will exhibit undesirable behavior.

Selecting appropriate priors will be especially important when the data does not provide enough information to determine the likelihood (see Figure F.2, especially g-i). This frequently happens, for instance, in hierarchical models when a "maximal" model is fitted for a small data set that does not constrain the estimation of variance and covariance parameters for all group-level effects.[1]

In such cases, a prior in a Bayesian analysis (or a more informative one instead of a relatively uninformative one) should incorporate just enough domain expertise to suppress extreme but not impossible parameter values. Since the posterior is now sufficiently constrained, it may now be possible to fit the model. Therefore, by incorporating prior knowledge into Bayesian computation, we can fit and understand models that frequentist tools are unable to reliably estimate.

Thus, more concentrated prior distributions are a welcome side-effect of adding more domain expertise (into what still constitutes weakly informative priors), which can help with Bayesian computation. This makes it possible to estimate more complex models; in other words, models that would not otherwise be able to be estimated with the tools at hand, can be fitted thanks to the use of prior knowledge. Put another way, by utilizing prior knowledge, the iterative model-building process can help us approach the aspirational model better. Moreover,

[1]This issue shows up as problems with optimizer convergence in frequentist methods (like the ones used in the `lme4` package), indicating that the likelihood is too flat and the parameter estimates are not limited by the data.

MCMC algorithms will converge more quickly once additional informative priors are provided.

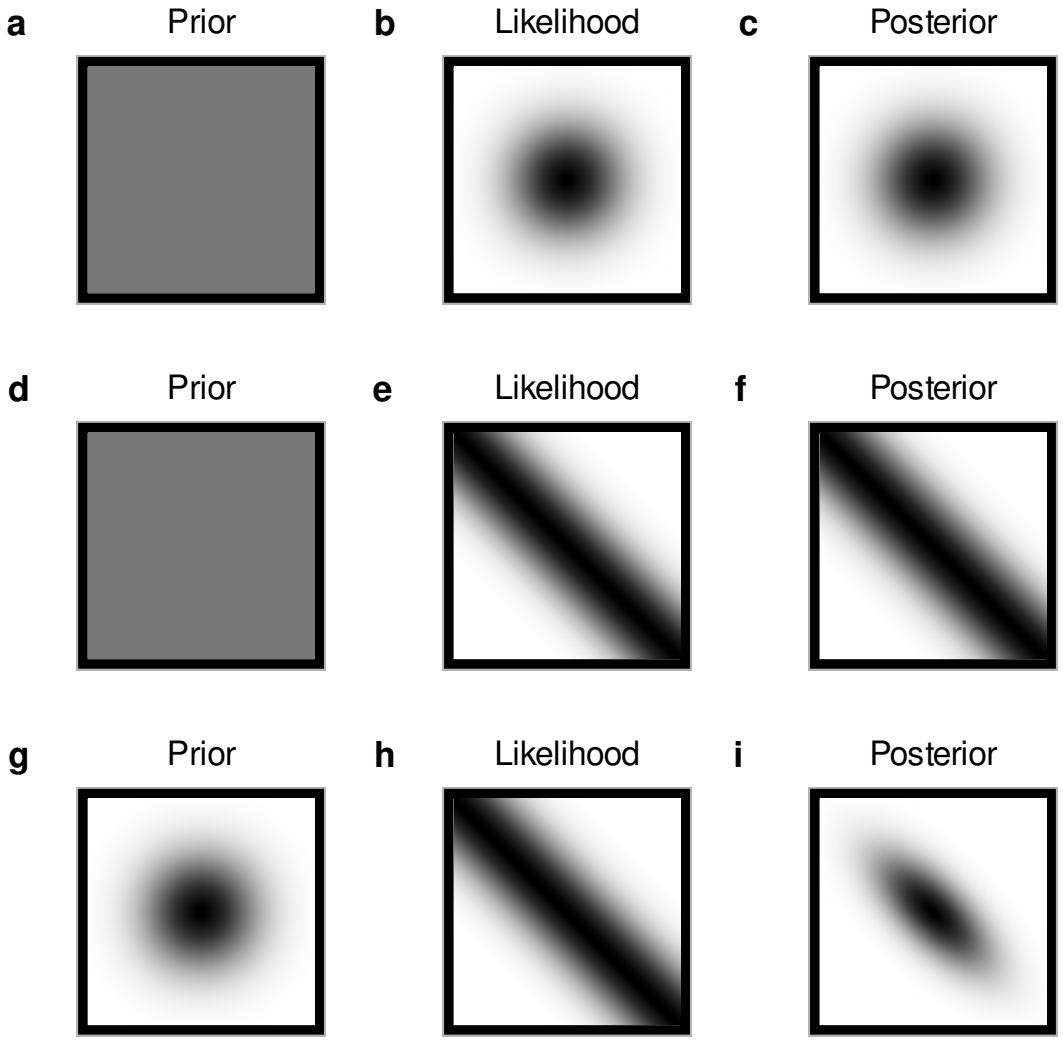

FIGURE F.2: The role of priors when data is informative or uninformative. a)-c) When the data offers good information through the likelihood (b), a flat uninformative prior (a) is sufficient to obtain a concentrated posterior (c). d)–f) When the data does not adequately constrain the parameters through the likelihood (e), using a flat uninformative prior (d) results in a widely distributed posterior (f) (i.e., different combinations of parameters are equally plausible; this hints at the fact that the model is empirically not identifiable). g)-i) When the data does not constrain the parameter through the likelihood (h), adding domain knowledge through an informative prior (g) can help constrain the posterior (i) to reasonable values.

When computing Bayes factors, adding more domain expertise to the prior

has important implications for Bayesian modeling (see chapter 13, on Bayes factors).

Crucially, the prior predictive distribution, which describes the interaction between the prior and the likelihood, can be used to simulate prior predictive data. It calculates an integral, or average, over various possible (prior) parameter values mathematically. As previously mentioned (also refer to chapter 3), the prior predictive distribution is:

$$\begin{aligned} p(\boldsymbol{y}_{pred}) &= \int p(\boldsymbol{y}_{pred}, \boldsymbol{\Theta})\, d\boldsymbol{\Theta} = \int p(\boldsymbol{y}_{pred} \mid \boldsymbol{\Theta}) p(\boldsymbol{\Theta})\, d\boldsymbol{\Theta} \\ &= \int \text{likelihood}(\boldsymbol{y}_{pred} \mid \boldsymbol{\Theta}) \cdot \text{prior}(\boldsymbol{\Theta})\, d\boldsymbol{\Theta} \end{aligned} \tag{F.1}$$

As an illustration, let's say that we consider our likelihood to be a normal distribution with mean μ and standard deviation σ. Assume that we now define $\sigma \sim \mathit{Uniform}(1, 2)$ and $\mu \sim \mathit{Normal}(0, 1)$ as priors on the parameters. The steps below can be used to create the prior predictive distribution:

- Perform the following 100,000 times:
  - Select one sample (m) from the distribution $\mathit{Normal}(0, 1)$.
  - Take one sample (s) from a $\mathit{Uniform}(1, 2)$ distribution
  - Create and store a data point from a $\mathit{Normal}(m, s)$ distribution.
- The prior predictive distribution is the generated data.

It is also possible to define more intricate generative processes involving data from repeated measures.

F.2.2 Testing for correct posterior approximations: Checks of computational faithfulness

Approximations of posterior expectations can be inaccurate. For example, a computer program that is designed to sample from a posterior can be erroneous. This may be due to an error in the likelihood specification (for example, an error in the R syntax formula) or to an inadequate sampling of the posterior's entire density. The sampler may be biased, meaning the parameter samples might be systematically larger or smaller than those drawn from the exact posterior. Alternatively, the variance of the posterior samples may differ, being either larger or smaller than the variance of the exact posterior. However, posterior sampling from simple and standard models should work properly in most cases. Thus, we think that in many applications, a further check of computational faithfulness may be asking for too much, and might need to be performed only once for a given research program, where different experiments are rather similar to each other. However, checking compu-

tational faithfulness can become an important issue when dealing with more advanced/non-standard models (such as those discussed in the later chapters of this book). Here, errors in the specification of the likelihood can occur more easily.

Designing a process to verify whether the posterior approximation of choice is accurate is crucial because posterior approximations can be erroneous. For example, one should make sure that the software utilized to implement the sampling is error-free for the particular problem at hand. This checking can be performed using simulation-based calibration (SBC; Talts et al. 2018; Schad, Betancourt, and Vasishth 2019; Modrák et al. 2023). This is a computationally intensive procedure that can take a long time to run for particularly complex models and large data sets. We do not discuss SBC in detail here, but refer the reader to its later treatment in chapter 16, where SBC is applied for models coded in Stan directly, as well as to the description in Schad, Betancourt, and Vasishth (2019; and also Modrák et al. 2023).

After confirming the accuracy and faithfulness of our posterior computations, we can move on to examine the model analyses' sensitivity.

F.2.3 Sensitivity of the model

What can we reasonably expect from a model's posterior, and how can we determine whether these expectations are reasonable given the current configuration? First, we might expect that the data are generated without bias as the posterior recovers the true values of the parameters. In other words, we could anticipate that the posterior mean will be near to the true value when simulating hypothetical data based on a true parameter value (a point value for the parameter). This expectation, however, might or might not be warranted for a particular model, experimental setup, and data set. In fact, some models–such as non-linear models–may have biased parameter estimates, making it nearly impossible to determine the parameter's true value from the data. Simultaneously, we could expect that the posterior is very informative concerning the parameters that produced the data. In other words, in comparison to our past knowledge, we could aim for a small posterior uncertainty, or a small posterior standard deviation. But posterior certainty isn't always high. When compared to our past knowledge, some experimental designs, models, priors, or data sets may produce extremely uninformative estimates where the degree of uncertainty is not decreased. This may occur when there is a dearth of data or when the model is too complex for the experimental design, preventing us from constraining specific model parameters.

In order to examine model sensitivity, two model-related questions can be looked into:

1) How closely does the estimated posterior mean of a parameter match its true (simulated) value?
2) To what extent is uncertainty reduced between the posterior and the prior?

To investigate these questions, it is again possible to perform extensive simulation studies. (Often it might be sufficient to simulate a few data sets with different parameter/design settings, instead of running a full simulation.) This is crucial to do for complex, non-standard, or cognitive models, but may be less important for simpler and more standard models. Indeed, the same set of simulations can be used that are also used in SBC. Therefore, both analyses can be usefully applied in tandem. Again, here we skip the details of how these computations can be implemented, and refer the interested reader to Schad, Betancourt, and Vasishth (2019).

F.2.4 Does the model adequately capture the data?–Posterior predictive checks

"*All models are wrong but some are useful.*" (Box 1979, 2).

We are aware that the observed data noisily reflects the true data generating process, which our model most likely does not fully capture. Therefore, we want to know to what extent our model accurately approximates the true process that produced the data. We can simulate data from the model and compare the simulated to the real data in order to compare the model to the actual data generating process (i.e., to the data). A posterior predictive distribution (refer to chapter 3) can be used to formulate this: the model is fitted to the data, and new data is simulated using the estimated posterior model parameters.

The posterior predictive distribution can be expressed mathematically as follows:

$$p(\boldsymbol{y}_{\boldsymbol{pred}} \mid \boldsymbol{y}) = \int p(\boldsymbol{y}_{\boldsymbol{pred}} \mid \boldsymbol{\Theta}) p(\boldsymbol{\Theta} \mid \boldsymbol{y}) \, d\boldsymbol{\Theta} \tag{F.2}$$

Here, the posterior distribution over model parameters, $p(\boldsymbol{\Theta} \mid \boldsymbol{y})$, is inferred from the observed data, $\boldsymbol{y}$. To generate future data, $\boldsymbol{y}_{\boldsymbol{pred}}$, the posterior distribution of $\boldsymbol{\Theta}$ is combined with the distribution of the future data $\boldsymbol{y}_{\boldsymbol{pred}}$ given $\boldsymbol{\Theta}$, $p(\boldsymbol{y}_{\boldsymbol{pred}} \mid \boldsymbol{\Theta})$. Averaging over various possible values for the posterior model parameters ($\boldsymbol{\Theta}$) is indicated by the integral $\int d\boldsymbol{\Theta}$.

As stated in chapter 3, we are unable to evaluate this integral exactly. Since $\boldsymbol{\Theta}$ can be a vector with multiple parameters, this integral is extremely complex

and lacks an analytical solution. On the other hand, sampling allows us to approximate it. We can first obtain samples from the parameter posterior. Next, we can use these posterior samples to simulate new, artificial data from the model. This process approximates the posterior predictive distribution (and also gets rid of the necessity of computing the exact value or integral of the posterior predictive density $p(\boldsymbol{y_{pred}} \mid \boldsymbol{y})$).

In summary, we first fit the model to the data to obtain the posterior, and then simulate new data using the estimated posterior model parameters. The crucial question is then how closely the new simulated data resembles the observed data.

One strategy for comparing the data and the model is to use important features from the data and gauging the model's ability to capture them. In fact, in the prior predictive checks, we had already defined summary statistics. Now that we have the data simulated from the posterior predictive distribution, we can compute these summary statistics. Every summary statistic will then have a distribution. We also compute the summary statistic for the observed data. We can now determine whether the observed data falls within the distribution of the model predictions (see Figure F.3a) or whether the model predictions deviate significantly from the observed data (see Figure F.3b).

Descriptive adequacy is supported if the observed data closely match the posterior-predicted data. A substantial disparity might suggest three possibilities: (1) Our model may be overlooking important details about the processes we care about, in which case we need to apply our domain knowledge to further enhance the model. (2) Our model does not overlook important details, but there are a number of very low probability observations that were nonetheless produced by the process we modeled. (3) Our model might be missing details about less critical processes that are producing extreme observations (e.g., lapses in attention or errors in data collection). We might choose to address these issues or simply remove the problematic observations. Generally speaking, it is very difficult to distinguish between these three possibilities, so we must use our best judgment. Specifically, we ought to modify the model exclusively in situations where the disparity aligns with a recognized absent model feature. Note that if we perform a lot of checks (e.g., per subject), it is not too surprising if the discrepancy is substantial a few times.

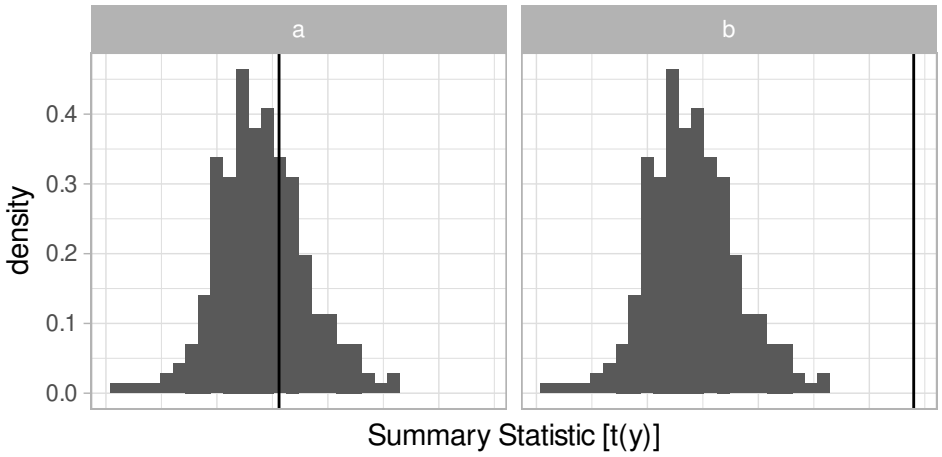

FIGURE F.3: Posterior predictive checks. For a particular summary statistic, t(y), compare the posterior model predictions (histogram) with the observed data (vertical line). a) This illustrates a situation where the posterior model predictions (histogram) and the observed summary statistic (vertical line) coincide. b) This illustrates an instance in which the observed data's summary statistic (vertical line) falls outside of the model's posteriori predictions (histogram).

F.3 Further reading

Some important articles relating to developing a principled Bayesian workflow are by Betancourt (2018), Gabry et al. (2019), Gelman et al. (2020), and Talts et al. (2018). The `stantargets` R package provides tools for a systematic, efficient, and reproducible workflow (Landau 2021). Also recommended is the article on reproducible workflows by Wilson et al. (2017).

G

Exercises

G.1 Introduction

G.1.1 Practice using the `pnorm()` function–Part 1

Given a normal distribution with mean 500 and standard deviation 100, use the `pnorm()` function to calculate the probability of obtaining values between 200 and 800 from this distribution.

G.1.2 Practice using the `pnorm()` function–Part 2

Calculate the following probabilities. Given a normal distribution with mean 800 and standard deviation 150, what is the probability of obtaining:

- a score of 700 or less
- a score of 900 or more
- a score of 800 or more

G.1.3 Practice using the `pnorm()` function–Part 3

Given a normal distribution with mean 600 and standard deviation 200, what is the probability of obtaining:

- a score of 550 or less.
- a score between 300 and 800.
- a score of 900 or more.

G.1.4 Practice using the `qnorm()` function–Part 1

Consider a normal distribution with mean 1 and standard deviation 1. Compute the lower and upper boundaries such that:

- the area (the probability) to the left of the lower boundary is 0.10.
- the area (the probability) to the left of the upper boundary is 0.90.

G.1.5 Practice using the `qnorm()` function–Part 2

Given a normal distribution with mean 650 and standard deviation 125. There exist two quantiles, the lower quantile q1 and the upper quantile q2, that are equidistant from the mean 650, such that the area under the curve of the normal between q1 and q2 is 80%. Find q1 and q2.

G.1.6 Practice getting summaries from samples–Part 1

Given data that is generated as follows:

```
data_gen1 <- rnorm(1000, mean = 300, sd = 200)
```

Calculate the mean, variance, and the lower quantile q1 and the upper quantile q2, that are equidistant and such that the range of probability between them is 80%.

G.1.7 Practice getting summaries from samples–Part 2.

This time we generate the data with a truncated normal distribution from the package `extraDistr`. The details of this distribution will be discussed later in section 4.1 and in the online section A.2, but for now we can treat it as an unknown generative process:

```
data_gen1 <- rtnorm(1000, mean = 300, sd = 200, a = 0)
```

Using the sample data, calculate the mean, variance, and the lower quantile q1 and the upper quantile q2, such that the probability of observing values between these two quantiles is 80%.

G.1.8 Practice with a variance-covariance matrix for a bivariate distribution.

Suppose that you have a bivariate distribution where one of the two random variables comes from a normal distribution with mean $\mu_X = 600$ and standard deviation $\sigma_X = 100$, and the other from a normal distribution with mean $\mu_Y = 400$ and standard deviation $\sigma_Y = 50$. The correlation ρ_{XY} between the two random variables is 0.4. Write down the variance-covariance matrix of this bivariate distribution as a matrix (with numerical values, not mathematical symbols), and then use it to generate 100 pairs of simulated data points. Plot the simulated data such that the relationship between the random variables X and Y is clear. Generate two sets of new data (100 pairs of data points

each) with correlation −0.4 and 0, and plot these alongside the plot for the data with correlation 0.4.

G.2 Introduction to Bayesian data analysis

G.2.1 Deriving Bayes' rule

Let A and B be two observable events. $P(A)$ is the probability that A occurs, and $P(B)$ is the probability that B occurs. $P(A|B)$ is the conditional probability that A occurs given that B has happened. $P(A, B)$ is the joint probability of A and B both occurring.

You are given the definition of conditional probability:

$$P(A|B) = \frac{P(A, B)}{P(B)} \text{ where } P(B) > 0 \tag{G.1}$$

Using the above definition, and using the fact that $P(A, B) = P(B, A)$ (i.e., the probability of A and B both occurring is the same as the probability of B and A both occurring), derive an expression for $P(B|A)$. Show the steps clearly in the derivation.

G.2.2 Conjugate forms 1

- Computing the general form of a PDF for a posterior

Suppose you are given data k consisting of the number of successes, coming from a $Binomial(n, \theta)$ distribution. Given k successes in n trials coming from a binomial distribution, we define a $Beta(a, b)$ prior on the parameter θ.

Write down the Beta distribution that represents the posterior, in terms of a, b, n, and k.

- Practical application

We ask 10 yes/no questions from a subject, and the subject returns 0 correct answers. We assume a binomial likelihood function for these data. Also assume a $Beta(1, 1)$ prior on the parameter θ, which represents the probability of success. Use the result you derived above to write down the posterior distribution of the θ parameter.

G.2.3 Conjugate forms 2

Suppose that we perform n independent trials until we get a success (e.g., a heads in a coin toss). For repeated coin tosses, observing T,T, H would correspond to a score of $n = 3$. The probability of success in each trial is θ. Then, the Geometric random variable, call it X, gives us the probability of getting a success in n trials as follows:

$$Prob(X = n) = \theta(1 - \theta)^{n-1} \tag{G.2}$$

where $n = 1, 2, \ldots$.

Let the prior on θ be $Beta(a, b)$, a beta distribution with parameters a,b. The posterior distribution is a beta distribution with parameters a^* and b^*. Determine these parameters in terms of a, b, and n.

G.2.4 Conjugate forms 3

Suppose that we have n data points, $x_1, \ldots, x_n$, drawn independently from an exponential distribution with parameter λ. The parameter of interest here (what we want to learn about from the data) is λ.

The exponential likelihood function is:

$$p(x_1, \ldots, x_n|\lambda) = \lambda^n \exp(-\lambda \sum_{i=1}^{n} x_i) \tag{G.3}$$

Starting with a Gamma prior distribution for λ (see below), show that the posterior distribution for λ is also a Gamma distribution. Provide formulas giving the posterior parameters a^*, b^* in terms of the prior parameters a, b and the data. Use the following facts about Gamma distributions.

The Gamma distribution is defined in terms of the parameters a, b: $Gamma(a, b)$. In general, if there is a random variable Y (where $y \geq 0$) that has a Gamma distribution as a PDF ($Y \sim Gamma(a, b)$), then:

$$Gamma(y|a, b) = \frac{b^a y^{a-1} \exp(-by)}{\Gamma(a)} \tag{G.4}$$

The $Gamma(a, b)$ prior on the λ parameter in the exponential distribution will be written:

$$Gamma(\lambda|a, b) = \frac{b^a \lambda^{a-1} \exp(-b\lambda)}{\Gamma(a)} \tag{G.5}$$

G.2.5 Conjugate forms 4

- Computing the posterior

This is a contrived example. Suppose we are modeling the number of times that a speaker says the word "I" per day. This could be of interest if we are studying, for example, how self-oriented a speaker is. The number of times x that the word is uttered in over a particular time period (here, one day) can be modeled by a Poisson distribution ($x = 0, 1, 2, \ldots$):

$$f(x \mid \theta) = \frac{\exp(-\theta)\theta^x}{x!} \text{ for } x = 0, 1, 2, \ldots \tag{G.6}$$

where the rate θ is unknown, and the numbers of utterances of the target word on each day are independent given θ.

As an aside: Given n independent observations of a Poisson random variable with rate parameter θ, the maximum-likelihood estimator (MLE) for θ turns out to be $\hat{\theta} = \frac{\sum_{i=1}^{n} x_i}{n}$. When we are talking about a particular sample of data, the maximum-likelihood estimate is computed using the formula for the estimator, $\frac{\sum_{i=1}^{n} x_i}{n}$, and is represented as $\bar{x}$.

We are told that the prior mean of θ is 100 and prior variance for θ is 225. This information is based on the results of previous studies on the topic. We will use the $Gamma(a, b)$ density (see previous question) as a prior for θ because this is a conjugate prior to the Poisson distribution.

a) First, visualize the prior, a Gamma density prior for θ based on the above information.

[Hint: we know that for a Gamma density with parameters a, b, the mean is $\frac{a}{b}$ and the variance is $\frac{a}{b^2}$. Since we are given values for the mean and variance, we can solve for a, b, which gives us the Gamma density.]

b) Next, derive the posterior distribution of the parameter θ up to proportionality, and write down the posterior distribution in terms of the parameters of a Gamma distribution.

- Practical application

Suppose we know that the number of "I" utterances from a particular individual is $115, 97, 79, 131$. Use the result you derived above to obtain the posterior distribution. In other words, write down the parameters of the Gamma distribution (call them a^*, b^*) representing the posterior distribution of θ.

Plot the prior and the posterior distributions alongside each other.

Now suppose you get one new data point: 200. Using the posterior $Gamma(a^*, b^*)$ as your prior, write down the updated posterior (in terms of the updated parameters of the Gamma distribution) given this new data point. Add the updated posterior to the plot you made above.

G.2.6 The posterior mean is a weighted mean of the prior mean and the MLE (Poisson-Gamma conjugate case)

The number of times an event happens per unit time can be modeled using a Poisson distribution, whose PMF is:

$$f(x \mid \theta) = \frac{\exp(-\theta)\theta^x}{x!} \tag{G.7}$$

Suppose that we define a Gamma(a,b) prior for the rate parameter θ. It is a fact (see exercises above) that the posterior of the θ parameter is a $Gamma(a^*, b^*)$ distribution, where a^* and b^* are the updated parameters given the data: $\theta \sim Gamma(a^*, b^*)$.

- Prove that the posterior mean is a weighted mean of the prior mean and the maximum likelihood estimate (mean) of the Poisson-distributed data, $\bar{x} = \sum_{i=1}^{n} x/n$. Hint: the mean of a Gamma distribution is $\frac{a}{b}$. Specifically, what you have to prove is that:

$$\frac{a^*}{b^*} = \frac{a}{b} \times \frac{w_1}{w_1 + w_2} + \bar{x} \times \frac{w_2}{w_1 + w_2} \tag{G.8}$$

  where $w_1 = 1$ and $w_2 = \frac{n}{b}$.

- Given equation (G.8), make an informal argument showing that as n increases (as sample size goes up), the maximum likelihood estimate $\bar{x}$ dominates in determining the posterior mean, and when n gets smaller and smaller, the prior mean dominates in determining the posterior mean.

- Finally, given that the variance of a Gamma distribution is $\frac{a}{b^2}$, show that as n increases, the posterior variance will get smaller and smaller (the uncertainty on the posterior will go down).

G.3 Computational Bayesian data analysis

G.3.1 Check for parameter recovery in a linear model using simulated data.

Generate some simulated independent and identically distributed data with $n = 100$ data points as follows:

```
y <- rnorm(100, mean = 500, sd = 50)
```

Next, fit a simple linear model with a normal likelihood:

$$y_n \sim Normal(\mu, \sigma) \tag{G.9}$$

Specify the following priors:

$$\begin{aligned} \mu &\sim Uniform(0, 60000) \\ \sigma &\sim Uniform(0, 2000) \end{aligned} \tag{G.10}$$

Generate posterior distributions of the parameters and check that the true values of the parameters $\mu = 500, \sigma = 50$ are recovered by the model. What this means is that you should check whether these true values lie within the range of the posterior distributions of the two parameters. This is a good sanity check for finding out whether a model can in principle recover the true parameter values correctly.

G.3.2 A simple linear model.

a. Fit the model `fit_press` with just a few iterations, say 50 iterations (set warmup to the default of 25, and use four chains). Does the model converge?

b. Using normal distributions, choose priors that better represent **your** assumptions/beliefs about finger tapping times. To think about a reasonable set of priors for μ and σ, you should come up with your own subjective assessment about what you think a reasonable range of values can be for μ and how much variability might happen. There is no correct answer here, we'll discuss priors in depth in chapter E. Fit this model to the data. Do the posterior distributions change?

G.3.3 Revisiting the button-pressing example with different priors.

a. Can you come up with very informative priors that influence the posterior in a noticeable way (use normal distributions for priors, not

uniform priors)? Again, there are no correct answers here; you may have to try several different priors before you can noticeably influence the posterior.

b. Generate and plot prior predictive distributions based on this prior and plot them.

c. Generate posterior predictive distributions based on this prior and plot them.

G.3.4 Posterior predictive checks with a log-normal model.

a. For the log-normal model `fit_press_ln`, change the prior of σ so that it is a log-normal distribution with location (μ) of -2 and scale (σ) of 0.5. What does such a prior imply about your belief regarding button-pressing times in milliseconds? Is it a good prior? Generate and plot prior predictive distributions. Do the new estimates change compared to earlier models when you fit the model?
b. For the log-normal model, what is the mean (rather than median) time that takes to press the space bar, what is the standard deviation of the finger tapping times in milliseconds?

G.3.5 A skew normal distribution.

Would it make sense to use a "skew normal distribution" instead of the log-normal? The skew normal distribution has three parameters: location ξ (this is the lower-case version of the Greek letter Ξ, pronounced "chi", with the "ch" pronounced like the "ch" in "Bach"), scale ω (omega), and shape α. The distribution is right skewed if $\alpha > 0$, is left skewed if $\alpha < 0$, and is identical to the regular normal distribution if $\alpha = 0$. For fitting this in `brms`, one needs to change `family` and set it to `skew_normal()`, and add a prior of `class = alpha` (location remains `class = Intercept` and scale, `class = sigma`).

a. Fit this model with a prior that assigns approximately 95% of the prior probability of `alpha` to be between 0 and 10.
b. Generate posterior predictive distributions and compare the posterior distribution of summary statistics of the skew normal with the normal and log-normal.

G.4 Bayesian regression models

G.4.1 A simple linear regression: Power posing and testosterone.

Load the following data set:

```
data("df_powerpose")
head(df_powerpose)
```

```
##   id hptreat female age testm1 testm2
## 2 29    High   Male  19   38.7   62.4
## 3 30     Low Female  20   32.8   29.2
## 4 31    High Female  20   32.3   27.5
## 5 32     Low Female  18   18.0   28.7
## 7 34     Low Female  21   73.6   44.7
## 8 35    High Female  20   80.7  105.5
```

The data set, which was originally published in Carney, Cuddy, and Yap (2010) but released in modified form by Fosse (2016), shows the testosterone levels of 39 different individuals, before and after treatment, where treatment refers to each individual being assigned to a high power pose or a low power pose. In the original paper by Carney, Cuddy, and Yap (2010), the unit given for testosterone measurement (estimated from saliva samples) was picograms per milliliter (pg/ml). One picogram per milliliter is 0.001 nanogram per milliliter (ng/ml).

The research hypothesis is that on average, assigning a subject a high power pose vs. a low power pose will lead to higher testosterone levels after treatment. Assuming that you know nothing about typical ranges of testosterone using salivary measurement, you can use the default priors in `brms` for the target parameter(s).

Investigate this claim using a linear model and the default priors of `brms`. You'll need to estimate the effect of a new variable that encodes the change in testosterone.

G.4.2 Another linear regression model: Revisiting attentional load effect on pupil size.

Here, we revisit the analysis shown in the chapter, on how attentional load affects pupil size.

a. Our priors for this experiment were quite arbitrary. How do the prior predictive distributions look like? Do they make sense?
b. Is our posterior distribution sensitive to the priors that we selected? Perform a sensitivity analysis to find out whether the posterior is affected by our choice of prior for the σ.
c. Our data set includes also a column that indicates the trial number. Could it be that trial has also an effect on the pupil size? As in `lm()`, we indicate another main effect with a `+` sign. How would you communicate the new results?

G.4.3 Log-normal model: Revisiting the effect of trial on finger tapping times.

We continue considering the effect of trial on finger tapping times.

a. Estimate the slowdown in milliseconds between the last two times the subject pressed the space bar in the experiment.
b. How would you change your model (keeping the log-normal likelihood) so that it includes centered log-transformed trial numbers or square-root-transformed trial numbers (instead of centered trial numbers)? Does the effect in milliseconds change?

G.4.4 Logistic regression: Revisiting the effect of set size on free recall.

Our data set includes also a column coded as `tested` that indicates the position of the queued word. (In Figure 4.9 `tested` would be 3). Could it be that position also has an effect on recall accuracy? How would you incorporate this in the model? (We indicate another main effect with a `+` sign).

G.4.5 Red is the sexiest color.

Load the following data set:

```
data("df_red")
head(df_red)
```

```
##    risk age red pink redorpink
## 8     0  19   0    0         0
## 9     0  25   0    0         0
## 10    0  20   0    0         0
## 11    0  20   0    0         0
## 14    0  20   0    0         0
## 15    0  18   0    0         0
```

The data set is from a study (Beall and Tracy 2013) that contains information about the color of the clothing worn (red, pink, or red or pink) when the subject (female) is at risk of becoming pregnant (is ovulating, self-reported). The broader issue being investigated is whether women wear red more often when they are ovulating (in order to attract a mate). Using logistic regressions, fit three different models to investigate whether being ovulating increases the probability of wearing (a) red, (b) pink, or (c) either pink or red. Use priors that are reasonable (in your opinion).

G.5 Bayesian hierarchical models

G.5.1 A hierarchical model (normal likelihood) of cognitive load on pupil size.

As in section 4.1, we focus on the effect of cognitive load on pupil size, but this time we look at all the subjects of Wahn et al. (2016):

```
data("df_pupil_complete")
df_pupil_complete
```

```
## # A tibble: 2,228 x 4
##    subj trial  load p_size
##   <int> <int> <int>  <dbl>
## 1   701     1     2  1021.
## 2   701     2     1   951.
## 3   701     3     5  1064.
## # i 2,225 more rows
```

You should be able to now fit a "maximal" model (correlated varying intercept and slopes for subjects) assuming a normal likelihood. Base your priors in the priors discussed in section 4.1.

(a) Examine the effect of load on pupil size, and the average pupil size. What do you conclude?
(b) Do a sensitivity analysis for the prior on the intercept (α). What is the estimate of the effect (β) under different priors?
(c) Is the effect of load consistent across subjects? Investigate this visually.

G.5.2 Are subject relatives easier to process than object relatives (log-normal likelihood)?

We begin with a classic question from the psycholinguistics literature: Are subject relatives easier to process than object relatives? The data come from Experiment 1 in a paper by Grodner and Gibson (2005).

Scientific question: Is there a subject relative advantage in reading?

Grodner and Gibson (2005) investigate an old claim in psycholinguistics that object relative clause (ORC) sentences are more difficult to process than subject relative clause (SRC) sentences. One explanation for this predicted difference is that the distance between the relative clause verb (*sent* in the example below) and the head noun phrase of the relative clause (*reporter* in the example below) is longer in ORC vs. SRC. Examples are shown below. The relative clause is shown in square brackets.

(1a) The *reporter* [who the photographer *sent* to the editor] was hoping for a good story. (ORC)

(1b) The *reporter* [who *sent* the photographer to the editor] was hoping for a good story. (SRC)

The underlying explanation has to do with memory processes: Shorter linguistic dependencies are easier to process due to either reduced interference or decay, or both. For implemented computational models that spell this point out, see Lewis and Vasishth (2005) and Engelmann, Jäger, and Vasishth (2020).

In the Grodner and Gibson data, the dependent measure is reading time at the relative clause verb, (e.g., *sent*) of different sentences with either ORC or SRC. The dependent variable is in milliseconds and was measured in a self-paced reading task. Self-paced reading is a task where subjects read a sentence or a short text word-by-word or phrase-by-phrase, pressing a button to get each word or phrase displayed; the preceding word disappears every time the button is pressed. In E.1, we provide a more detailed explanation of this experimental method.

For this experiment, we are expecting longer reading times at the relative clause verbs of ORC sentences in comparison to the relative clause verb of SRC sentences.

```
data("df_gg05_rc")
df_gg05_rc
```

```
## # A tibble: 672 x 7
##    subj  item condition    RT residRT qcorrect experiment
```

```
##   <int> <int> <chr>     <int>   <dbl>    <int> <chr>
## 1     1     1 objgap      320   -21.4        0 tedrg3
## 2     1     2 subjgap     424    74.7        1 tedrg2
## 3     1     3 objgap      309   -40.3        0 tedrg3
## # i 669 more rows
```

You should use a sum coding for the predictors. Here, object relative clauses (`"objgaps"`) are coded +1/2, subject relative clauses −1/2.

```
df_gg05_rc <- df_gg05_rc %>%
  mutate(c_cond = if_else(condition == "objgap", 1/2, -1/2))
```

You should be able to now fit a "maximal" model (correlated varying intercept and slopes for subjects and for items) assuming a log-normal likelihood.

(a) Examine the effect of relative clause attachment site (the predictor `c_cond`) on reading times `RT` (β).
(b) Estimate the median difference between relative clause attachment sites in milliseconds, and report the mean and 95% CI.
(c) Do a sensitivity analysis. What is the estimate of the effect (β) under different priors? What is the difference in milliseconds between conditions under different priors?

G.5.3 Relative clause processing in Mandarin Chinese

Load the following two data sets:

```
data("df_gibsonwu")
data("df_gibsonwu2")
```

The data are taken from two experiments that investigate (inter alia) the effect of relative clause type on reading time in Chinese. The data are from Gibson and Wu (2013) and Vasishth et al. (2013) respectively. The second data set is a direct replication attempt of the Gibson and Wu (2013) experiment.

Chinese relative clauses are interesting theoretically because they are prenominal: the relative clause appears before the head noun. For example, the English relative clauses shown above would appear in the following order in Mandarin. The square brackets mark the relative clause, and REL refers to the Chinese equivalent of the English relative pronoun *who*.

(2a) [The photographer *sent* to the editor] REL the *reporter* was hoping for a good story. (ORC)

(2b) [*sent* the photographer to the editor] REL the *reporter* who was hoping for a good story. (SRC)

As discussed in Gibson and Wu (2013), the consequence of Chinese relative clauses being prenominal is that the distance between the verb in relative clause and the head noun is larger in subject relatives than object relatives. Hsiao and Gibson (2003) were the first to suggest that the larger distance in subject relatives leads to longer reading time at the head noun. Under this view, the prediction is that subject relatives are harder to process than object relatives. If this is true, this is interesting and surprising because in most other languages that have been studied, subject relatives are easier to process than object relatives; so Chinese will be a very unusual exception cross-linguistically.

The data provided are for the critical region (the head noun; here, *reporter*). The experiment method is self-paced reading, so we have reading times in milliseconds. The second data set is a direct replication attempt of the first data set, which is from Gibson and Wu (2013).

The research hypothesis is whether the difference in reading times between object and subject relative clauses is negative. For the first data set (`df_gibsonwu`), investigate this question by fitting two "maximal" hierarchical models (correlated varying intercept and slopes for subjects and items). The dependent variable in both models is the raw reading time in milliseconds. The first model should use the normal likelihood in the model; the second model should use the log-normal likelihood. In both models, use ± 0.5 sum coding to model the effect of relative clause type. You will need to decide on appropriate priors for the various parameters.

(a) Plot the posterior predictive distributions from the two models. What is the difference in the posterior predictive distributions of the two models; and why is there a difference?
(b) Examine the posterior distributions of the effect estimates (in milliseconds) in the two models. Why are these different?
(c) Given the posterior predictive distributions you plotted above, why is the log-normal likelihood model better for carrying out inference and hypothesis testing?

Next, work out a normal approximation of the log-normal model's posterior distribution for the relative clause effect that you obtained from the above data analysis. Then use that normal approximation as an informative prior for the slope parameter when fitting a hierarchical model to the second data set. This is an example of incrementally building up knowledge by successively using a previous study's posterior as a prior for the next study; this is essentially

equivalent to pooling both data sets (check that pooling the data and using a Normal(0,1) prior for the effect of interest, with a log-normal likelihood, gives you approximately the same posterior as the informative-prior model fit above).

G.5.4 Agreement attraction in comprehension

Load the following data:

```
data("df_dillonE1")
dillonE1 <- df_dillonE1
head(dillonE1)
```

```
##         subj       item   rt int     expt
## 49 dillonE11 dillonE119 2918 low dillonE1
## 56 dillonE11 dillonE119 1338 low dillonE1
## 63 dillonE11 dillonE119  424 low dillonE1
## 70 dillonE11 dillonE119  186 low dillonE1
## 77 dillonE11 dillonE119  195 low dillonE1
## 84 dillonE11 dillonE119 1218 low dillonE1
```

The data are taken from an experiment that investigate (inter alia) the effect of number similarity between a noun and the auxiliary verb in sentences like the following. There are two levels to a factor called Int(erference): low and high.

(3a) low: The key to the cabinet *are* on the table (3b) high: The key to the *cabinets are* on the table

Here, in (3b), the auxiliary verb *are* is predicted to be read faster than in (3a), because the plural marking on the noun *cabinets* leads the reader to think that the sentence is grammatical. (Both sentences are ungrammatical.) This phenomenon, where the high condition is read faster than the low condition, is called **agreement attraction**.

The data provided are for the critical region (the auxiliary verb *are*). The experiment method is eye-tracking; we have total reading times in milliseconds.

The research question is whether the difference in reading times between high and low conditions is negative.

- First, using a log-normal likelihood, fit a hierarchical model with correlated varying intercept and slopes for subjects and items. You will need to decide on the priors for the model.
- By simply looking at the posterior distribution of the slope parameter β,

what would you conclude about the theoretical claim relating to agreement attraction?

G.5.5 Attentional blink (Bernoulli likelihood)

The attentional blink (AB; first described by Raymond, Shapiro, and Arnell 1992; though it has been noticed before e.g., Broadbent and Broadbent 1987) refers to a temporary reduction in the accuracy of detecting a *probe* (e.g., a letter "X") presented closely after a *target* that has been detected (e.g., a white letter). We will focus on the experimental condition of Experiment 2 of Raymond, Shapiro, and Arnell (1992). Subjects are presented with letters in rapid serial visual presentation (RSVP) at the center of the screen at a constant rate and are required to identify the only white letter (target) in the stream of black letters, and then to report whether the letter X (probe) occurred in the subsequent letter stream. The AB is defined as having occurred when the target is reported correctly but the report of the probe is inaccurate at a short *lag* or *target-probe* interval.

The data set `df_ab` is a subset of the data of this paradigm from a replication conducted by Grassi et al. (2021). In this subset, the probe was always present and the target was correctly identified. We want to find out how the lag affects the accuracy of the identification of the probe.

```
data("df_ab")
df_ab
```

```
## # A tibble: 2,101 x 4
##    subj probe_correct trial   lag
##   <int>         <int> <int> <int>
## 1     1             0     2     5
## 2     1             1     4     4
## 3     1             1     8     6
## # i 2,098 more rows
```

Fit a logistic regression assuming a linear relationship between `lag` and accuracy (`probe_correct`). Assume a hierarchical structure with correlated varying intercept and slopes for subjects. You will need to decide on the priors for this model.

(a) How is the accuracy of the probe identification affected by the lag? Estimate this in log-odds and percentages.
(b) Is the linear relationship justified? Use posterior predictive checks to verify this.

(c) Can you think about a better relationship between lag and accuracy? Fit a new model and use posterior predictive checks to verify if the fit improved.

G.5.6 Is there a Stroop effect in accuracy?

Instead of the response times of the correct answers, we want to find out whether accuracy also changes by condition in the Stroop task. Fit the Stroop data with a hierarchical logistic regression (i.e., a Bernoulli likelihood with a logit link). Use the complete data set, `df_stroop_complete` which also includes incorrect answers, and subset it selecting the first 50 subjects.

(a) Fit the model.
(b) Report the Stroop effect in log-odds and accuracy.

G.5.7 Distributional regression for the Stroop effect.

We will relax some of the assumptions of the model of Stroop presented in section 5.3. We will no longer assume that all subjects share the same variance component, and, in addition, we'll investigate whether the experimental manipulation affects the scale of the response times. A reasonable hypothesis could be that the incongruent condition is noisier than the congruent one.

Assume the following likelihood, and fit the model with sensible priors (recall that our initial prior for β wasn't reasonable). (Priors for all the `sigma` parameters require us to set `dpar = sigma`).

$$\begin{aligned} rt_n &\sim LogNormal(\alpha + u_{subj[n],1} + c\_cond_n \cdot (\beta + u_{subj[n],2}), \sigma_n) \\ \sigma_n &= \exp(\sigma_\alpha + \sigma_{u_{subj[n],1}} + c\_cond \cdot (\sigma_\beta + \sigma_{u_{subj[n],2}})) \end{aligned} \tag{G.11}$$

In this likelihood σ_n has both population- and group-level parameters: σ_α and σ_β are the intercept and slope of the population level effects repectively, and $\sigma_{u_{subj[n],1}}$ and $\sigma_{u_{subj[n],2}}$ are the intercept and slope of the group-level effects.

(a) Is our hypothesis reasonable in light of the results?
(b) Why is the intercept for the scale negative?
(c) What's the posterior estimate of the scale for congruent and incongruent conditions?

G.5.8 The grammaticality illusion

Load the following two data sets:

```
data("df_english")
english <- df_english
data("df_dutch")
dutch <- df_dutch
```

In an offline accuracy rating study on English double center-embedding constructions, Gibson and Thomas (1999) found that grammatical constructions (e.g., example 4a below) were no less acceptable than ungrammatical constructions (e.g., example 4b) where a middle verb phrase (e.g., *was cleaning every week*) was missing.

(4a) The apartment that the maid who the service had sent over was cleaning every week was well decorated.

(4b) *The apartment that the maid who the service had sent over — was well decorated

Based on these results from English, Gibson and Thomas (1999) proposed that working-memory overload leads the comprehender to forget the prediction of the upcoming verb phrase (VP), which reduces working-memory load. This came to be known as the *VP-forgetting hypothesis*. The prediction is that in the word immediately following the final verb, the grammatical condition (which is coded as +1 in the data frames) should be harder to read than the ungrammatical condition (which is coded as -1).

The design shown above is set up to test this hypothesis using self-paced reading for English (Vasishth et al. 2011), and for Dutch (Frank, Trompenaars, and Vasishth 2015). The data provided are for the critical region (the noun phrase, labeled NP1, following the final verb); this is the region for which the theory predicts differences between the two conditions. We have reading times in log milliseconds.

(a) First, fit a linear model with a full hierarchical structure by subjects and by items for the English data. Because we have log milliseconds data, we can simply use the normal likelihood (not the log-normal). What scale will be the parameters be in, milliseconds or log milliseconds?

(b) Second, using the posterior for the effect of interest from the English data, derive a prior distribution for the effect in the Dutch data. Then fit two linear mixed models: (i) one model with relatively uninformative priors for β (for example, $Normal(0, 1)$), and (ii) one model with the prior for β you derived from the English data. Do the posterior dis-

tributions of the Dutch data's effect show any important differences given the two priors? If yes, why; if not, why not?

(c) Finally, just by looking at the English and Dutch posteriors, what can we say about the VP-forgetting hypothesis? Are the posteriors of the effect from these two languages consistent with the hypothesis?

G.6 Contrast coding

G.6.1 Contrast coding for a four-condition design

Load the following data. These data are from Experiment 1 in a set of reading studies on Persian (Safavi, Husain, and Vasishth 2016). This is a self-paced reading study on particle-verb constructions, with a 2×2 design: distance (short, long) and predictability (predictable, unpredictable). The data are from a critical region in the sentence. All the data from the Safavi, Husain, and Vasishth (2016) paper are available from https://github.com/vasishth/SafaviEtAl2016.

```
library(bcogsci)
data("df_persianE1")
dat1 <- df_persianE1
head(dat1)
```

```
##     subj item   rt distance   predability
## 60     4    6  568    short   predictable
## 94     4   17  517     long unpredictable
## 146    4   22  675    short   predictable
## 185    4    5  575     long unpredictable
## 215    4    3  581     long   predictable
## 285    4    7 1171     long   predictable
```

The four conditions are:

- Distance=short and Predictability=unpredictable
- Distance=short and Predictability=predictable
- Distance=long and Predictability=unpredictable
- Distance=long and Predictability=predictable

The researcher wants to do the following sets of comparisons between condition means:

Compare the condition labeled Distance=short and Predictability=unpredictable with each of the following conditions:

- Distance=short and Predictability=predictable
- Distance=long and Predictability=unpredictable
- Distance=long and Predictability=predictable

Questions:

- Which contrast coding is needed for such a comparison?
- First, define the relevant contrast coding. Hint: You can do it by creating a condition column labeled a,b,c,d and then use a built-in contrast coding function.
- Then, use the `hypr` library function to confirm that your contrast coding actually does the comparison you need.
- Fit a simple linear model with the above contrast coding and display the slopes, which constitute the relevant comparisons.
- Now, compute each of the four conditions' means and check that the slopes from the linear model correspond to the relevant differences between means that you obtained from the data.

G.6.2 Helmert coding for a six-condition design.

This data-set is from a psycholinguistics study, and although we explain the theoretical background below, one does not need to deeply understand the research questions to be able to define the contrasts.

Load the following data:

```
library(bcogsci)
data("df_polarity")
head(df_polarity)
```

```
##   subject item condition times value
## 1       1    6         f   SFD   328
## 2       1   24         f   SFD   206
## 3       1   35         e   SFD   315
## 4       1   17         e   SFD   265
## 5       1   34         d   SFD   252
## 6       1    7         a   SFD   156
```

The data come from an eyetracking study in German reported in Vasishth et al. (2008). The experiment is a reading study involving six conditions. The sentences are in English, but the original design was involved German sentences. In German, the word *durchaus* (certainly) is a positive polarity item: in the

constructions used in this experiment, *durchaus* cannot have a c-commanding element that is a negative polarity item licensor. By contrast, the German negative polarity item *jemals* (ever) is a negative polarity item: in the constructions used in this experiment, *jemals* must have a c-commanding element that is a negative polarity item licensor.

Here are the conditions:

- Negative polarity items
  - (a) Grammatical: No man who had a beard was ever thrifty.
  - (b) Ungrammatical (Intrusive NPI licensor): A man who had no beard was ever thrifty.
  - (c) Ungrammatical: A man who had a beard was ever thrifty.
- Positive polarity items
  - (d) Ungrammatical: No man who had a beard was certainly thrifty.
  - (e) Grammatical (Intrusive NPI licensor): A man who had no beard was certainly thrifty.
  - (f) Grammatical: A man who had a beard was certainly thrifty.

We will focus only on re-reading time in this data set. Subset the data so that we only have re-reading times in the data frame:

```
dat2 <- subset(df_polarity, times == "RRT")
head(dat2)
```

```
##      subject item condition times value
## 6365       1   20         b   RRT   240
## 6366       1    3         c   RRT  1866
## 6367       1   13         a   RRT   530
## 6368       1   19         a   RRT   269
## 6369       1   27         c   RRT   845
## 6370       1   26         b   RRT   635
```

The comparisons we are interested in are as follows:

- What is the difference in reading time between negative polarity items and positive polarity items? In other words, we want to compare the mean of conditions (a), (b), (c) with the mean of (d), (e), (f).
- Within negative polarity items, what is the difference between grammatical and ungrammatical conditions? In other words, we want to compare condition (a) with the average of (b) and (c).
- Within negative polarity items, what is the difference between the two ungrammatical conditions? Here, we want to compare conditions (b) and (c).

- Within positive polarity items, what is the difference between grammatical and ungrammatical conditions? Here, we want to compare condition (d) with the average of (e) and (f).
- Within positive polarity items, what is the difference between the two grammatical conditions? Here, the comparison is between (e) and (f).

Use the `hypr` package to specify the comparisons specified above, and then extract the contrast matrix. Finally, specify the contrasts to the condition column in the data frame. Fit a linear model using this contrast specification, and then check that the estimates from the model match the mean differences between the conditions being compared.

G.6.3 Number of possible comparisons in a single model.

- How many comparisons can one make in a single model when there is a single factor with four levels? Why can we not code four comparisons in a single model?
- How many comparisons can one code in a model where there are two factors, one with three levels and one with two levels?
- How about a model for a $2 \times 2 \times 3$ design?

G.7 Contrast coding with two predictor variables

G.7.1 ANOVA coding for a four-condition design.

Load the following data. These data are from Experiment 1 in a set of reading studies on Persian (Safavi, Husain, and Vasishth 2016); we encountered these data in the preceding chapter's exercises.

```
library(bcogsci)
data("df_persianE1")
dat1 <- df_persianE1
head(dat1)
```

```
##     subj item   rt distance   predability
## 60     4    6  568    short   predictable
## 94     4   17  517     long unpredictable
## 146    4   22  675    short   predictable
## 185    4    5  575     long unpredictable
## 215    4    3  581     long   predictable
## 285    4    7 1171     long   predictable
```

The four conditions are:

- Distance=short and Predictability=unpredictable
- Distance=short and Predictability=predictable
- Distance=long and Predictability=unpredictable
- Distance=long and Predictability=predictable

For the data given above, define an ANOVA-style contrast coding, and compute main effects and interactions. Check with `hypr` what the estimated comparisons are with an ANOVA coding.

G.7.2 ANOVA and nested comparisons in a $2 \times 2 \times 2$ design

Load the following data set. This is a $2 \times 2 \times 2$ design from Jäger et al. (2020), with the factors Grammaticality (grammatical vs. ungrammatical), Dependency (Agreement vs. Reflexives), and Interference (Interference vs. no interference). The experiment is a replication attempt of Experiment 1 reported in Dillon et al. (2013).

```
library(bcogsci)
data("df_dillonrep")
```

- The grammatical conditions are a,b,e,f. The rest of the conditions are ungrammatical.
- The agreement conditions are a,b,c,d. The other conditions are reflexives.
- The interference conditions are a,d,e,h, and the others are the no-interference conditions.

The dependent measure of interest is TFT (total fixation time, in milliseconds).

Using a linear model, do a main effects and interactions ANOVA contrast coding, and obtain an estimate of the main effects of Grammaticality, Dependency, and Interference, and all interactions. You may find it easier to code the contrasts coding the main effects as +1, -1, using `ifelse()` in R to code vectors corresponding to each main effect. This will make the specification of the interactions easy.

The researchers had a further research hypothesis: in ungrammatical sentences only, agreement would show an interference effect but reflexives would not. In grammatical sentences, both agreement and reflexives are expected to show interference effects. This kind of research question can be answered with nested contrast coding.

To carry out the relevant nested contrasts, define contrasts that estimate the effects of

- grammaticality
- dependency type
- the interaction between grammaticality and dependency type
- reflexives interference within grammatical conditions
- agreement interference within grammatical conditions
- reflexives interference within ungrammatical conditions
- agreement interference within ungrammatical conditions

Do the estimates match expectations? Check this by computing the condition means and checking that the estimates from the models match the relevant differences between conditions or clusters of conditions.

G.8 Introduction to the probabilistic programming language Stan

G.8.1 A very simple model.

In this exercise we revisit the model from 3.2.1. Assume the following:

1. There is a true underlying time, μ, that the subject needs to press the space bar.
2. There is some noise in this process.
3. The noise is normally distributed (this assumption is questionable given that response times are generally skewed; we fix this assumption later).

That is the likelihood for each observation n will be:

$$t_n \sim Normal(\mu, \sigma) \tag{G.12}$$

a. Decide on appropriate priors and fit this model in Stan. Data can be found in `df_spacebar`.
b. Change the likelihood to a log-normal distribution and change the priors. Fit the model in Stan.

G.8.2 Incorrect Stan model.

We want to fit both response times and accuracy with the same model. We simulate the data as follows:

```
N <- 500
df_sim <- tibble(rt = rlnorm(N, mean = 6, sd = .5),
                 correct = rbern(N, prob = .85))
```

We build the following model:

```
data {
  int<lower = 1> N;
  vector[N] rt;
  array[N] int correct;
}
parameters {
  real<lower = 0> sigma;
  real theta;
}
model {
  target += normal_lpdf(mu | 0, 20);
  target += lognormal_lpdf(sigma | 3, 1)
  for(n in 1:N)
    target += lognormal_lpdf(rt[n] | mu, sigma);
    target += bernoulli_lpdf(correct[n] | theta);
}
```

Why does this model not work?

```
ls_sim <- list(rt = df_sim$rt,
               correct = df_sim$correct)
incorrect <- system.file("stan_models",
                         "incorrect.stan",
                         package = "bcogsci")
fit_sim <- stan(incorrect, data = ls_sim)
```

```
## Error in stanc(file = file, model_code = model_code, model_name = model_name, : 0
## Syntax error in 'string', line 13, column 2 to column 5, parsing error:
##    -------------------------------------------------
##     11:    target += normal_lpdf(mu | 0, 20);
##     12:    target += lognormal_lpdf(sigma | 3, 1)
##     13:    for(n in 1:N)
##            ^
##     14:      target += lognormal_lpdf(rt[n] | mu, sigma);
##     15:      target += bernoulli_lpdf(correct[n] | theta);
##    -------------------------------------------------
```

```
##
## Unexpected input after the conclusion of a valid expression.
## You may be missing a "," between expressions, an operator, or a terminating "}", ")", "]", or ";".
```

Try to make it run. (Hint: There are several problems.)

G.8.3 Using Stan documentation.

Edit the simple example with Stan from section 8.2, and replace the normal distribution with a skew normal distribution. (Don't forget to add a prior to the new parameter, and check the Stan documentation or a statistics textbook for more information about the distribution).

Fit the following data:

```
Y <- rnorm(1000, mean = 3, sd = 10)
```

Does the estimate of the new parameter make sense?

G.8.4 The probit link function as an alternative to the logit function.

The probit link function is the inverse of the CDF of the standard normal distribution ($Normal(0, 1)$). Since the CDF of the standard normal is usually written using the Greek letter Φ (Phi), the probit function is written as its inverse, Φ^{-1}. Refit the model presented in 8.4.3 changing the logit link function for the probit link (that is transforming the regression to a constrained space using `Phi()` in Stan).

You will probably see the following as the model runs; this is because the probit link is less numerically stable (i.e., under- and overflows) than the logit link in Stan. Don't worry, it is good enough for this exercise.

```
Rejecting initial value:
Log probability evaluates to log(0), i.e. negative infinity.
Stan can't start sampling from this initial value.
```

a. Do the results of the coefficients α and β change?
b. Do the results in probability space change?

G.8.5 Examining the position of the queued word on recall.

Refit the model presented in section 8.4.3 and examine whether set size, trial effects, the position of the queued word (`tested` in the data set), and their interaction affect free recall. (Tip: You can do this exercise without changing the Stan code.).

How does the accuracy change from position one to position two?

G.8.6 The conjunction fallacy.

Paolacci, Chandler, and Ipeirotis (2010) examined whether the results of some classic experiments differ between a university pool population and subjects recruited from Mechanical Turk. We'll examine whether the results of the conjunction fallacy experiment (or Linda problem: Tversky and Kahneman 1983) are replicated for both groups.

```
data("df_fallacy")
df_fallacy
```

```
## # A tibble: 268 x 2
##   source answer
##   <chr>   <int>
## 1 mturk       1
## 2 mturk       1
## 3 mturk       1
## # i 265 more rows
```

The conjunction fallacy shows that people often fail to regard a combination of events as less probable than a single event in the combination (Tversky and Kahneman 1983):

Linda is 31 years old, single, outspoken, and very bright. She majored in philosophy. As a student, she was deeply concerned with issues of discrimination and social justice, and also participated in anti-nuclear demonstrations.

Which is more probable?

a. *Linda is a bank teller.*
b. *Linda is a bank teller and is active in the feminist movement.*

The majority of those asked chose option b even though it's less probable ($\Pr(a \wedge b) \leq \Pr(b)$. The data set is named `df_fallacy` and it indicates with `0` option "a" and with `1` option `b`. Fit a logistic regression in Stan and report:

a. The estimated overall probability of answering (b) ignoring the group.
b. The estimated overall probability of answering (b) for each group.

G.9 Hierarchical models and reparameterization

G.9.1 A log-normal model in Stan.

Refit the Stroop example from section 5.3 in Stan (`df_stroop`).

Assume the following likelihood and priors:

$$rt_n \sim LogNormal(\alpha + u_{subj[n],1} + c\_cond_n \cdot (\beta + u_{subj[n],2}), \sigma) \tag{G.13}$$

$$\begin{aligned} \alpha &\sim Normal(6, 1.5) \\ \beta &\sim Normal(0, .1) \\ \sigma &\sim Normal_+(0, 1) \end{aligned} \tag{G.14}$$

$$\begin{aligned} \tau_{u_1} &\sim Normal_+(0, 1) \\ \tau_{u_2} &\sim Normal_+(0, 1) \\ \begin{bmatrix} 1 & \rho_u \\ \rho_u & 1 \end{bmatrix} &\sim LKJcorr(2) \end{aligned} \tag{G.15}$$

G.9.2 A by-subjects and by-items hierarchical model with a log-normal likelihood.

Revisit the question "Are subject relatives easier to process than object relatives?" Fit the model from the exercise G.5.2 using Stan.

G.9.3 A hierarchical logistic regression with Stan.

Revisit the question "Is there a Stroop effect in accuracy?" Fit the model the exercise G.5.6 using Stan.

G.9.4 A distributional regression model of the effect of cloze probability on the N400.

In section 5.2.6, we saw how to fit a distributional regression model. We might want to extend this approach to Stan. Fit the EEG data to a hierarchical model with by-subject and by-items varying intercept and slopes, and in addition assume that the residual standard deviation (the scale of the normal likelihood) can vary by subject.

$$\begin{aligned} signal_n \sim Normal(&\alpha + u_{subj[n],1} + w_{item[n],1} + \\ &c\_cloze_n \cdot (\beta + u_{subj[n],2} + w_{item[n],2}), \sigma_n) \\ \sigma_n = \exp(&\alpha_\sigma + u_{\sigma_{subj[n]}}) \end{aligned} \tag{G.16}$$

$$\begin{aligned} \alpha_\alpha &\sim Normal(0, log(50)) \\ u_\sigma &\sim Normal(0, \tau_{u_\sigma}) \\ \tau_{u_\sigma} &\sim Normal_+(0, 5) \end{aligned} \tag{G.17}$$

To fit this model, take into account that `sigma` is now a vector, and it is a transformed parameter which depends on two parameters: `alpha_sigma` and the vector with `N_subj` elements `u_sigma`. In addition, `u_sigma` depends on the hyperparameter `tau_u_sigma` (τ_{u_σ}). (Using the non-centered parameterization for `u_sigma` speeds up the model fit considerably).

G.10 Custom distributions in Stan

G.10.1 Fitting a shifted log-normal distribution.

A random variable Y has a shifted log-normal distribution with shift ψ, location μ, and scale σ, if $Z = Y - \psi$ and $Z \sim LogNormal(\mu, \sigma)$.

1. Implement a `shifted_lognormal_ldpf` function in Stan with three parameters, `mu`, `sigma`, and `psi`. Tip: One can use the regular log-normal distribution and apply a change of variable. In this case the adjustment of the Jacobian would be $|\frac{d}{dY}Y - \psi| = 1$, which in log-space is conveniently zero.
2. Verify the correctness of the model by recovering the true values of (your choice) of the parameters of the model and by using simulation-based calibration. In order to use simulation-based calibration, you will need to decide on sensible priors; assume that $\psi \sim Normal_+(100, 50)$, and choose priors for μ and σ so that the prior predictive distributions are adequate for response times.

G.10.2 Fitting a Wald distribution.

The Wald distribution (or inverse Gaussian distribution) and its variants have been proposed as another useful distribution for response times (see for example Heathcote 2004).

The probability density function of the Wald distribution is the following.

$$f(x; \mu, \lambda) = \sqrt{\frac{\lambda}{2\pi x^3}} \exp\Big(- \frac{\lambda(x-\mu)^2}{2\mu^2 x} \Big) \tag{G.18}$$

1. Implement this distribution in Stan as `wald_lpdf`. In order to do this, you will need to derive the logarithm of the PDF presented above. You can adapt the code of the following R function.

```
dwald <- function(x, lambda, mu, log = FALSE) {
  log_density <- 0.5 * log(lambda / (2 * pi())) -
    1.5 * log(x) -
    0.5 * lambda * ((x - mu) / (mu * sqrt(x)))^2
  if (log == FALSE) {
    exp(log_density)
  } else {
    log_density
  }
}
```

2. Verify the correctness of the model by recovering the true values of (your choice) of the parameters of the model and by using simulation-based calibration. As with the previous exercise, you will need to decide on sensible priors by deriving prior predictive distributions that are adequate for response times.

G.11 Meta-analysis and measurement error models

G.11.1 Extracting estimates from published papers

Researchers often do not release the data that lie behind the published paper. This creates the problem that one has to figure out the estimated effect size and standard error from published statistics. This exercise gives some practice on how to do this by considering some typical cases that are encountered in repeated measures designs.

(a) Suppose that in a repeated measures reading study, the observed t-value from a paired t-test is 2.3 with degrees of freedom 23. Suppose also that the estimated standard deviation is 150 ms. What is the estimated standard error and the estimated effect size?

(b) A repeated measures reading study reports an F-score from an analysis of variance as $F(1, 34) = 6.1$, with an effect size of 25 ms. What is the estimated standard error?

G.11.2 A meta-analysis of picture-word interference data

Load the following data set:

```
data("df_buerki")
head(df_buerki)
```

```
##                   study   d    se study_id
## 1 Collina 2013 Exp.1 a  24 13.09        1
## 2 Collina 2013 Exp.1 b -25 17.00        2
## 3   Collina 2013 Exp.2  46 22.79        3
## 4     Mahon 2007 Exp.1  17 12.24        4
## 5     Mahon 2007 Exp.2  57 13.96        5
## 6    Mahon 2007 Exp. 4  17  8.01        6
```

```
df_buerki <- subset(df_buerki, se > 0.60)
```

The data are from Bürki et al. (2020). We have a summary of the effect estimates (d) and standard errors (se) of the estimates from 162 published experiments on a phenomenon called *semantic picture-word interference*. We removed an implausibly low SE in the code above, but the results don't change regardless of whether we keep them or not, because we have data from a lot of studies.

In this experimental paradigm, subjects are asked to name a picture while ignoring a distractor word (which is either related or unrelated to the picture). The word can be printed on the picture itself, or presented auditorily. The dependent measure is the response latency, or time interval between the presentation of the picture and the onset of the vocal response. Theory says that distractors that come from the same semantic category as the picture to be named lead to a slower response then when the distractor comes from a different semantic category.

Carry out a random effects meta-analysis using `brms` and display the posterior distribution of the effect, along with the posterior of the between study standard deviation.

Choose $Normal(0, 100)$ priors for the intercept and between study sd parameters. You can also try vague priors (sensitivity analysis). Examples would be:

- $Normal(0, 200)$
- $Normal(0, 400)$

G.11.3 Measurement error model for English VOT data

Load the following data:

```
data("df_VOTenglish")
head(df_VOTenglish)
```

```
##   subject meanVOT seVOT meanvdur sevdur
## 1     F01   108.1  4.56      171   11.7
## 2     F02    92.5  4.62      189   12.7
## 3     F03    82.6  3.13      171   10.0
## 4     F04    88.3  3.21      168   11.8
## 5     F05    94.6  3.67      166   15.0
## 6     F06    75.9  3.70      176   12.9
```

You are given mean voice onset time (VOT) data (with SEs) in milliseconds for English, along with mean vowel durations (with SEs) in milliseconds. Fit a measurement-error model investigating the effect of mean vowel duration on mean VOT duration. First plot the relationship between the two variables; does it look like there is an association between the two?

Then use `brms` with measurement error included in both the dependent and independent variables. Do a sensitivity analysis to check the influence of the priors on the posteriors of the relevant parameters.

G.12 Introduction to model comparison

G.13 Bayes factors

G.13.1 Is there evidence for differences in the effect of cloze probability among the subjects?

Use Bayes factor to compare the log cloze probability model that we examined in section 13.2.2 with a similar model but that incorporates the strong assumption of no difference between subjects for the effect of cloze ($\tau_{u_2} = 0$).

G.13.2 Is there evidence for the claim that English subject relative clauses are easier to process than object relative clauses?

Consider again the reading time data coming from Experiment 1 of Grodner and Gibson (2005) presented in exercise G.5.2:

```
data("df_gg05_rc")
df_gg05_rc
```

```
## # A tibble: 672 x 7
##    subj  item condition    RT residRT qcorrect experiment
##   <int> <int> <chr>     <int>   <dbl>    <int> <chr>
## 1     1     1 objgap      320   -21.4        0 tedrg3
## 2     1     2 subjgap     424    74.7        1 tedrg2
## 3     1     3 objgap      309   -40.3        0 tedrg3
## # i 669 more rows
```

As in exercise G.5.2, you should use a sum coding for the predictors. Here, object relative clauses (`"objgaps"`) are coded +1/2, and subject relative clauses as −1/2.

```
df_gg05_rc <- df_gg05_rc %>%
  mutate(c_cond = if_else(condition == "objgap", 1/2, -1/2))
```

Using the Bayes factors function shown in this chapter, quantify the evidence against the null model (no population-level reading time difference between SRC and ORC) relative to the following alternative models:

a. $\beta \sim Normal(0, 1)$
b. $\beta \sim Normal(0, 0.1)$
c. $\beta \sim Normal(0, 0.01)$
d. $\beta \sim Normal_{+}(0, 1)$
e. $\beta \sim Normal_{+}(0, 0.1)$
f. $\beta \sim Normal_{+}(0, 0.01)$

(A $Normal_{+}(.)$ prior can be set in `brms` by defining a lower boundary as 0, with the argument `lb = 0`.)

What are the Bayes factors in favor of the alternative models a-f, compared to the null model?

Now carry out a standard frequentist likelihood ratio test using the `anova()` function that is used with the `lmer()` function. The commands for doing this comparison would be:

```
m_full <- lmer(log(RT) ~ c_cond +
                 (c_cond || subj) + (c_cond || item),
               df_gg05_rc)
```

```
m_null <- lmer(log(RT) ~ 1 + (c_cond||subj) + (c_cond || item),
               df_gg05_rc)
anova(m_null, m_full)
```

How do the conclusions from the Bayes factor analyses compare with the conclusion we obtain from the frequentist model comparison?

G.13.3 In the Grodner and Gibson 2005 data, in question-response accuracies, is there evidence for the claim that sentences with subject relative clauses are easier to comprehend?

Consider the question response accuracy of the data of Experiment 1 of Grodner and Gibson (2005).

a. Compare a model that assumes that RC type affects question accuracy on the population-level and with the effect varying by-subjects and by-items with *a null model* that assumes that there is no population-level effect present.
b. Compare a model that assumes that RC type affects question accuracy on the population level and with the effect varying by-subjects and by-items with *another null model* that assumes that there is no population-level or group-level effect present, that is no by-subject or by-item effects. What's the meaning of the results of the Bayes factor analysis?

Assume that for the effect of RC on question accuracy, $\beta \sim Normal(0, 0.1)$ is a reasonable prior, and that for all the variance components, the same prior, $\tau \sim Normal_+(0, 1)$, is a reasonable prior.

G.13.4 Bayes factor and bounded parameters using Stan.

Re-fit the data of a single subject pressing a button repeatedly from 4.2 from `data("df_spacebar")`, coding the model in Stan.

Start by assuming the following likelihood and priors:

$$rt_n \sim LogNormal(\alpha + c\_trial_n \cdot \beta, \sigma) \tag{G.19}$$

$$\begin{aligned} \alpha &\sim Normal(6, 1.5) \\ \beta &\sim Normal_+(0, 0.1) \\ \sigma &\sim Normal_+(0, 1) \end{aligned} \tag{G.20}$$

Use the Bayes factor to answer the following questions:

a. Is there evidence for any effect of trial number in comparison with no effect?
b. Is there evidence for a positive effect of trial number (as the subject reads further, they slowdown) in comparison with no effect?
c. Is there evidence for a negative effect of trial number (as the subject reads further, they speedup) in comparison with no effect?
d. Is there evidence for a positive effect of trial number in comparison with a negative effect?

(Expect very large Bayes factors in this exercise.)

G.14 Cross-validation

G.14.1 Predictive accuracy of the linear and the logarithm effect of cloze probability.

Is there a difference in predictive accuracy between the model that incorporates a linear effect of cloze probability and one that incorporates log-transformed cloze probabilities?

G.14.2 Log-normal model

Use PSIS-LOO to compare a model of Stroop as the one in G.9.1 with a model that assumes no population-level effect

(a) in `brms`.
(b) in Stan.

G.14.3 Log-normal vs rec-normal model in Stan

In section 10.1, we proposed a reciprocal truncated normal distribution (rec-normal) to response times data, as an alternative to the log-normal distribution. The log-likelihood (of μ and σ) of an individual observation, RT_n, for the rec-normal distribution would be the following one.

$$\log \mathscr{L} = \log(Normal(1/RT_n|\mu, \sigma)) - 2 \cdot \log(RT_n) \tag{G.21}$$

As explained in 10.1, we obtain the log-likelihood based on all the N observations by summing the log-likelihood of individual observations.

$$\log \mathscr{L} = \sum_{n}^{N} \log(Normal(1/RT_n|\mu, \sigma)) - \sum_{n}^{N} 2 \cdot \log(RT_n) \tag{G.22}$$

Since these two models assume right-skewed data with only positive values, the question that we are interested in here is if we can really distinguish between them. Investigate this in the following way:

(a) Generate data (N = 100 and N = 1000) with a rec-normal distribution (e.g., `rt = 1 / rtnorm(N, mu, sigma, a = 0)`).
(b) Generate data (N = 100 and N = 1000) with a log-normal distribution

Fit a rec-normal and a log-normal model using Stan to each of the four data sets, and use PSIS-LOO to compare the models.

What do you conclude?

G.15 Introduction to cognitive modeling

G.16 Multinomial processing trees

G.16.1 Modeling multiple categorical responses.

a. Re-fit the model presented in section 16.1.2, adding the assumption that you have more information about the probability of giving a correct response in the task. Assume that you know that subjects' answers have around 60% accuracy. Encode this information in the priors with two different degrees of certainty. (Hint: 1. As with the Beta distribution, you can increase the pseudo-counts to increase the amount of information and reduce the "width" of the distribution; compare $\textit{Beta}(9, 1)$ with $\textit{Beta}(900, 100)$. 2. You'll need to use a column vector for the Dirichlet concentration parameters. `[.., .., ]` is a `row_vector` that can be transposed and converted into a column vector by adding the transposition symbol `'` after the right bracket.)
b. What is the difference between the multinomial and categorical parameterizations?
c. What can we learn about impaired picture naming from the models in sections 16.1.1 and 16.1.2?

G.16.2 An alternative MPT to model the picture recognition task.

Build *any* alternative tree with four parameters w, x, y, z to fit the data generated in 16.2.2. Compare the posterior distribution of the auxiliary vector `theta` (that goes in the `multinomial_lpmf()`) with the one derived in section 16.2.2.

G.16.3 A simple MPT model that incorporates phonological complexity in the picture recognition task.

Edit the Stan code `mpt_cat.stan` from `bcogsci` presented in section 16.2.3 to incorporate the fact that `f` is now a transformed parameter that depends on the trial information and two new parameters, α_f and β_f. The rest of the latent parameters do not need to vary by trial.

$$\begin{aligned} f'_j &= \alpha_f + complexity_j \cdot \beta_f \\ f_j &= logit^{-1}(f'_j) \end{aligned} \tag{G.23}$$

The inverse logit or logistic function is called `inv_logit()` in Stan. Fit the model to the data of 16.2.3 and report the posterior distributions of the latent parameters.

G.16.4 A more hierarchical MPT.

Modify the hierarchical MPT presented in section 16.2.4 so that all the parameters are affected by individual differences. Simulate data and fit it. How well can you recover the parameters? You should use the non-centered parameterization for the by-subject adjustments. (Hint: Convergence will be reached much faster if you don't assume that the adjustment parameters are correlated as in 9.1.2, but you could also assume a correlation between all (or some of) the adjustments by using the Cholesky factorization discussed in section 9.1.3.)

G.16.5 Advanced: Multinomial processing trees.

The data set `df_source_monitoring` in `bcogsci` contains data from the package `psychotools` coming from a source-monitoring experiment (Batchelder and Riefer 1990) performed by Wickelmaier and Zeileis (2018).

In this type of experiment, subjects study items from (at least) two different sources, A and B. After the presentation of the study items, subjects are required to classify each item as coming from source A, B, or as new: N (that is, a distractor). In their version of the experiment, Wickelmaier and Zeileis used two different A-B pairs: Half of the subjects had to read items either

quietly (source A = think) or aloud (source B = say). The other half had to write items down (source A = write) or read them aloud (source B = say).

- `experiment`: write-say or think-say
- `age`: Age of the respondent in years.
- `gender`: Gender of the respondent.
- `subj`: Subject id.
- `source`: Item source, a, b or n (new)
- `a`, `b`, `N`: Number of responses for each type of stimuli

Fit a multinomial processing tree following Figures G.1 and G.2 to investigate whether experiment type, age and/or gender affects the different processes assumed in the model. As in Batchelder and Riefer (1990), assume that $a = g$ (for identifiability) and that discriminability is equal for both sources ($d_1 = d_2$).

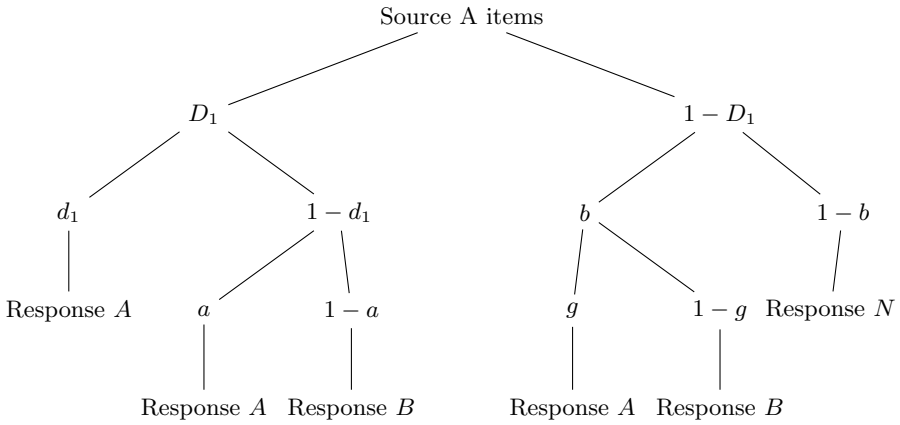

FIGURE G.1: Multinomial processing tree for the source A items from the source monitoring paradigm (Batchelder and Riefer, 1990). D_1 stands for the detectability of source A, d_1 stands for the source discriminabilities for source A items, b stands for the bias for responding "old" to a nondetected item, a stands for guessing that a detected but nondiscriminated item belongs to source A, and g stands for guessing that the item is a source A item.

Notice the following:

- The data are aggregated at the level of source, so you should use `multinomial_lpmf` for every row of the data set rather than `categorical_lpmf()`.
- In contrast to the previous example, `source` determines three different trees, this means that the parameter `theta` has to be defined in relationship to the item source.
- All the predictors are between subject, this means that only a by-intercept adjustment (for every latent process) is possible.

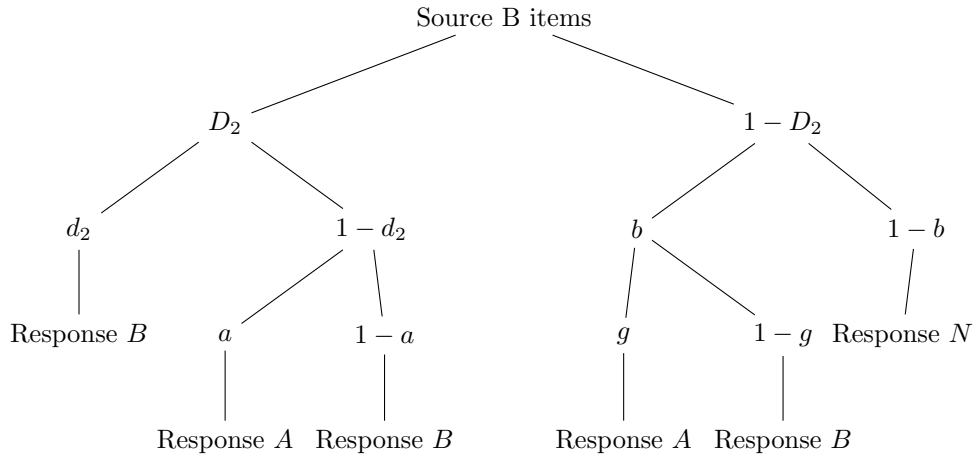

FIGURE G.2: Multinomial processing tree for the source B items from source monitoring paradigm (Batchelder and Riefer, 1990). $\boldsymbol{D_2}$ stand for the detectability of source B items, $\boldsymbol{d_2}$ stands for the source discriminabilities for source B, $\boldsymbol{b}$ stands for the bias for responding "old" to a nondetected item, $\boldsymbol{a}$ stands for guessing that a detected but nondiscriminated item belongs to Source A, and $\boldsymbol{g}$ stands for guessing that the item is a source A item.

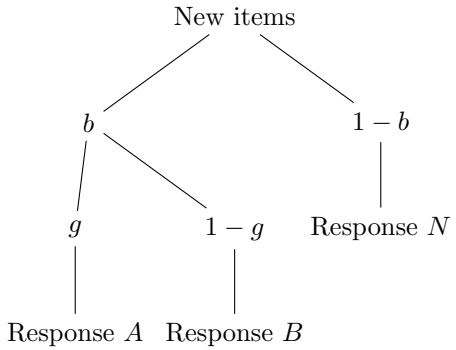

FIGURE G.3: Multinomial processing tree for the new items in the source monitoring paradigm (Batchelder and Riefer, 1990). $\boldsymbol{b}$ stands for the bias for responding "old" to a nondetected item, $\boldsymbol{a}$ stands for guessing that a detected but nondiscriminated item belongs to source A, and $\boldsymbol{g}$ stands for guessing that the item is a source A item.

If you want some basis to start with, you can have a look at the incomplete code in `source.stan`, by typing the following in R:

```
cat(readLines(system.file("stan_models",
                          "source.stan",
                          package = "bcogsci")),
    sep = "\n")
```

G.17 Mixture models

G.17.1 Changes in the true point values.

Change the true point value of `p_correct` to 0.5 and 0.1, and generate data for the non-hierarchical model. Can you recover the value of this parameter without changing the model `mixture_rtacc2.stan`? Perform posterior predictive checks.

G.17.2 RTs in schizophrenic patients and control.

Response times for schizophrenic patients in a simple visual tracking experiment show more variability than for non-schizophrenic controls; see Figure G.4. It has been argued that at least some of this extra variability arises from an attentional lapse that delays some responses. We'll use the data examined in Belin and Rubin (1990) (`df_schizophrenia` in the `bcogsci` package) analysis to investigate some potential models:

- M_1. Both schizophrenic and controls show attentional lapses, but the lapses are more common in schizophrenics. Other than that there is no difference in the latent response times and the lapses of attention.
- M_2. Only schizophrenic patients show attentional lapses. Other than that there is no difference in the latent response times.
- M_3. There are no (meaningful number of) lapses of attention in either group.

1. Fit the three models.
2. Carry out posterior predictive checks for each model; can they account for the data?
3. Carry out model comparison (with Bayes factor and cross-validation).

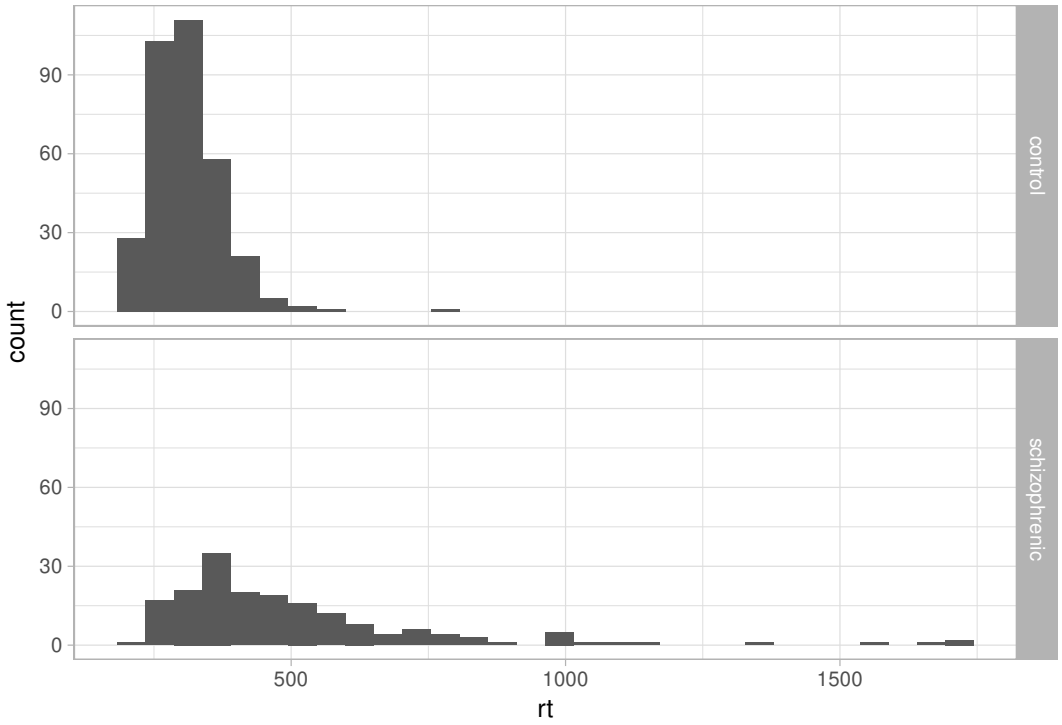

FIGURE G.4: The distribution of response times for control and schizophrenic patients in `df_schizophrenia`.

G.17.3 Advanced: Guessing bias in the model.

In the original model, it was assumed that subjects might have a bias (a preference) to one of the two answers when they were in the guessing mode. To fit this model we need to change the dependent variable and add more information; now we not only care if the participant answered correctly or not, but also which answer they gave (left or right).

- Implement a unique bias for all the subjects. Fit the new model to (a subset of) the data.
- Implement a hierarchical bias, that is there is a common bias, but every subject has its adjustment. Fit the new model to (a subset of) the data.

G.18 A simple accumulator model to account for choice response time

G.18.1 Can we recover the true point values of the parameters of a model when dealing with a contaminant distribution?

In Section 18.1.5, we fit a hierarchical model that assumed a contaminant distribution (`lnrace_h_cont.stan`) without first verifying that we can recover the true point values of its parameters if we simulate data. An important first step would be to work with a non-hierarchical version of this model.

1. Generate data of one subject as in section 18.1.2, but assume a contaminant distribution as in section 18.1.5.
2. Fit a non-hierarchical version of `lnrace_h_cont.stan` without restricting the parameter `theta_c` to be smaller than `0.1`.
3. Plot the posterior distributions of the model and verify that you can recover the true values of the parameters.

G.18.2 Can the log-normal race model account for fast errors?

Subject 13 shows fast errors for incorrect responses. This can be seen in the left side of the quantile probability plot in Figure G.5.

1. Fit a log-normal race model (with equal scales for the two accumulator) that accounts for contaminant responses.
2. Fit a variation of this model, where whether the lexicality of the string matches or not the accumulator affects its scale.
3. Visualize the fit of each model with quantile probability plots.
4. Use cross-validation to compare the models.

Notice that the models should be fit to only one subject and they should not have a hierarchical structure.

G.18.3 Accounting for response time and choice in the lexical decision task using the log-normal race model.

In Chapter 17, we modeled the data of the global motion detection task from Dutilh et al. (2011) (`df_dots`) using a mixture model. Now, we'll investigate what happens if we fit a log-normal race model to the same data. As a reminder, in this type of task, subjects see a number of random dots on the screen from which a proportion of them move in a single direction (left or right) and the rest move in random directions. The goal of the task is to estimate the overall

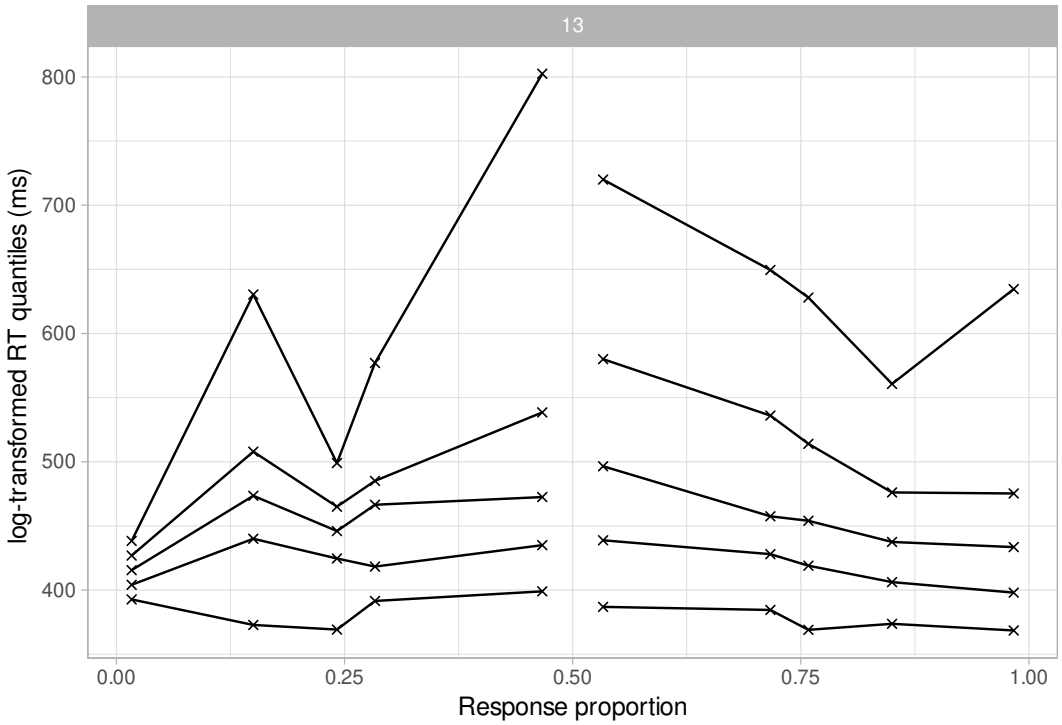

FIGURE G.5: Quantile probability plot showing 0.1, 0.3, 0.5, 0.7, and 0.9 response times quantiles plotted against proportion of incorrect responses (left) and proportion of correct responses (right) for only words of different frequency for subject number 13.

direction of the movement. In this data set, there are two difficulty levels (`diff`) and two types of instructions (`emphasis`) that focus on accuracy or speed. (More information about the data set can be found by loading the `bcogsci` package and typing `?df_dots` in the R console). For the sake of speed, we'll fit only one subject from this data set.

1. Before modeling the data, show the relationship between response times and accuracy with a quantile probability plot that shows quantiles and accuracy of easy and hard difficulty conditions.

2. Fit a non-hierarchical log-normal race model to account for how both choice and response time are affected by task difficulty and emphasis. Assume no contaminant distribution of responses.

Note that the direction of the dots is indicated with `stim`, when `stim` and `resp` match, both are `L`, left, or both are `R`, right, the accuracy, `acc`, is 1. For modeling this task with a log-normal race model, the difficulty of the task should be

coded in a way that reflects that the stimuli will be harder to detect for the relevant accumulator. One way to do it is the following:

```
df_dots_subset <- df_dots %>%
  filter(subj == 1)

df_dots_subset <- df_dots_subset %>%
  mutate(c_diff = case_when(stim == "L" & diff == "easy" ~ .5,
                            stim == "L" & diff == "hard" ~ -.5,
                            stim == "R" & diff == "easy" ~ -.5,
                            stim == "R" & diff == "hard" ~ .5,
                            ))
```

3. Expand the previous model including a contaminant distribution of responses.
4. Visualize the fit of the two previous models by doing posterior predictive checks using quantile probability plots.
5. Use cross-validation to compare the models.

G.19 The Art and Science of Prior Elicitation

G.19.1 Develop a plausible informative prior for the difference between object and subject relative clause reading times

Do a literature search on reading studies involving subject-modifying relative clause processing, and use the estimates from these published studies to work out a plausible informative prior for reading time differences at the relative clause verb. Some examples of relevant studies are Gibson et al. (2005), experiment 1 of Grodner and Gibson (2005), and Fedorenko, Gibson, and Rohde (2006); there are many others. (Note: This is an open-ended exercise with no correct answer.)

G.19.2 Extracting an informative prior from a published paper for a future study

In their experiment 1, Tabor, Galantucci, and Richardson (2004) present a self-paced reading study with a repeated measures 2×2 design. The details of the experiment are not important here, but a key estimate of interest in their reported results is the interaction of the two factors. The reported interaction

estimate is 73 ms, with a by-subjects repeated measures ANOVA F-score of 6.32. Given this information, work out the standard error (SE) of the estimate of the 73 ms. Using the estimated interaction effect and the estimated SE, derive an informative prior for a planned study that attempts to directly replicate experiment 1 in Tabor, Galantucci, and Richardson (2004). (Hint: The F-score here is the square of the corresponding observed t-value, and we know that the t-value in a one-sample t-test is computed using the formula $t = \frac{\bar{x}-0}{SE}$, where $\bar{x}$ is the estimate of the effect of interest, here, the interaction effect.)

References

Barr, Dale J., Roger P. Levy, Christoph Scheepers, and Harry J. Tily. 2013. "Random Effects Structure for Confirmatory Hypothesis Testing: Keep It Maximal." *Journal of Memory and Language* 68 (3): 255–78.

Batchelder, William H., and David M. Riefer. 1990. "Multinomial Processing Models of Source Monitoring." *Psychological Review* 97 (4): 548.

Bates, Douglas M., Reinhold Kliegl, Shravan Vasishth, and R. Harald Baayen. 2015. "Parsimonious Mixed Models." *arXiv Preptin arXiv:1506.04967*.

Bates, Douglas M., Martin Mächler, Ben Bolker, and Steve Walker. 2015. "Fitting Linear Mixed-Effects Models Using lme4." *Journal of Statistical Software* 67 (1): 1–48. https://doi.org/10.18637/jss.v067.i01.

Beall, Alec T., and Jessica L. Tracy. 2013. "Women Are More Likely to Wear Red or Pink at Peak Fertility." *Psychological Science* 24 (9): 1837–41.

Belin, T. R., and Donald B. Rubin. 1990. "Analysis of a Finite Mixture Model with Variance Components." In *Proceedings of the Social Statistics Section*, 211–15.

Betancourt, Michael J. 2018. "Towards a Principled Bayesian Workflow." https://betanalpha.github.io/assets/case_studies/principled_bayesian_workflow.html.

Betancourt, Michael J., and Mark Girolami. 2015. "Hamiltonian Monte Carlo for Hierarchical Models." *Current Trends in Bayesian Methodology with Applications* 79 (30): 2–4.

Box, George E. P. 1979. "Robustness in the Strategy of Scientific Model Building." In *Robustness in Statistics*, 201–36. Elsevier.

Brée, David S. 1975. "The Distribution of Problem-Solving Times: An Examination of the Stages Model." *British Journal of Mathematical and Statistical Psychology* 28 (2): 177–200. https://doi.org/10/cnx3q7.

Broadbent, Donald E., and Margaret H. P. Broadbent. 1987. "From Detection to Identification: Response to Multiple Targets in Rapid Serial Visual Presentation." *Perception & Psychophysics* 42 (2): 105–13. https://doi.org/10.3758/BF03210498.

Browne, William J., and David Draper. 2006. "A Comparison of Bayesian and Likelihood-Based Methods for Fitting Multilevel Models." *Bayesian Analysis* 1 (3): 473–514.

Bürki, Audrey, Shereen Elbuy, Sylvain Madec, and Shravan Vasishth. 2020. "What Did We Learn from Forty Years of Research on Semantic Interference? A Bayesian Meta-Analysis." *Journal of Memory and Language* 114. https://doi.org/10.1016/j.jml.2020.104125.

Buzsáki, György, and Kenji Mizuseki. 2014. "The Log-Dynamic Brain: How Skewed Distributions Affect Network Operations." *Nature Reviews Neuroscience* 15 (4): 264–78. https://doi.org/10.1038/nrn3687.

Carney, Dana R., Amy J. C. Cuddy, and Andy J. Yap. 2010. "Power Posing: Brief Nonverbal Displays Affect Neuroendocrine Levels and Risk Tolerance." *Psychological Science* 21 (10): 1363–8.

Chambers, Chris. 2019. *The Seven Deadly Sins of Psychology: A Manifesto for Reforming the Culture of Scientific Practice*. Princeton University Press.

DerSimonian, Rebecca, and Nan M. Laird. 1986. "Meta-Analysis in Clinical Trials." *Controlled Clinical Trials* 7 (3): 177–88. https://doi.org/https://doi.org/10.1016/0197-2456(86)90046-2.

Dillon, Brian W., Alan Mishler, Shayne Sloggett, and Colin Phillips. 2013. "Contrasting Intrusion Profiles for Agreement and Anaphora: Experimental and Modeling Evidence." *Journal of Memory and Language* 69 (2): 85–103. https://doi.org/https://doi.org/10.1016/j.jml.2013.04.003.

Dutilh, Gilles, Eric-Jan Wagenmakers, Ingmar Visser, and Han L. J. van der Maas. 2011. "A Phase Transition Model for the Speed-Accuracy Trade-Off in Response Time Experiments." *Cognitive Science* 35 (2): 211–50. https://doi.org/10.1111/j.1551-6709.2010.01147.x.

Engelmann, Felix, Lena A. Jäger, and Shravan Vasishth. 2020. "The Effect of Prominence and Cue Association in Retrieval Processes: A Computational Account." *Cognitive Science* 43 (12): e12800. https://doi.org/10.1111/cogs.12800.

Fedorenko, Evelina, Edward Gibson, and Douglas Rohde. 2006. "The Nature of Working Memory Capacity in Sentence Comprehension: Evidence Against Domain-Specific Working Memory Resources." *Journal of Memory and Language* 54 (4): 541–53. https://doi.org/https://doi.org/10.1016/j.jml.2005.12.006.

Fosse, Nathan E. 2016. "Replication Data for "Power Posing: Brief Nonverbal Displays Affect Neuroendocrine Levels and Risk Tolerance" by Carney, Cuddy, Yap (2010)." Harvard Dataverse. https://doi.org/10.7910/DVN/FMEGS6.

Frank, Stefan L., Thijs Trompenaars, and Shravan Vasishth. 2015. "Cross-Linguistic Differences in Processing Double-Embedded Relative Clauses: Working-Memory Constraints or Language Statistics?" *Cognitive Science* 40: 554–78. https://doi.org/10.1111/cogs.12247.

Freedman, Laurence S., D. Lowe, and P. Macaskill. 1984. "Stopping Rules for Clinical Trials Incorporating Clinical Opinion." *Biometrics* 40 (3): 575–86.

Gabry, Jonah, Daniel P. Simpson, Aki Vehtari, Michael J. Betancourt, and Andrew Gelman. 2019. "Visualization in Bayesian Workflow." *Journal of the Royal Statistical Society Series A: Statistics in Society* 182 (2): 389–402. https://doi.org/10.1111/rssa.12378.

Gelman, Andrew. 2006. "Prior Distributions for Variance Parameters in Hierarchical Models (Comment on Article by Browne and Draper)." *Bayesian Analysis* 1 (3): 515–34.

Gelman, Andrew, and John B. Carlin. 2014. "Beyond Power Calculations: Assessing Type S (Sign) and Type M (Magnitude) Errors." *Perspectives on Psychological Science* 9 (6): 641–51. https://doi.org/https://doi.org/10.1177/1745691614551642.

Gelman, Andrew, Daniel P. Simpson, and Michael J. Betancourt. 2017. "The Prior Can Often Only Be Understood in the Context of the Likelihood." *Entropy* 19 (10): 555. https://doi.org/10.3390/e19100555.

Gelman, Andrew, Aki Vehtari, Daniel P. Simpson, Charles C. Margossian, Bob Carpenter, Yuling Yao, Lauren Kennedy, Jonah Gabry, Paul-Christian Bürkner, and Martin Modrák. 2020. "Bayesian Workflow." *arXiv Preprint arXiv:2011.01808*.

Gentle, James E. 2007. "Matrix Algebra: Theory, Computations, and Applications in Statistics." *Springer Texts in Statistics* 10.

Gibson, Edward, Timothy Desmet, Daniel Grodner, Duane Watson, and Kara Ko. 2005. "Reading Relative Clauses in English." *Cognitive Linguistics* 16 (2): 313–53. https://doi.org/10.1515/cogl.2005.16.2.313.

Gibson, Edward, and James Thomas. 1999. "Memory Limitations and Structural Forgetting: The Perception of Complex Ungrammatical Sentences as Grammatical." *Language and Cognitive Processes* 14(3): 225–48. https://doi.org/https://doi.org/10.1080/016909699386293.

Gibson, Edward, and H.-H. Iris Wu. 2013. "Processing Chinese Relative Clauses in Context." *Language and Cognitive Processes* 28 (1-2): 125–55. https://doi.org/https://doi.org/10.1080/01690965.2010.536656.

Gordon, P. C., Randall Hendrick, and Marcus Johnson. 2001. "Memory Interference During Language Processing." *Journal of Experimental Psychology: Learning, Memory, and Cognition* 27 (6): 1411–23. https://doi.org/%20https://doi.org/10.1037/0278-7393.27.6.1411.

Grassi, Massimo, Camilla Crotti, David Giofrè, Ingrid Boedker, and Enrico Toffalini. 2021. "Two Replications of Raymond, Shapiro, and Arnell (1992), the Attentional Blink." *Behavior Research Methods* 53 (2): 656–68. https://doi.org/10.3758/s13428-020-01457-6.

Grodner, Daniel, and Edward Gibson. 2005. "Consequences of the Serial Nature of Linguistic Input." *Cognitive Science* 29: 261–90. https://doi.org/https://doi.org/10.1207/s15516709cog0000_7.

Hammerly, Christopher, Adrian Staub, and Brian W. Dillon. 2019. "The Grammaticality Asymmetry in Agreement Attraction Reflects Response Bias: Experimental and Modeling Evidence." *Cognitive Psychology* 110: 70–104. https://doi.org/https://doi.org/10.1016/j.cogpsych.2019.01.001.

Heathcote, Andrew. 2004. "Fitting Wald and ex-Wald Distributions to Response Time Data: An Example Using Functions for the S-Plus Package." *Behavior Research Methods, Instruments, & Computers* 36 (4): 678–94. https://doi.org/https://doi.org/10.3758/BF03206550.

Higgins, Julian, and Sally Green. 2008. *Cochrane Handbook for Systematics Reviews of Interventions*. New York: Wiley-Blackwell.

Hsiao, Fanny Pai-Fang, and Edward Gibson. 2003. "Processing Relative Clauses in Chinese." *Cognition* 90: 3–27. https://doi.org/https://doi.org/10.1016/S0010-0277(03)00124-0.

Jackman, Simon. 2009. *Bayesian Analysis for the Social Sciences*. Vol. 846. John Wiley & Sons.

Jäger, Lena A., Felix Engelmann, and Shravan Vasishth. 2017. "Similarity-Based Interference in Sentence Comprehension: Literature review and Bayesian meta-analysis." *Journal of Memory and Language* 94: 316–39. https://doi.org/https://doi.org/10.1016/j.jml.2017.01.004.

Jäger, Lena A., Daniela Mertzen, Julie A. Van Dyke, and Shravan Vasishth. 2020. "Interference Patterns in Subject-Verb Agreement and Reflexives Revisited: A Large-Sample Study." *Journal of Memory and Language* 111. https://doi.org/https://doi.org/10.1016/j.jml.2019.104063.

Johnson, Norman L., Samuel Kotz, and Narayanaswamy Balakrishnan. 1995. *Continuous Univariate Distributions, Volume 2*. Vol. 289. John Wiley; Sons.

Just, Marcel Adam, and Patricia A. Carpenter. 1992. "A Capacity Theory of Comprehension: Individual Differences in Working Memory." *Psychological Review* 99 (1): 122–49. https://doi.org/https://doi.org/10.1037/0033-295X.99.1.122.

Kadane, Joseph, and Lara J. Wolfson. 1998. "Experiences in Elicitation." *Journal of the Royal Statistical Society: Series D (The Statistician)* 47 (1): 3–19. https://doi.org/https://doi.org/10.1111/1467-9884.00113.

Kass, Robert E., and Joel B. Greenhouse. 1989. "Investigating Therapies of Potentially Great Benefit: ECMO: Comment: A Bayesian Perspective." *Statistical Science* 4 (4): 310–17. https://doi.org/https://www.jstor.org/stable/2245831.

Kim, Shinyoung, Hyunji Moon, Martin Modrák, and Teemu Säilynoja. 2024. *SBC: Simulation Based Calibration for Rstan/Cmdstanr Models*. https://hyunjimoon.github.io/SBC/.

Kruschke, John K. 2014. *Doing Bayesian Data Analysis: A tutorial with R, JAGS, and Stan*. Academic Press.

Kruschke, John K., and Torrin M. Liddell. 2018. "The Bayesian New Statistics: Hypothesis Testing, Estimation, Meta-Analysis, and Power Analysis from a Bayesian Perspective." *Psychonomic Bulletin & Review* 25 (1): 178–206. https://doi.org/https://doi.org/10.3758/s13423-016-1221-4.

Laird, Nan M., and James H. Ware. 1982. "Random-Effects Models for Longitudinal Data." *Biometrics*, 963–74. https://doi.org/https://doi.org/10.2307/2529876.

Landau, William Michael. 2021. "The Stantargets R Package: A Workflow Framework for Efficient Reproducible Stan-Powered Bayesian Data Analysis Pipelines." *Journal of Open Source Software* 6 (60): 3193. https://doi.org/10.21105/joss.03193.

Lee, Michael D, Amy H Criss, Berna Devezer, Christopher Donkin, Alexander Etz, Fábio P Leite, Dora Matzke, et al. 2019. "Robust Modeling in Cognitive Science." *Computational Brain & Behavior* 2: 141–53.

Levy, Dan. 2021. *Maxims for Thinking Analytically: The Wisdom of Legendary Harvard Professor Richard Zeckhauser*. Dan Levy.

Lewis, Richard L., and Shravan Vasishth. 2005. "An Activation-Based Model of Sentence Processing as Skilled Memory Retrieval." *Cognitive Science* 29: 1–45. https://doi.org/%2010.1207/s15516709cog0000_25.

Limpert, Eckhard, Werner A. Stahel, and Markus Abbt. 2001. "Log-Normal Distributions Across the Sciences: Keys and Clues." *BioScience* 51 (5): 341. https://doi.org/10.1641/0006-3568(2001)051%5B0341:LNDATS%5D2.0.CO;2.

Lindley, Dennis V. 1991. *Making Decisions*. Second. John Wiley & Sons.

Lunn, David J., Chris Jackson, David J. Spiegelhalter, Nichola G. Best, and Andrew Thomas. 2012. *The BUGS Book: A Practical Introduction to Bayesian Analysis*. Vol. 98. CRC Press.

Ly, Alexander, Alexander Etz, Maarten Marsman, and Eric-Jan Wagenmakers. 2019. "Replication Bayes Factors from Evidence Updating." *Behavior Research Methods* 51: 2498–2508.

Mahajan, Sanjoy. 2010. *Street-Fighting Mathematics: The Art of Educated Guessing and Opportunistic Problem Solving*. Cambridge, MA: The MIT Press.

———. 2014. *The Art of Insight in Science and Engineering: Mastering Complexity*. Cambridge, MA: The MIT Press.

Matuschek, Hannes, Reinhold Kliegl, Shravan Vasishth, R. Harald Baayen, and Douglas M. Bates. 2017. "Balancing Type I Error and Power in Linear Mixed Models." *Journal of Memory and Language* 94: 305–15. https://doi.org/10.1016/j.jml.2017.01.001.

Modrák, Martin, Angie H. Moon, Shinyoung Kim, Paul-Christian Bürkner, Niko Huurre, Kateřina Faltejsková, Andrew Gelman, and Aki Vehtari. 2023. "Simulation-Based Calibration Checking for Bayesian Computation: The Choice of Test Quantities Shapes Sensitivity." *Bayesian Analysis*, 1–28. https://doi.org/10.1214/23-BA1404.

Newall, Philip W. S., Taylor R. Hayes, Henrik Singmann, Leonardo Weiss-Cohen, Elliot A. Ludvig, and Lukasz Walasek. 2023. "Evaluation of the 'Take Time to Think' Safer Gambling Message: A Randomised, Online Experimental Study." *Behavioural Public Policy*, 1–18. https://doi.org/10.1017/bpp.2023.2.

Nicenboim, Bruno, and Shravan Vasishth. 2016. "Statistical methods for linguistic research: Foundational Ideas - Part II." *Language and Linguistics Compass* 10 (11): 591–613. https://doi.org/10.1111/lnc3.12207.

Nicenboim, Bruno, Shravan Vasishth, Felix Engelmann, and Katja Suckow. 2018. "Exploratory and Confirmatory Analyses in Sentence Processing: A case study of number interference in German." *Cognitive Science* 42 (S4). https://doi.org/10.1111/cogs.12589.

Normand, S. L. T. 1999. "Tutorial in Biostatistics Meta-Analysis: Formulating, Evaluating, Combining, and Reporting." *Statistics in Medicine* 18 (3): 321–59. https://doi.org/https://doi.org/10.1002/(SICI)1097-0258(19990215)18:3%3C321::AID-SIM28%3E3.0.CO;2-P.

O'Hagan, Anthony, Caitlin E. Buck, Alireza Daneshkhah, J. Richard Eiser, Paul H. Garthwaite, David J. Jenkinson, Jeremy E. Oakley, and Tim Rakow. 2006. *Uncertain Judgements: Eliciting Experts' Probabilities*. John Wiley & Sons.

Paolacci, Gabriele, Jesse Chandler, and Panagiotis G Ipeirotis. 2010. "Running Experiments on Amazon Mechanical Turk." *Judgment and Decision Making* 5 (5): 411–19. https://doi.org/10.1017/S1930297500002205.

Phillips, Colin, Matthew W. Wagers, and Ellen F. Lau. 2011. "Grammatical Illusions and Selective Fallibility in Real-Time Language Comprehension." In *Experiments at the Interfaces*, 37:147–80. Emerald Bingley, UK.

Raymond, Jane E., Kimron L. Shapiro, and Karen M. Arnell. 1992. "Temporary Suppression of Visual Processing in an RSVP Task: An Attentional Blink?" *Journal of Experimental Psychology: Human Perception and Performance* 18 (3): 849. https://doi.org/https://doi.org/10.1037/0096-1523.18.3.849.

Rayner, K. 1998. "Eye movements in reading and information processing: 20 years of research." *Psychological Bulletin* 124 (3): 372–422. https://doi.org/https://doi.org/10.1037/0033-2909.124.3.372.

Reali, Florencia, and Morten H. Christiansen. 2007. "Processing of Relative Clauses Is Made Easier by Frequency of Occurrence." *Journal of Memory and Language* 57 (1): 1–23. https://doi.org/https://doi.org/10.1016/j.jml.2006.08.014.

Rouder, Jeffrey N, Julia M. Haaf, and Joachim Vandekerckhove. 2018. "Bayesian Inference for Psychology, Part IV: Parameter Estimation and Bayes Factors." *Psychonomic Bulletin & Review* 25 (1): 102–13. https://doi.org/https://doi.org/10.3758/s13423-017-1420-7.

Safavi, Molood Sadat, Samar Husain, and Shravan Vasishth. 2016. "Dependency Resolution Difficulty Increases with Distance in Persian Separable Complex Predicates: Implications for Expectation and Memory-Based Accounts." *Frontiers in Psychology* 7 (403). https://doi.org/10.3389/fpsyg.2016.00403.

Säilynoja, Teemu, Paul-Christian Bürkner, and Aki Vehtari. 2022. "Graphical Test for Discrete Uniformity and Its Applications in Goodness-of-Fit Evaluation and Multiple Sample Comparison." *Statistics and Computing* 32 (2): 1–21. https://doi.org/https://doi.org/10.1007/s11222-022-10090-6.

Schad, Daniel J., Michael J. Betancourt, and Shravan Vasishth. 2019. "Toward a Principled Bayesian Workflow in Cognitive Science." *arXiv Preprint*. https://doi.org/10.48550/ARXIV.1904.12765.

———. 2020. "Toward a Principled Bayesian Workflow in Cognitive Science." *Psychological Methods* 26 (1): 103–26. https://doi.org/https://doi.org/10.1037/met0000275.

Schad, Daniel J., Shravan Vasishth, Sven Hohenstein, and Reinhold Kliegl. 2020. "How to Capitalize on a Priori Contrasts in Linear (Mixed) Models: A Tutorial." *Journal of Memory and Language* 110 (February): 104038. https://doi.org/10/gf9tjp.

Simpson, Daniel P., Håvard Rue, Andrea Riebler, Thiago G. Martins, and Sigrunn H. Sørbye. 2017. "Penalising Model Component Complexity: A Principled, Practical Approach to Constructing Priors." *Statistical Science* 32 (1): 1–28. https://doi.org/10.1214/16-STS576.

Sorensen, Tanner, Sven Hohenstein, and Shravan Vasishth. 2016. "Bayesian Linear Mixed Models Using Stan: A Tutorial for Psychologists, Linguists, and Cognitive Scientists." *Quantitative Methods for Psychology* 12 (3): 175–200.

Spiegelhalter, David J., Keith R. Abrams, and Jonathan P. Myles. 2004. *Bayesian Approaches to Clinical Trials and Health-Care Evaluation*. Vol. 13. John Wiley & Sons.

Spiegelhalter, David J., Laurence S. Freedman, and Mahesh K. B. Parmar. 1994. "Bayesian Approaches to Randomized Trials." *Journal of the Royal Statistical Society. Series A (Statistics in Society)* 157 (3): 357–416.

Stan Development Team. 2024. "Stan Modeling Language Users Guide and Reference Manual, Version 2.32." https://mc-stan.org/docs/2_35/.

Sutton, Alexander J., Nicky J. Welton, Nicola Cooper, Keith R. Abrams, and A. E. Ades. 2012. *Evidence Synthesis for Decision Making in Healthcare*. Vol. 132. John Wiley & Sons.

Szollosi, Aba, David Kellen, Danielle J. Navarro, Richard M Shiffrin, Iris van Rooij, Trisha Van Zandt, and Christopher Donkin. 2020. "Is Preregistration Worthwhile?" *Trends in Cognitive Sciences* 24 (2): 94–95.

Tabor, Whitney, Bruno Galantucci, and Daniel Richardson. 2004. "Effects of Merely Local Syntactic Coherence on Sentence Processing." *Journal of Memory and Language* 50: 355–70.

Talts, Sean, Michael J. Betancourt, Daniel P. Simpson, Aki Vehtari, and Andrew Gelman. 2018. "Validating Bayesian Inference Algorithms with Simulation-Based Calibration." *arXiv Preprint arXiv:1804.06788*.

Tetlock, Philip, and Dan Gardner. 2015. *Superforecasting: The Art and Science of Prediction*. Crown Publishers.

Tversky, Amos, and Daniel Kahneman. 1983. "Extensional Versus Intuitive Reasoning: The Conjunction Fallacy in Probability Judgment." *Psychological Review* 90 (4): 293. https://doi.org/https://doi.org/10.1037/0033-295X.90.4.293.

Ulrich, Rolf, and Jeff Miller. 1993. "Information Processing Models Generating Lognormally Distributed Reaction Times." *Journal of Mathematical Psychology* 37 (4): 513–25. https://doi.org/10.1006/jmps.1993.1032.

———. 1994. "Effects of Truncation on Reaction Time Analysis." *Journal of Experimental Psychology: General* 123 (1): 34–80. https://doi.org/10/b8tsnh.

Vasishth, Shravan. 2015. "A Meta-Analysis of Relative Clause Processing in Mandarin Chinese Using Bias Modelling." Master's thesis, Sheffield, UK: School of Mathematics; Statistics, University of Sheffield. https://doi.org/https://doi.org/10.31234/osf.io/4un9k.

Vasishth, Shravan, Sven Bruessow, Richard L. Lewis, and Heiner Drenhaus. 2008. "Processing Polarity: How the Ungrammatical Intrudes on the Grammatical." *Cognitive Science* 32 (4, 4): 685–712. https://doi.org/https://doi.org/10.1080/03640210802066865.

Vasishth, Shravan, Zhong Chen, Qiang Li, and Gueilan Guo. 2013. "Processing Chinese Relative Clauses: Evidence for the Subject-Relative Advantage." *PLoS ONE* 8 (10): 1–14. https://doi.org/https://doi.org/10.1371/journal.pone.0077006.

Vasishth, Shravan, and Felix Engelmann. 2022. *Sentence Comprehension as a Cognitive Process: A Computational Approach*. Cambridge, UK: Cambridge University Press. https://books.google.de/books?id=6KZKzgEACAAJ.

Vasishth, Shravan, Daniela Mertzen, Lena A. Jäger, and Andrew Gelman. 2018. "The Statistical Significance Filter Leads to Overoptimistic Expectations of Replicability." *Journal of Memory and Language* 103: 151–75. https://doi.org/https://doi.org/10.1016/j.jml.2018.07.004.

Vasishth, Shravan, Bruno Nicenboim, Felix Engelmann, and Frank Burchert. 2019. "Computational Models of Retrieval Processes in Sentence Processing." *Trends in Cognitive Sciences* 23: 968–82. https://doi.org/https://doi.org/10.1016/j.tics.2019.09.003.

Vasishth, Shravan, Katja Suckow, Richard L. Lewis, and Sabine Kern. 2011. "Short-Term Forgetting in Sentence Comprehension: Crosslinguistic Evidence from Head-Final Structures." *Language and Cognitive Processes* 25: 533–67. https://doi.org/https://doi.org/10.1080/01690960903310587.

Vasishth, Shravan, Himanshu Yadav, Daniel J. Schad, and Bruno Nicenboim. 2022. "Sample Size Determination for Bayesian Hierarchical Models Commonly Used in Psycholinguistics." *Computational Brain and Behavior*. https://doi.org/https://doi.org/10.1007/s42113-021-00125-y.

Vehtari, Aki, Andrew Gelman, and Jonah Gabry. 2017. "Practical Bayesian Model Evaluation Using Leave-One-Out Cross-Validation and WAIC." *Statistics and Computing* 27 (5): 1413–32. https://doi.org/10.1007/s11222-016-9696-4.

Von Baeyer, Hans Christian. 1988. "How Fermi Would Have Fixed It." *The Sciences* 28 (5): 2–4. https://doi.org/https://doi.org/10.1002/j.2326-1951.1988.tb03037.x.

Wagenmakers, Eric-Jan, and Scott D. Brown. 2007. "On the Linear Relation Between the Mean and the Standard Deviation of a Response Time Distribution." *Psychological Review* 114 (3): 830. https://doi.org/https://doi.org/10.1037/0033-295X.114.3.830.

Wagenmakers, Eric-Jan, Raoul P. P. P. Grasman, and Peter C. M. Molenaar. 2005. "On the Relation Between the Mean and the Variance of a Diffusion Model Response Time Distribution." *Journal of Mathematical Psychology* 49 (3): 195–204. https://doi.org/10.1016/j.jmp.2005.02.003.

Wahn, Basil, Daniel P. Ferris, W. David Hairston, and Peter König. 2016. "Pupil Sizes Scale with Attentional Load and Task Experience in a Multiple Object Tracking Task." *PLOS ONE* 11 (12): e0168087. https://doi.org/10.1371/journal.pone.0168087.

Wickelmaier, Florian, and Achim Zeileis. 2018. "Using Recursive Partitioning to Account for Parameter Heterogeneity in Multinomial Processing Tree Models." *Behavior Research Methods* 50 (3): 1217–33. https://doi.org/https://doi.org/10.3758/s13428-017-0937-z.

Wilson, Greg, Jennifer Bryan, Karen Cranston, Justin Kitzes, Lex Nederbragt, and Tracy K. Teal. 2017. "Good Enough Practices in Scientific Computing." *PLoS Computational Biology* 13 (6): e1005510.