Chapter 4: Bayesian regression models

Shravan Vasishth (vasishth.github.io)

July 2025

Contents

Textbook

Introduction to Bayesian Data Analysis for Cognitive Science

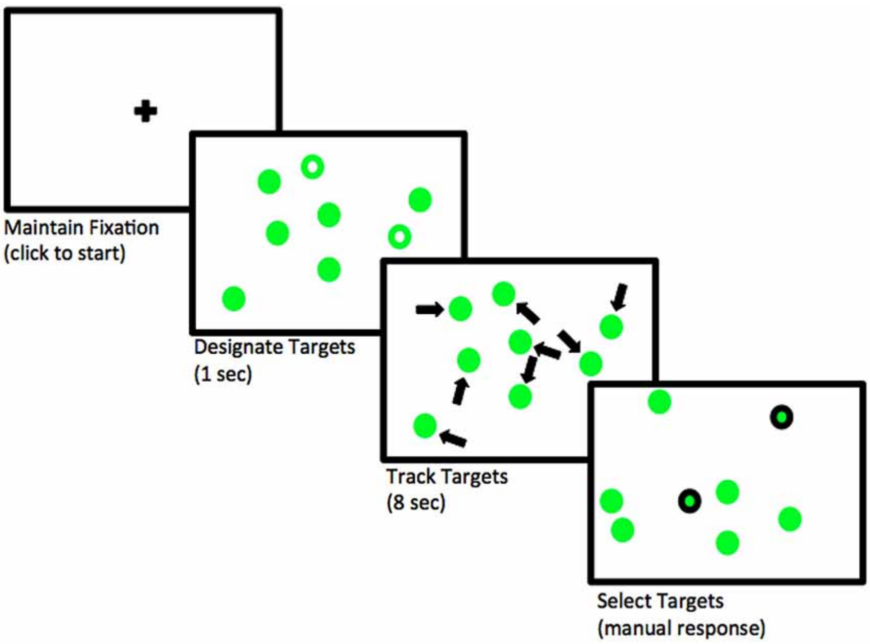
Nicenboim, Schad, Vasishth

- Online version: https://bruno.nicenboim.me/bayescogsci/
- Source code: https://github.com/bnicenboim/bayescogsci
- Physical book: here

Example: Multiple object tracking

- The subject covertly tracks between zero and five objects among several randomly moving objects on a computer screen.
- First, several objects appear on the screen, and a subset of them are indicated as "targets" at the beginning.
- Then, the objects start moving randomly across the screen and become indistinguishable.
- After several seconds, the objects stop moving and the subject need to indicate which objects were the targets.

Our research goal is to examine **how the attentional load affects pupil size**.



A model for this design

A model for this experiment design:

$$p\_size_n \sim Normal(\alpha + c\_load_n \cdot \beta, \sigma) \quad (1)$$

- n indicates the observation number with $n = 1, \ldots, N$
- c_load refers to centered load.
- Every data point is assumed to be independent (in frequentist terms: iid).

Pilot data for working out priors

Some pilot data helps us work out priors:

```
data("df_pupil_pilot")
df_pupil_pilot$p_size %>% summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   851.5   856.0   862.0   860.8   866.5   868.0
```

This suggests we can use the following regularizing prior for α:

$$\alpha \sim Normal(1000, 500) \qquad (2)$$

What we are expressing with this prior:

```
qnorm(c(0.025, 0.975), mean = 1000, sd = 500)
```

```
## [1]   20.01801 1979.98199
```

For σ, we use an uninformative prior:

$$\sigma \sim Normal_{+}(0, 1000) \qquad (3)$$

```
extraDistr::qtnorm(c(.025,0.975),
mean = 0, sd = 1000, a = 0)
```

```
## [1]   31.33798 2241.40273
```

$$\beta \sim Normal(0, 100) \tag{4}$$

```r
qnorm(c(0.025, 0.975), mean = 0, sd = 100)
```

```
## [1] -195.9964  195.9964
```

Fit the model

First, center the predictor:

```r
data("df_pupil")
(df_pupil <- df_pupil %>%
  mutate(c_load = load - mean(load)))
```

```
## # A tibble: 41 x 5
##     subj trial  load p_size c_load
##    <int> <int> <int>  <dbl>  <dbl>
##  1   701     1     2  1021. -0.439
##  2   701     2     1   951. -1.44
##  3   701     3     5  1064.  2.56
##  4   701     4     4   913.  1.56
##  5   701     5     0   603. -2.44
##  6   701     6     3   826.  0.561
##  7   701     7     0   464. -2.44
##  8   701     8     4   758.  1.56
##  9   701     9     2   733. -0.439
## 10   701    10     3   591.  0.561
## # i 31 more rows
```

```r
fit_pupil <- brm(p_size ~ 1 + c_load,
  data = df_pupil,
  family = gaussian(),
  prior = c(
    prior(normal(1000, 500), class = Intercept),
```

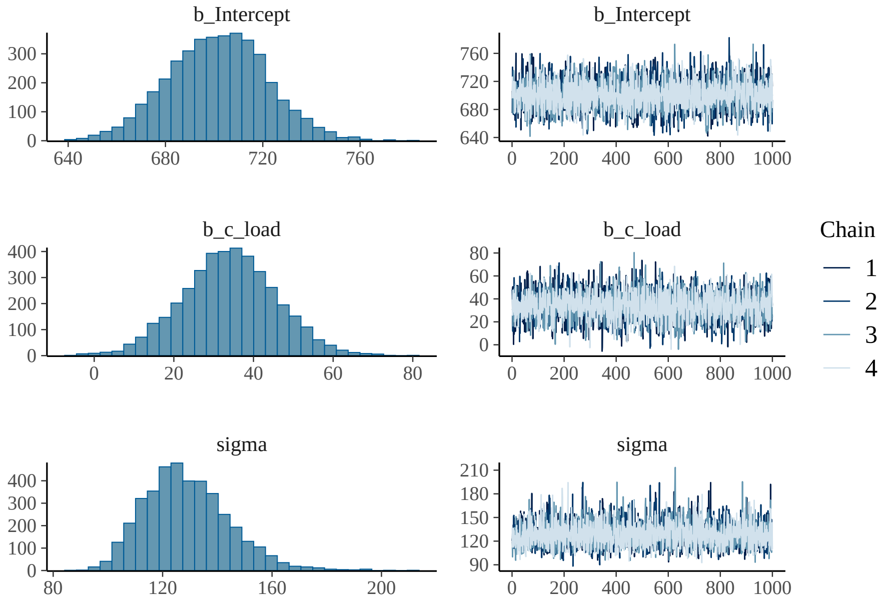
```r
    prior(normal(0, 1000), class = sigma),
    prior(normal(0, 100), class = b, coef = c_load)
  )
)
```

Posterior distributions of the parameters

Next, we will plot the posterior distributions of the parameters, and the posterior predictive distributions for the different load levels.

```r
data("df_pupil")
(df_pupil <- df_pupil %>%
  mutate(c_load = load - mean(load)))

fit_pupil <- brm(p_size ~ 1 + c_load,
  data = df_pupil,
  family = gaussian(),
  prior = c(
    prior(normal(1000, 500), class = Intercept),
    prior(normal(0, 1000), class = sigma),
    prior(normal(0, 100), class = b, coef = c_load)
  )
)
```

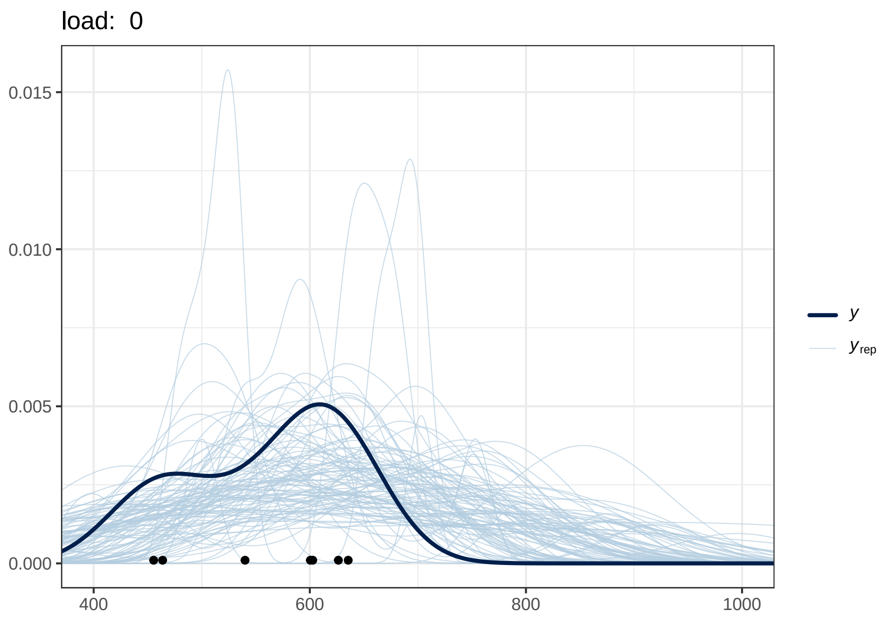
```r
plot(fit_pupil)
```



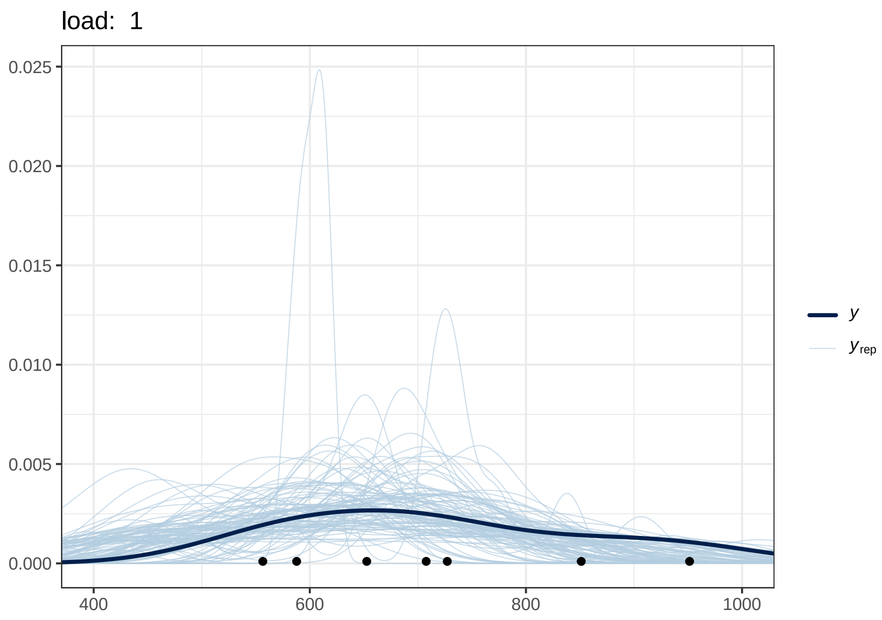

```r
## Note: short_summary is
## a function we wrote
short_summary(fit_pupil)
```

```
## ...
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept   701.54     20.25   661.50   741.19 1.00     3719     3088
## c_load       33.96     11.86    10.97    57.34 1.00     3333     2826
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma   128.89     15.51   103.58   162.35 1.00     3598     2882
##
## ...
```

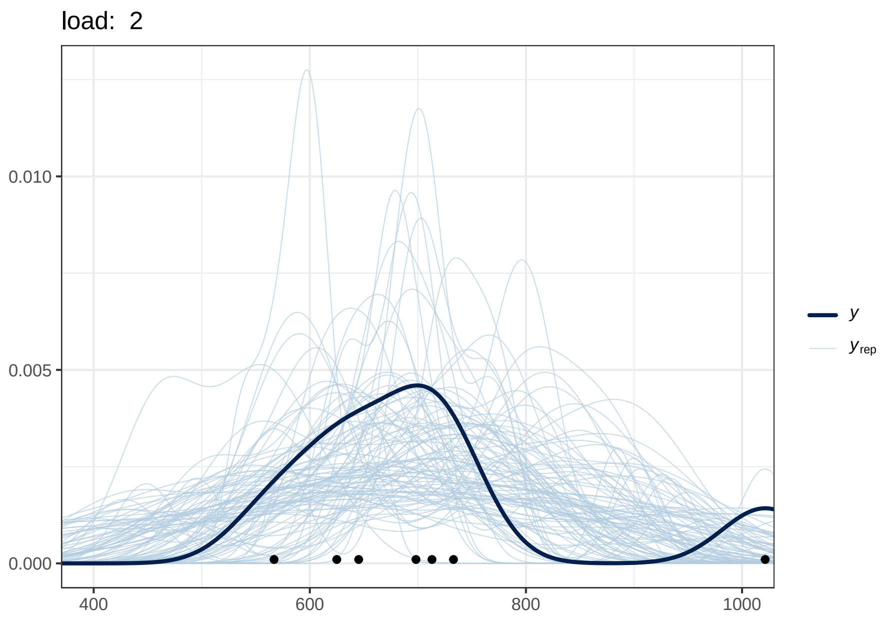
```r
l<-0
  df_sub_pupil <- filter(df_pupil, load == l)
  p <- pp_check(fit_pupil,
    type = "dens_overlay",
    ndraws = 100,
    newdata = df_sub_pupil
  ) +
    geom_point(data = df_sub_pupil,
    aes(x = p_size, y = 0.0001)) +
    ggtitle(paste("load: ", l)) +
    coord_cartesian(xlim = c(400, 1000)) +
    theme_bw()
  print(p)
```



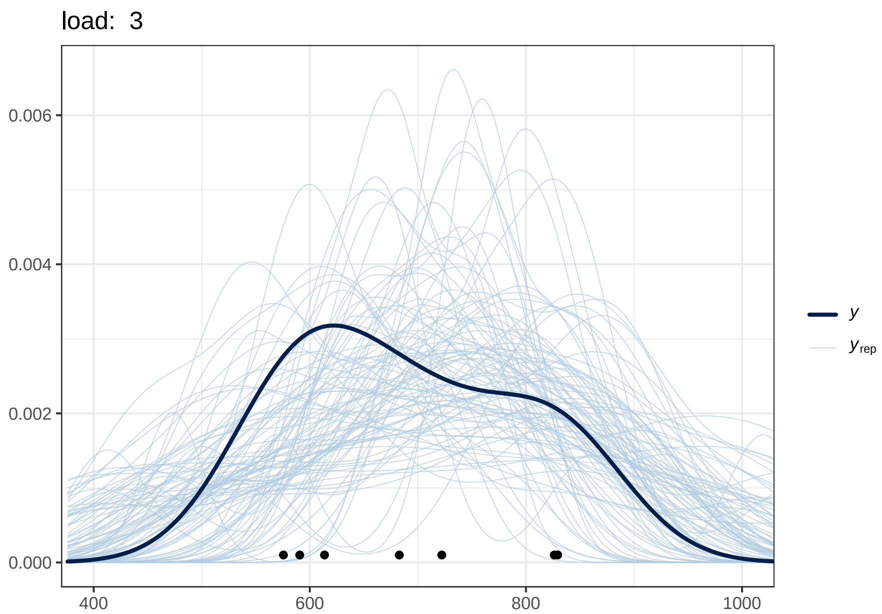

```
l<-1
  df_sub_pupil <- filter(df_pupil, load == l)
  p <- pp_check(fit_pupil,
    type = "dens_overlay",
    ndraws = 100,
    newdata = df_sub_pupil
  ) +
    geom_point(data = df_sub_pupil,
    aes(x = p_size, y = 0.0001)) +
    ggtitle(paste("load: ", l)) +
    coord_cartesian(xlim = c(400, 1000)) +
    theme_bw()
  print(p)
```



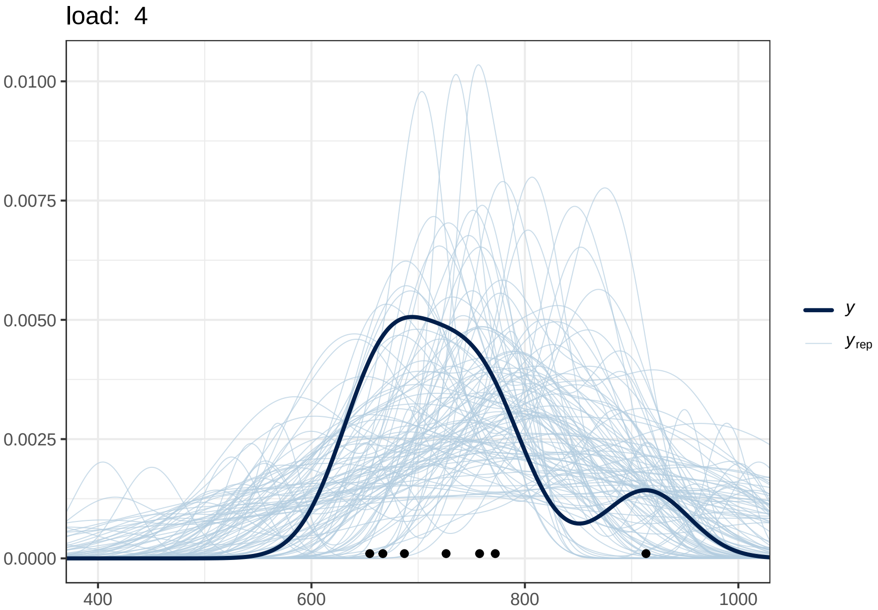

```
l<-2
  df_sub_pupil <- filter(df_pupil, load == l)
  p <- pp_check(fit_pupil,
    type = "dens_overlay",
    ndraws = 100,
    newdata = df_sub_pupil
  ) +
    geom_point(data = df_sub_pupil,
    aes(x = p_size, y = 0.0001)) +
    ggtitle(paste("load: ", l)) +
    coord_cartesian(xlim = c(400, 1000)) + theme_bw()
  print(p)
```




```
l<-3
  df_sub_pupil <- filter(df_pupil, load == l)
  p <- pp_check(fit_pupil,
    type = "dens_overlay",
    ndraws = 100,
    newdata = df_sub_pupil
  ) +
    geom_point(data = df_sub_pupil,
    aes(x = p_size, y = 0.0001)) +
    ggtitle(paste("load: ", l)) +
    coord_cartesian(xlim = c(400, 1000)) +
    theme_bw()
  print(p)
```




```
l<-4
  df_sub_pupil <- filter(df_pupil,
  load == l)
  p <- pp_check(fit_pupil,
    type = "dens_overlay",
    ndraws = 100,
    newdata = df_sub_pupil
  ) +
    geom_point(data = df_sub_pupil,
    aes(x = p_size, y = 0.0001)) +
    ggtitle(paste("load: ", l)) +
    coord_cartesian(xlim = c(400, 1000)) +
    theme_bw()
  print(p)
```



Using the log-normal likelihood

Next, we will look at another example: the effect of trial id on button-pressing times. This time, we will use the log-normal likelihood.

```
df_spacebar <- df_spacebar %>%
  mutate(c_trial = trial - mean(trial))
```

If we assume that button-pressing times are log-normally distributed, we could proceed as follows:

$$t_n \sim LogNormal(\alpha + c\_trial_n \cdot \beta, \sigma) \tag{5}$$

where

- N is the total number of (independent!) data points
- $n = 1, \ldots, N$, and
- rt is the dependent variable (response times in milliseconds).

The priors have to be defined on the log scale:

$$\begin{aligned} \alpha &\sim Normal(6, 1.5) \\ \sigma &\sim Normal_{+}(0, 1) \end{aligned} \tag{6}$$

A new parameter, β, needs a prior specification:

$$\beta \sim Normal(0, 1) \tag{7}$$

This prior on β is very uninformative.

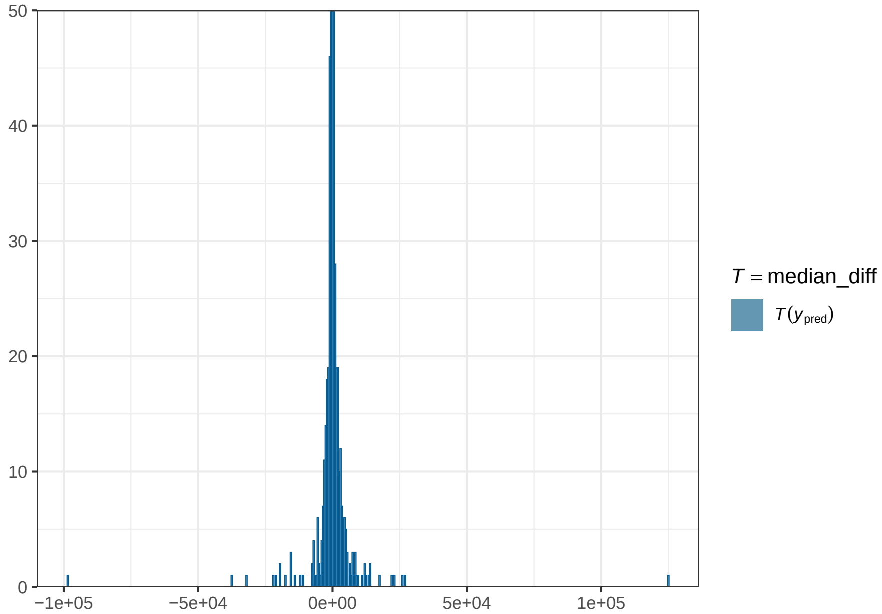
Prior predictive distributions

```
df_spacebar_ref <- df_spacebar %>%
  mutate(rt = rep(1, n()))
fit_prior_press_trial <- brm(t ~ 1 + c_trial,
  data = df_spacebar_ref,
  family = lognormal(),
  prior = c(
    prior(normal(6, 1.5), class = Intercept),
    prior(normal(0, 1), class = sigma),
    prior(normal(0, 1), class = b,
    coef = c_trial)
  ),
  sample_prior = "only",
  control = list(adapt_delta = 0.9)
)
```

```
median_diff <- function(x) {
  median(x - lag(x), na.rm = TRUE)
}
pp_check(fit_prior_press_trial,
         type = "stat",
         stat = "median_diff",
```

```
  # show only prior predictive
  # distributions
          prefix = "ppd",
  # each bin has a width of 500ms
          binwidth = 500) +
  # cut the top of the plot to improve its scale
  coord_cartesian(ylim = c(0, 50))+theme_bw()
```

```
## Using all posterior draws for ppc type 'stat' by default.
```



What would the prior predictive distribution look like if we set the following more informative prior on β?

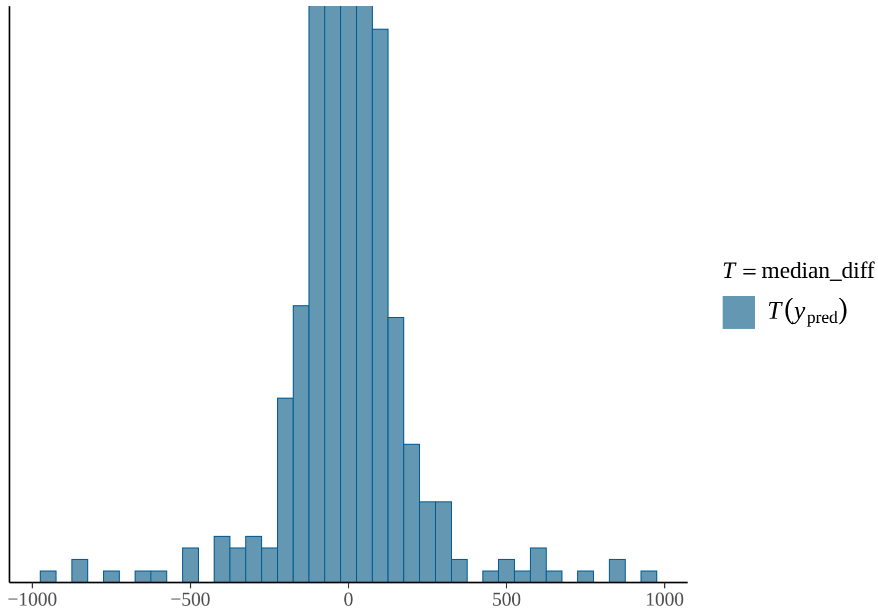
$$\beta \sim Normal(0, 0.01) \tag{8}$$

```
fit_prior_press_trial <- brm(t ~ 1 + c_trial,
  data = df_spacebar_ref,
  family = lognormal(),
  prior = c(
    prior(normal(6, 1.5), class = Intercept),
    prior(normal(0, 1), class = sigma),
    prior(normal(0, .01), class = b, coef = c_trial)
```

```
  ),
  sample_prior = "only",
  control = list(adapt_delta = .9)
)
```

```
pp_check(fit_prior_press_trial,
         type = "stat",
         prefix = "ppd",
         binwidth = 50,
         stat = "median_diff") +
  coord_cartesian(ylim = c(0, 50))
```

```
## Using all posterior draws for ppc type 'stat' by default.
```



Now that we have decided on our priors, we fit the model.

```
data("df_spacebar")
df_spacebar <- df_spacebar %>%
  mutate(c_trial = trial - mean(trial))
```

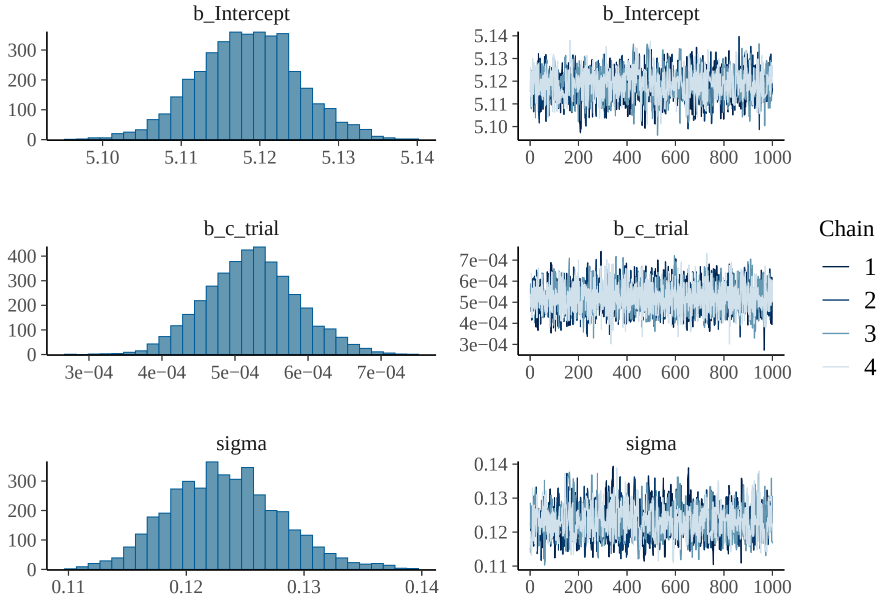
Fit the model:

```
fit_press_trial <- brm(t ~ 1 + c_trial,
  data = df_spacebar,
```

```r
  family = lognormal(),
  prior = c(
    prior(normal(6, 1.5), class = Intercept),
    prior(normal(0, 1), class = sigma),
    prior(normal(0, .01), class = b, coef = c_trial)
  )
)
```

Summarize posteriors (graphically or in a table, or both):

```r
plot(fit_press_trial)
```



Summarize results on the ms scale (the effect estimate from the middle of the expt to the preceding trial):

```r
alpha_samples <- as_draws_df(fit_press_trial)$b_Intercept
beta_samples <- as_draws_df(fit_press_trial)$b_c_trial

beta_ms <- exp(alpha_samples) -
           exp(alpha_samples - beta_samples)

beta_msmean <- round(mean(beta_ms), 5)
```

```r
beta_mslow <- round(quantile(beta_ms, prob = 0.025), 5)
beta_mshigh <- round(quantile(beta_ms, prob = 0.975), 5)
c(beta_msmean , beta_mslow, beta_mshigh)
```

```
##              2.5%   97.5%
## 0.08731 0.06716 0.10855
```

The effect estimate at the first vs second trial:

```r
first_trial <- min(df_spacebar$c_trial)
second_trial <- min(df_spacebar$c_trial) + 1
effect_beginning_ms <-
  exp(alpha_samples + second_trial * beta_samples) -
  exp(alpha_samples + first_trial * beta_samples)
## ms effect from first to second trial:
c(mean = mean(effect_beginning_ms),
  quantile(effect_beginning_ms, c(0.025, 0.975)))
```

```
##       mean       2.5%      97.5%
## 0.07940676 0.06250213 0.09668913
```

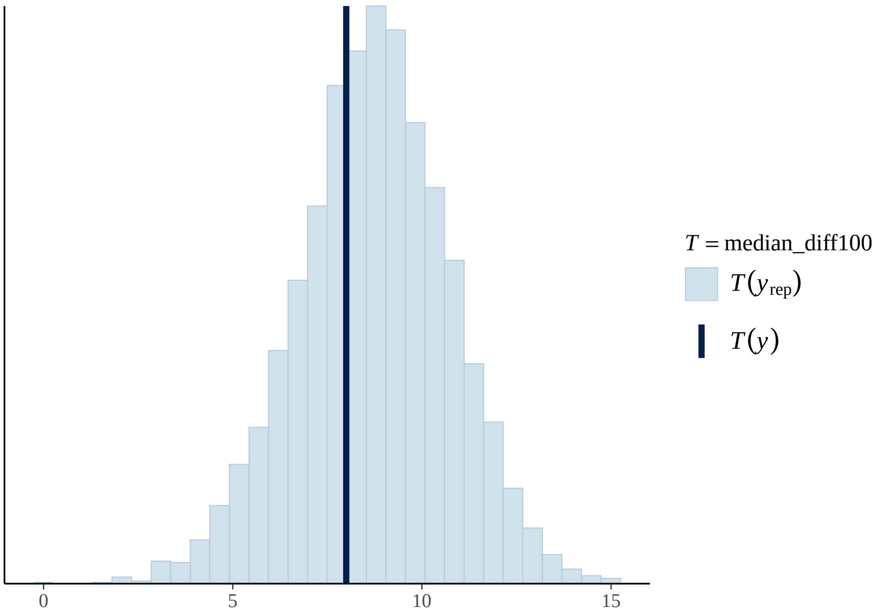
Slowdown after 100 trials from the middle of the expt:

```r
effect_100 <-
  exp(alpha_samples + 100 * beta_samples) -
  exp(alpha_samples)
c(mean = mean(effect_100),
  quantile(effect_100, c(0.025, 0.975)))
```

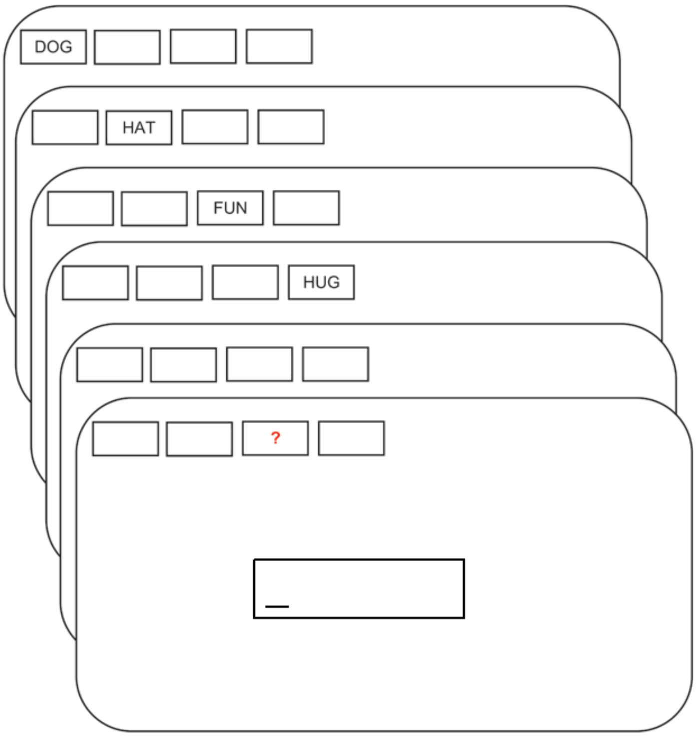
```
##      mean      2.5%     97.5%
##  8.968848  6.853161 11.217428
```

The posterior predictive distribution (distribution of predicted median differences between the n and n-100th trial):

```r
median_diff100 <- function(x) median(x -
                         lag(x, 100), na.rm = TRUE)
pp_check(fit_press_trial,
         type = "stat",
         stat = "median_diff100")
```



Logistic regression



```r
data("df_recall")
head(df_recall)
```

```
## # A tibble: 6 x 7
##   subj  set_size correct trial session block tested
##   <chr>    <int>   <int> <int>   <int> <int>  <int>
## 1 10           4       1     1       1     1      2
## 2 10           8       0     4       1     1      8
## 3 10           2       1     9       1     1      2
## 4 10           6       1    23       1     1      2
## 5 10           4       1     5       1     2      3
## 6 10           8       0     7       1     2      5
```

```r
df_recall <- df_recall %>%
  mutate(c_set_size = set_size - mean(set_size))
```

```r
# Set sizes in the data set:
df_recall$set_size %>%
  unique() %>% sort()
```

```
## [1] 2 4 6 8
```

```r
# Trials by set size
df_recall %>%
  group_by(set_size) %>%
  count()
```

```
## # A tibble: 4 x 2
## # Groups:   set_size [4]
##   set_size     n
##      <int> <int>
## 1        2    23
## 2        4    23
## 3        6    23
## 4        8    23
```

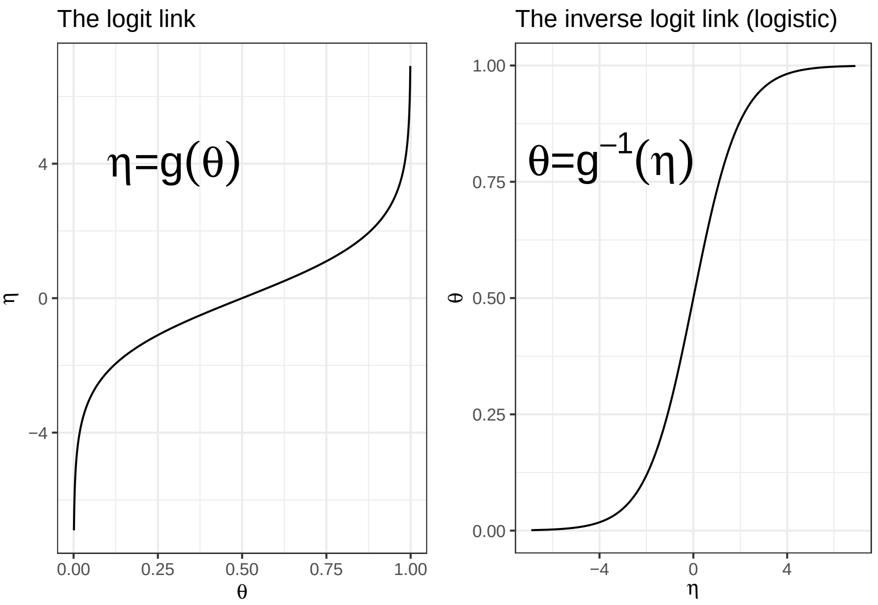
$$correct_n \sim Bernoulli(\theta_n) \tag{9}$$

$$\eta_n = g(\theta_n) = \log\left(\frac{\theta_n}{1-\theta_n}\right) \tag{10}$$

```
x <- seq(0.001, 0.999, by = 0.001)
y <- log(x / (1 - x))
logistic_dat <- data.frame(theta = x, eta = y)

p1 <- qplot(logistic_dat$theta,
    logistic_dat$eta, geom = "line") +
  xlab(expression(theta)) +
  ylab(expression(eta)) +
  ggtitle("The logit link") +
  annotate("text",
    x = 0.3, y = 4,
    label = expression(paste(eta, "=",
                            g(theta))),
                            parse = TRUE,
    size = 8
  ) + theme_bw()
```

```
p2 <- qplot(logistic_dat$eta, logistic_dat$theta,
            geom = "line") + xlab(expression(eta)) +
  ylab(expression(theta)) +
  ggtitle("The inverse logit link (logistic)") +
  annotate("text",
  x = -3.5, y = 0.80,
  label = expression(paste(theta, "=", g^-1,
                          (eta))),
                          parse = TRUE, size = 8
) + theme_bw()
```

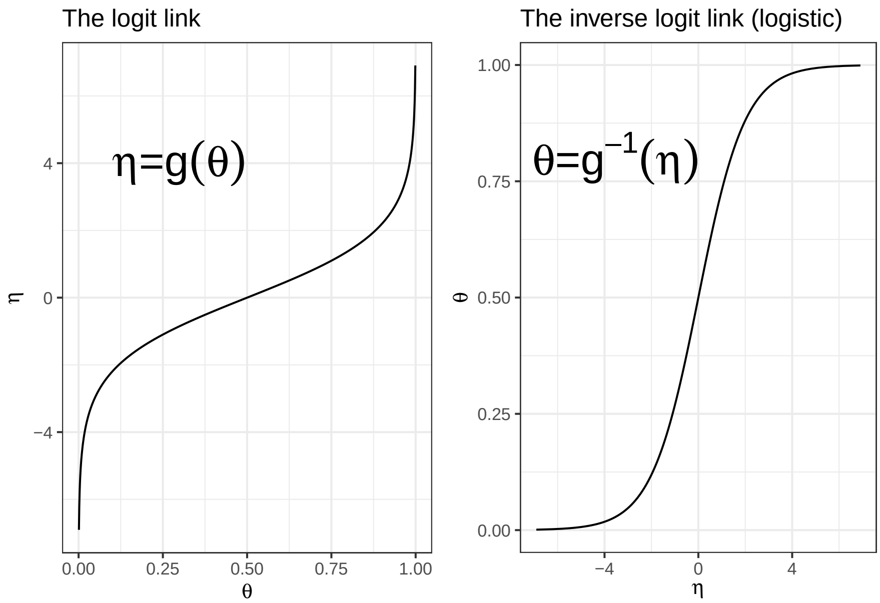
```
gridExtra::grid.arrange(p1, p2, ncol = 2)
```




```
x <- seq(0.001, 0.999, by = 0.001)
y <- log(x / (1 - x))
logistic_dat <- data.frame(theta = x, eta = y)

p1 <- qplot(logistic_dat$theta,
      logistic_dat$eta, geom = "line") +
      xlab(expression(theta)) +
      ylab(expression(eta)) +
      ggtitle("The logit link") +
      annotate("text",
      x = 0.3, y = 4,
      label = expression(paste(eta, "=",
      g(theta))), parse = TRUE, size = 8
  ) +
  theme_bw()


p2 <- qplot(logistic_dat$eta,
      logistic_dat$theta, geom = "line") +
      xlab(expression(eta)) +
```

```
    ylab(expression(theta)) +
    ggtitle("The inverse logit link (logistic)") +
    annotate("text",
  x = -3.5, y = 0.80,
  label = expression(paste(theta, "=", g^-1, (eta))),
  parse = TRUE, size = 8
) + theme_bw()

gridExtra::grid.arrange(p1, p2, ncol = 2)
```

```
## Warning in is.na(x): is.na() applied to non-(list or vector)
## 'expression'
## Warning in is.na(x): is.na() applied to non-(list or vector)
## 'expression'
```



Deciding on priors

```
data("df_recall")
head(df_recall)
```

```
## # A tibble: 6 x 7
##   subj  set_size correct trial session block tested
##   <chr>    <int>   <int> <int>   <int> <int>  <int>
```

```
## 1 10          4    1    1    1    1    2
## 2 10          8    0    4    1    1    8
## 3 10          2    1    9    1    1    2
## 4 10          6    1   23    1    1    2
## 5 10          4    1    5    1    2    3
## 6 10          8    0    7    1    2    5
```

```r
df_recall <- df_recall %>%
  mutate(c_set_size = set_size - mean(set_size))
```

The linear model is now fit not to the 0,1 responses as the dependent variable, but to η_n, i.e., log-odds, as the dependent variable:

$$\eta_n = \log\left(\frac{\theta_n}{1-\theta_n}\right) = \alpha + \beta \cdot c\_set\_size_n \quad (11)$$

- Unlike the linear models, the model is defined so that there is no residual error term (ε) in this model.
- Once η_n is estimated, one can solve the above equation for θ_n (in other words, we compute the inverse of the logit function and obtain the estimates on the probability scale).

This gives the above-mentioned logistic regression function:

$$\theta_n = g^{-1}(\eta_n) = \frac{\exp(\eta_n)}{1+\exp(\eta_n)} = \frac{1}{1+exp(-\eta_n)} \quad (12)$$

In summary, the generalized linear model with the logit link fits the following Bernoulli likelihood:

$$correct_n \sim Bernoulli(\theta_n) \tag{13}$$

- The model is fit on the log-odds scale, $\eta_n = \alpha + c\_set\_size_n \cdot \beta$.
- Once η_n has been estimated, the inverse logit or the logistic function is used to compute the probability estimates $\theta_n = \frac{\exp(\eta_n)}{1+\exp(\eta_n)}$.

There are two functions in R that implement the logit and inverse logit functions:

- `qlogis(p)` for the logit function and
- `plogis(x)` for the inverse logit or logistic function.

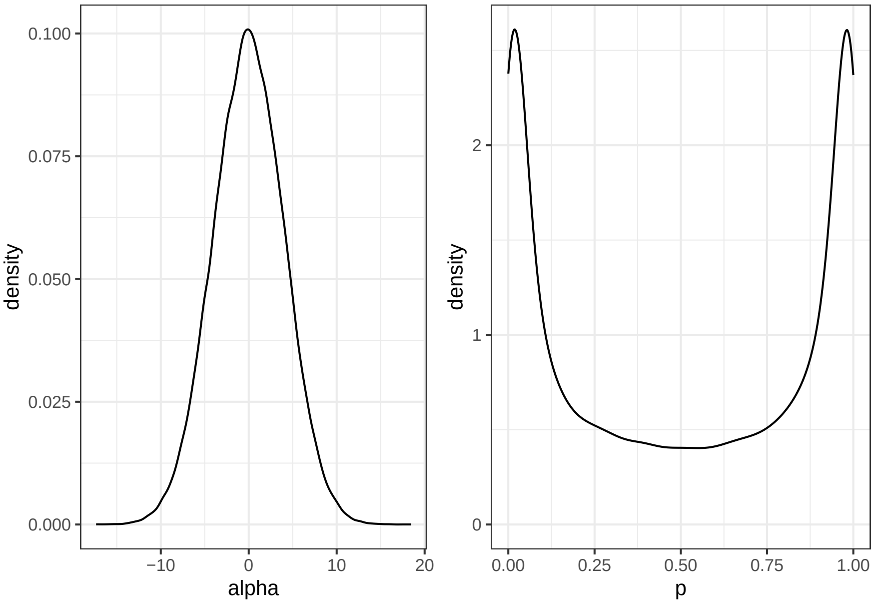
$$\alpha \sim Normal(0, 4) \tag{14}$$

Let's plot this prior in log-odds and in probability scale by drawing random samples.

Prior for $\alpha \sim Normal(0, 4)$ in log-odds and in probability space.

```
samples_logodds <- tibble(alpha = rnorm(100000,
                          0, 4))
samples_prob <- tibble(p = plogis(rnorm(100000,
                         0, 4)))
pa<-ggplot(samples_logodds, aes(alpha)) +
  geom_density()+theme_bw()
pb<-ggplot(samples_prob, aes(p)) +
```

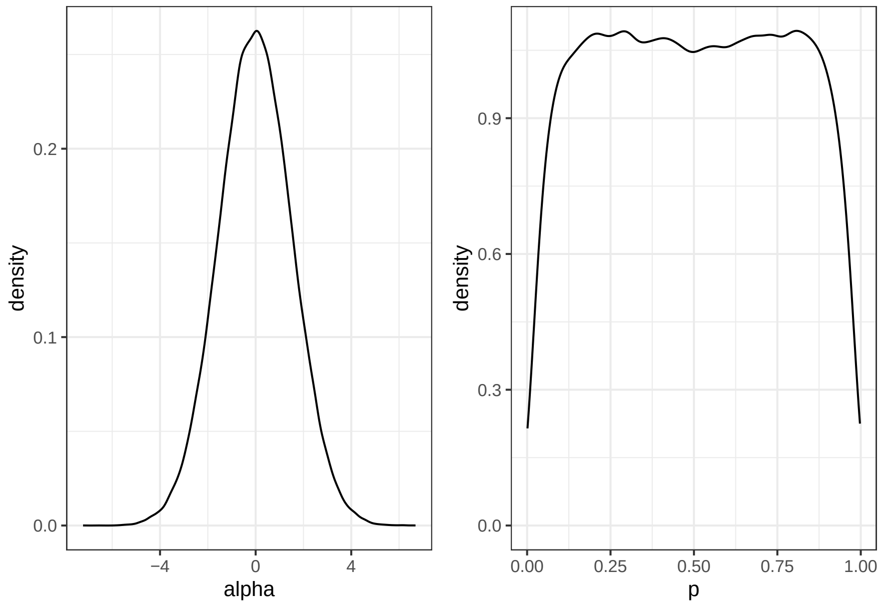
```
  geom_density()+theme_bw()
gridExtra::grid.arrange(pa, pb, ncol = 2)
```



$$\alpha \sim Normal(0, 1.5) \tag{15}$$

Prior for $\alpha \sim Normal(0, 1.5)$ in log-odds and in probability space.

```
samples_logodds <- tibble(alpha = rnorm(100000,
                          0, 1.5))
samples_prob <- tibble(p = plogis(rnorm(100000,
                         0, 1.5)))
paa<-ggplot(samples_logodds, aes(alpha)) +
  geom_density()+theme_bw()
pbb<-ggplot(samples_prob, aes(p)) +
  geom_density()+theme_bw()
gridExtra::grid.arrange(paa, pbb, ncol = 2)
```



We can examine the consequences of each of the following prior specifications:

1. $\beta \sim Normal(0, 1)$
2. $\beta \sim Normal(0, 0.5)$
3. $\beta \sim Normal(0, 0.1)$
4. $\beta \sim Normal(0, 0.01)$
5. $\beta \sim Normal(0, 0.001)$

```
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:
        theta = plogis(alpha + c_set_size * beta),
        correct_pred = extraDistr::rbern(n = N_obs,
```

```r
        prob = theta)
      )
    },
    .id = "iter"
  ) %>%
    # .id is always a string and has to
    # be converted to a number
    mutate(iter = as.numeric(iter))
}
```

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

```r
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)
})
```

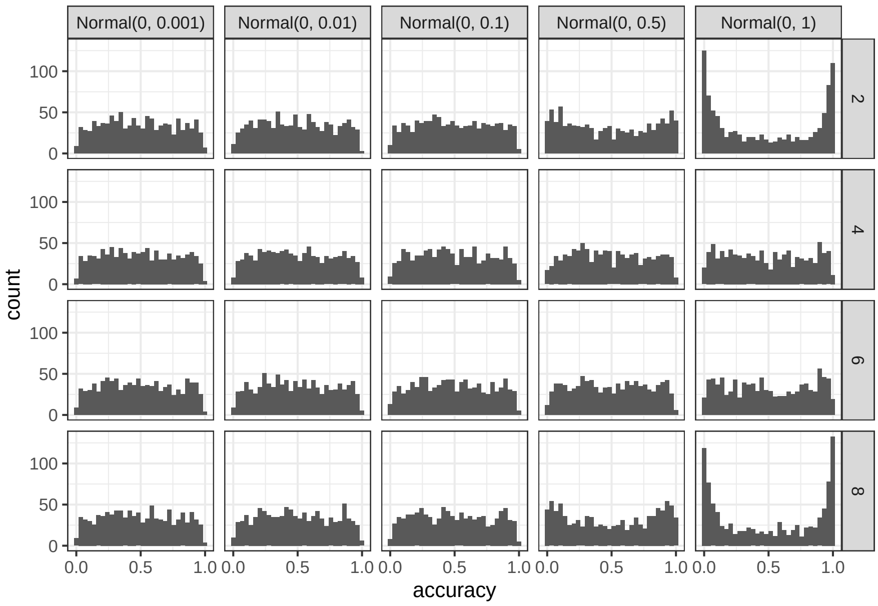
```r
mean_accuracy <-
  prior_pred %>%
  group_by(prior_beta_sd, iter, set_size) %>%
  summarize(accuracy = mean(correct_pred)) %>%
  mutate(prior = paste0("Normal(0, ",
  prior_beta_sd, ")"))
```

```
## `summarise()` has grouped output by 'prior_beta_sd', 'iter'.
```

```
## using the `.groups` argument.
```

```r
mean_accuracy %>%
  ggplot(aes(accuracy)) +
  geom_histogram() +
  facet_grid(set_size ~ prior) +
  scale_x_continuous(breaks = c(0, 0.5, 1))+
  theme_bw()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwi
```

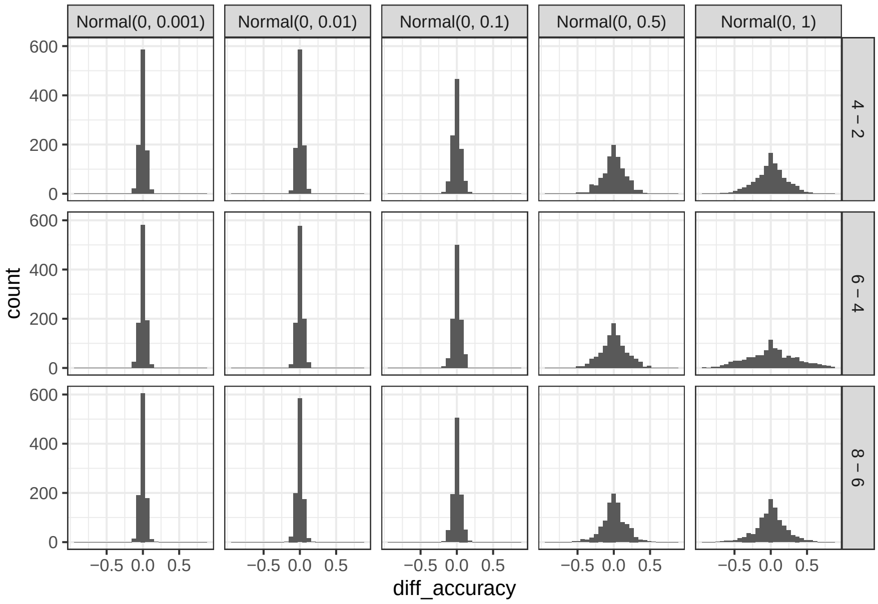


It's usually more useful to look at the predicted differences in accuracy between set sizes.

```r
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)
```

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

```r
  facet_grid(diffsize ~ prior) +
  scale_x_continuous(breaks = c(-0.5, 0, 0.5)) +
  theme_bw()
```



These priors seem reasonable:

$$
\begin{aligned}
\alpha &\sim Normal(0, 1.5) \\
\beta &\sim Normal(0, 0.1)
\end{aligned}
\tag{16}
$$

Fit the model

Next: fit the model and examine the posterior distributions of the parameters.

```r
data("df_recall")
head(df_recall)
```

```
## # A tibble: 6 x 7
##   subj  set_size correct trial session block tested
##   <chr>    <int>   <int> <int>   <int> <int>  <int>
## 1 10           4       1     1       1     1      2
## 2 10           8       0     4       1     1      8
## 3 10           2       1     9       1     1      2
## 4 10           6       1    23       1     1      2
## 5 10           4       1     5       1     2      3
## 6 10           8       0     7       1     2      5
```

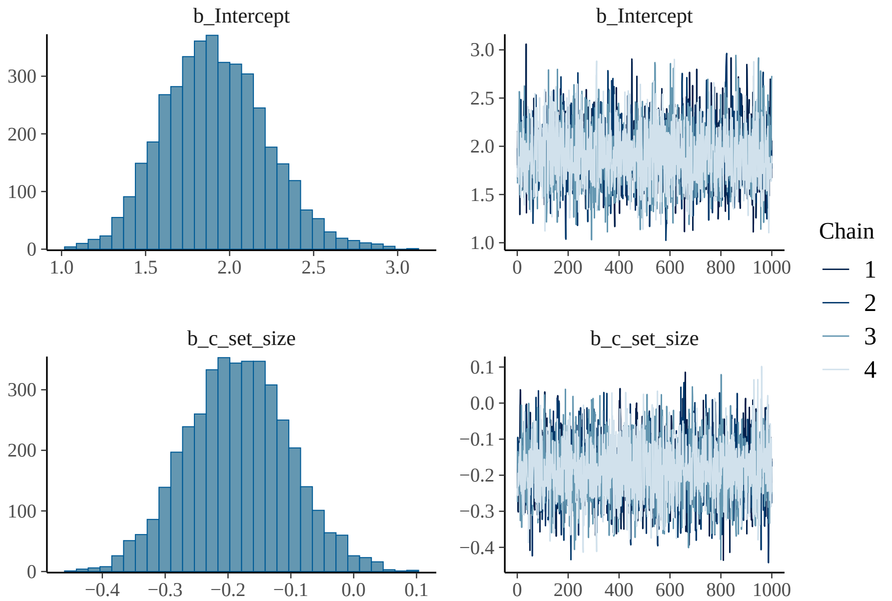
```r
df_recall <- df_recall %>%
  mutate(c_set_size = set_size - mean(set_size))
```

```r
fit_recall <- brm(correct ~ 1 + c_set_size,
  data = df_recall,
  family = bernoulli(link = logit),
  prior = c(
    prior(normal(0, 1.5), class = Intercept),
    prior(normal(0, .1), class = b,
    coef = c_set_size)
  )
)
```

```r
posterior_summary(fit_recall,
                  variable = c("b_Intercept",
                  "b_c_set_size"))
```

```
##                Estimate  Est.Error       Q2.5       Q97.5
## b_Intercept   1.9160614 0.30618832  1.360939  2.54766496
## b_c_set_size -0.1842353 0.08192451 -0.344429 -0.01897062
```

```r
plot(fit_recall)
```




```r
alpha_samples <- as_draws_df(fit_recall)$b_Intercept
beta_samples <- as_draws_df(fit_recall)$b_c_set_size
beta_mean <- round(mean(beta_samples), 5)
beta_low <- round(quantile(beta_samples,
                  prob = 0.025), 5)
beta_high <- round(quantile(beta_samples,
```

```r
              prob = 0.975), 5)

alpha_samples <- as_draws_df(fit_recall)$b_Intercept
av_accuracy <- plogis(alpha_samples)
c(mean = mean(av_accuracy), quantile(av_accuracy,
                                     c(0.025, 0.975)))
```

```
##      mean      2.5%     97.5%
## 0.8678746 0.7959122 0.9274165
```

Does set size affect free recall?

Find out the decrease in accuracy in proportions or probability scale:

```r
beta_samples <- as_draws_df(fit_recall)$b_c_set_size
effect_middle <- plogis(alpha_samples) -
  plogis(alpha_samples - beta_samples)
c(mean = mean(effect_middle),
  quantile(effect_middle, c(0.025, 0.975)))
```

```
##         mean         2.5%        97.5%
## -0.019002261 -0.037342092 -0.002084453
```

```r
four <- 4 - mean(df_recall$set_size)
two <- 2 - mean(df_recall$set_size)
effect_4m2 <-
  plogis(alpha_samples + four * beta_samples) -
  plogis(alpha_samples + two * beta_samples)
c(mean = mean(effect_4m2),
  quantile(effect_4m2, c(0.025, 0.975)))
```

```
##         mean         2.5%        97.5%
## -0.029656381 -0.054228972 -0.004103123
```

Conclusion (careful about the wording!)

The posterior distributions of the parameters (transformed to probability scale) are consistent with the claim that increasing set size reduces accuracy.

Notice that I did not write any of the following sentences:

- "There is a significant effect of set size on accuracy". This sentence is basically nonsensical since we didn't do a frequentist significance test.
- "We found that set size reduces accuracy": That is a discovery claim. Such a claim of the existence of an effect requires us to quantify the evidence for a model assuming that set size affects accuracy, relative to a baseline model. Later, we will use Bayes factors (or, even later, k-fold cross validation).

The wording I used simply states that the observed **pattern** is consistent with set size reducing accuracy. I am careful not to make a discovery claim. In particular, I am not claiming that I found a general truth about the nature of things.