HW 4 Solutions

Shravan Vasishth

Exercise 1

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

Fit a "maximal" model (correlated varying intercept and slopes for subjects) assuming a normal likelihood.

```
fit_pupil <- brm(p_size ~ 1 + c_load + (c_load | subj),
  data = df_pupil_complete,
  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),
    prior(normal(0, 1000), class = sd),
    prior(lkj(2), class = cor)),
  control=list(adapt_delta=0.99, max_treedepth=15))
```

(a) Examine the effect of load on pupil size, and the average pupil size.

```
fit_pupil
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: p_size ~ 1 + c_load + (c_load | subj)
##    Data: df_pupil_complete (Number of observations: 2228)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Multilevel Hyperparameters:
## ~subj (Number of levels: 20)
##                       Estimate Est.Error l-95% CI u-95% CI Rhat
## sd(Intercept)          3355.39    449.51  2546.78  4316.00 1.00
## sd(c_load)               71.45     15.23    47.53   106.55 1.00
## cor(Intercept,c_load)     0.30      0.24    -0.23     0.71 1.01
##                       Bulk_ESS Tail_ESS
## sd(Intercept)              717     1008
## sd(c_load)                1267     1951
## cor(Intercept,c_load)     1415     1886
##
## Regression Coefficients:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept  2469.01    481.63  1533.89  3391.77 1.00      638      952
```

```
## c_load       40.22     24.87    -9.67    88.54 1.01     1112     1580
##
## Further Distributional Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma   505.09      7.65   490.61   520.07 1.00     4557     2437
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

If we want to just report the change in pupil size as a function of one unit increase in (centered) load:

```
posterior_summary(fit_pupil, variable = "b_c_load")
```

```
##          Estimate Est.Error  Q2.5 Q97.5
## b_c_load     40.2      24.9 -9.67  88.5
```

There isn't a clear effect of load across all the subject.

But the intercept seems to be quite large,

```
posterior_summary(fit_pupil, variable = "b_Intercept")
```

```
##             Estimate Est.Error Q2.5 Q97.5
## b_Intercept     2469       482 1534  3392
```

and we assumed that it shouldn't be:

```
brms::prior_summary(fit_pupil)
```

```
##                 prior     class      coef group resp dpar nlpar lb ub
##                (flat)         b
##        normal(0, 100)         b    c_load
##     normal(1000, 500) Intercept
##  lkj_corr_cholesky(2)         L
##  lkj_corr_cholesky(2)         L            subj
##       normal(0, 1000)        sd                                    0
##       normal(0, 1000)        sd            subj                    0
##       normal(0, 1000)        sd    c_load  subj                    0
##       normal(0, 1000)        sd Intercept  subj                    0
##       normal(0, 1000)     sigma                                    0
##        source
##       default
##          user
##          user
##          user
##  (vectorized)
##          user
##  (vectorized)
##  (vectorized)
##  (vectorized)
##          user
```

See the row with class Intercept.

So maybe our prior for the intercept was overly informative.

(b) Do a sensitivity analysis for the prior on the intercept (α). What is the estimate of the effect (β) under different priors?

We'll try a wider prior for α. A more complete sensitivity analysis would investigate several possible priors

$$\alpha \sim Normal(4000, 2000)$$

```
fit_pupil_2 <- brm(p_size ~ 1 + c_load + (c_load | subj),
  data = df_pupil_complete,
  family = gaussian(),
  prior = c(
    prior(normal(4000, 2000), class = Intercept),
    prior(normal(0, 1000), class = sigma),
    prior(normal(0, 100), class = b, coef = c_load),
    prior(normal(0, 2000), class = sd),
  prior(lkj(2), class = cor))
)
```

```
fit_pupil_2
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: p_size ~ 1 + c_load + (c_load | subj)
##    Data: df_pupil_complete (Number of observations: 2228)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Multilevel Hyperparameters:
## ~subj (Number of levels: 20)
##                       Estimate Est.Error l-95% CI u-95% CI Rhat
## sd(Intercept)          2548.48    399.12  1896.16  3459.68 1.01
## sd(c_load)               69.05     14.58    46.09   103.36 1.00
## cor(Intercept,c_load)     0.27      0.20    -0.17     0.62 1.00
##                       Bulk_ESS Tail_ESS
## sd(Intercept)              872     1498
## sd(c_load)                1232     1743
## cor(Intercept,c_load)     1950     2526
##
## Regression Coefficients:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept  5464.64    572.30  4338.88  6579.55 1.01      549      773
## c_load       58.42     17.33    22.88    90.24 1.00     1411     1730
##
## Further Distributional Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma   505.40      7.79   490.45   520.88 1.00     5315     2559
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Now there seems to be a clear effect! Our bad prior for the intercept was messing up our inferences!

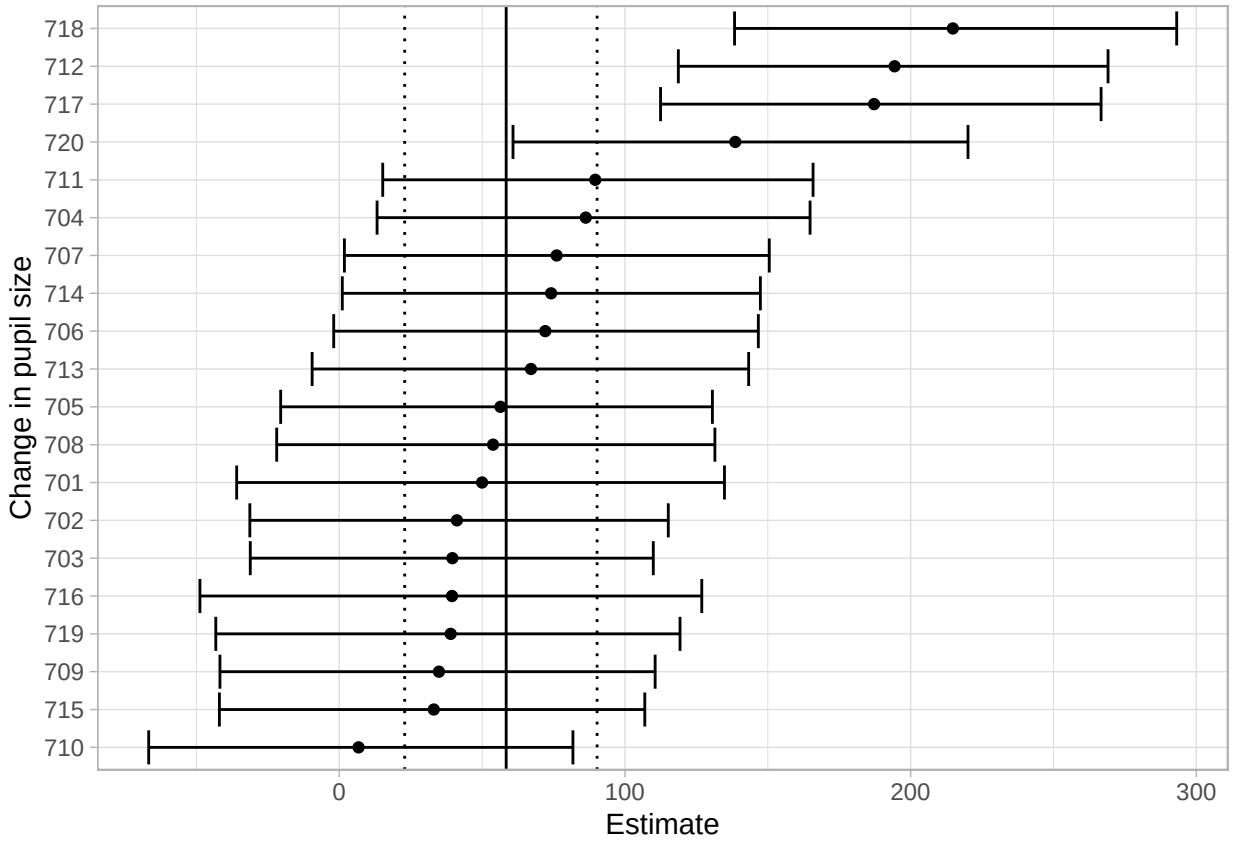
(c) Is the effect of load consistent across subjects?

```
## For the hierarchical model, this is more complicated,
# because we want the effect (beta) + adjustment:
# we extract the overall group level effect:
beta <- c(as_draws_df(fit_pupil_2)$b_c_load)
# We extract the individual adjustments
ind_effects_v <- paste0("r_subj[", unique(df_pupil_complete$subj), ",c_load]")
adjustment <- as.matrix(as_draws_df(fit_pupil)[ind_effects_v])
```

```
## Warning: Dropping 'draws_df' class as required metadata was removed.
```

```
# We get the by subject effects in a data frame where each adjustment
# is in each column.
by_subj_effect <- as_tibble(beta + adjustment)
# We summarize them by getting a table with the mean and the
# quantiles for each column and then binding them.
par_h <- lapply(by_subj_effect, function(x) {
  tibble(
    Estimate = mean(x),
    Q2.5 = quantile(x, .025),
    Q97.5 = quantile(x, .975)
  )
}) %>%
  bind_rows() %>%
  # We add a column to identify that the model,
  # and one with the subject labels:
  mutate(
    subj = unique(df_pupil_complete$subj))%>%
  arrange(Estimate) %>%
  mutate(subj = factor(subj, levels = subj))


ggplot(
  par_h,
  aes(
    ymin = Q2.5, ymax = Q97.5, x = subj, y = Estimate
  )
) +
  geom_errorbar() +
  geom_point() +
  # We'll also add the mean and 95% CrI of the overall difference
  # to the plot:
  geom_hline(
    yintercept =
      posterior_summary(fit_pupil_2)["b_c_load", "Estimate"]
  ) +
  geom_hline(
    yintercept =
      posterior_summary(fit_pupil_2)["b_c_load", "Q2.5"],
    linetype = "dotted", linewidth = 0.5
  ) +
  geom_hline(
    yintercept =
      posterior_summary(fit_pupil_2)["b_c_load", "Q97.5"],
```

```
  linetype = "dotted", linewidth = 0.5
) +
xlab("Change in pupil size") +
coord_flip()
```



Exercise 2

We are going to fit the data with the following log-normal likelihood:

$$rt_n \sim LogNormal(\alpha + u_{i[n],0} + w_{j[n],0} + c\_cond_n \cdot (\beta + u_{i[n],1} + w_{j[n],1}), \sigma) \tag{1}$$

We can use similar priors as we used for the Stroop task paying attention to use a not too tight prior for β:

- Priors:

$$\begin{aligned} \alpha &\sim Normal(6, 1.5) \\ \beta &\sim Normal(0, 0.1) \\ \sigma &\sim Normal_+(0, 1) \end{aligned} \tag{2}$$

Here, we will need priors for the group-level parameters. Given that we assume a correlation between by-subject intercept and slope, and by-item intercept and slope, our model has the following structure which requires us to assign priors to Σ_u and Σ_w

$$\begin{aligned} \begin{pmatrix} u_{i,0} \\ u_{i,1} \end{pmatrix} &\sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \boldsymbol{\Sigma_u}\right) \\ \begin{pmatrix} w_{i,0} \\ w_{i,1} \end{pmatrix} &\sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \boldsymbol{\Sigma_w}\right) \end{aligned} \tag{3}$$

$$\begin{aligned}\boldsymbol{\Sigma_u} &= \begin{pmatrix} \tau^2_{u_0} & \rho_u \tau_{u_0} \tau_{u_1} \\ \rho_u \tau_{u_0} \tau_{u_1} & \tau^2_{u_1} \end{pmatrix} \\ \boldsymbol{\Sigma_w} &= \begin{pmatrix} \tau^2_{w_0} & \rho_w \tau_{w_0} \tau_{w_1} \\ \rho_w \tau_{w_0} \tau_{w_1} & \tau^2_{w_1} \end{pmatrix}\end{aligned} \tag{4}$$

In practice this means that we need priors for the by-subject and by-item variances and correlations:

$$\begin{aligned}\tau_{u_0} &\sim Normal_+(0, 1) \\ \tau_{u_1} &\sim Normal_+(0, 1) \\ \rho_u &\sim LKJcorr(2) \\ \tau_{w_0} &\sim Normal_+(0, 1) \\ \tau_{w_1} &\sim Normal_+(0, 1) \\ \rho_w &\sim LKJcorr(2)\end{aligned} \tag{5}$$

Read the data:

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

We're ready to fit a model now:

```
fit_df_gg05_rc <- brm(RT ~ c_cond + (c_cond | subj) + (c_cond | item),
  family = lognormal(),
  iter = 10000,
  prior =
    c(
      prior(normal(6, 1.5), class = Intercept),
      prior(normal(0, .1), class = b),
      prior(normal(0, 1), class = sigma),
      prior(normal(0, 1), class = sd),
      prior(lkj(2), class = cor)
    ),
  data = df_gg05_rc,
  control=list(adapt_delta=0.99, max_treedepth=15)
)
```

(a) Examine the effect of relative clause attachment site (the predictor `c_cond`) on reading times `RT` in log-scale.

We'll focus on β:

```
posterior_summary(fit_df_gg05_rc, variable = "b_c_cond")
```

```
##          Estimate Est.Error    Q2.5 Q97.5
## b_c_cond   0.0984     0.046 0.00444 0.187
```

(b) Estimate the median difference between relative clause attachment sites in milliseconds, and report the mean and 95% CI.

```
alpha <- as_draws_df(fit_df_gg05_rc)$b_Intercept
beta <- as_draws_df(fit_df_gg05_rc)$b_c_cond
# Difference between object RC coded as .5 and subject RC coded as .5
```

```
effect <- exp(alpha + beta * .5) - exp(alpha + beta * -.5)
c(mean = mean(effect), quantile(effect, c(.025,.975)))
```

```
##  mean  2.5% 97.5%
## 35.33  1.54 69.35
```

(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?

Next, we do a sensitivity analysis using a tighter prior for β, $\beta \sim Normal(0, 0.01)$:

```
fit_df_gg05_rc2 <- brm(RT ~ c_cond + (c_cond | subj) + (c_cond | item),
  family = lognormal(),
  prior =
    c(
      prior(normal(6, 1.5), class = Intercept),
      prior(normal(0, 0.01), class = b),
      prior(normal(0, 1), class = sigma),
      prior(normal(0, 1), class = sd),
      prior(lkj(2), class = cor)
    ),
  data = df_gg05_rc,
  control=list(adapt_delta=0.99, max_treedepth=15)
)
```

```
posterior_summary(fit_df_gg05_rc2, variable = "b_c_cond")
```

```
##          Estimate Est.Error    Q2.5  Q97.5
## b_cond  0.00425   0.00973 -0.0147 0.0233
```

And here we use the prior, $\beta \sim Normal(0, 1)$:

```
fit_df_gg05_rc3 <- 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),
      prior(normal(0, 1), class = sigma),
      prior(normal(0, 1), class = sd),
      prior(lkj(2), class = cor)
    ),
  data = df_gg05_rc,
  control=list(adapt_delta=0.99, max_treedepth=15)
)
```

```
posterior_summary(fit_df_gg05_rc3, variable = "b_c_cond")
```

```
##          Estimate Est.Error   Q2.5 Q97.5
## b_cond    0.122    0.0511 0.0197  0.22
```

We can summarize the estimates of β given different priors in the following way:

Estimate	Q2.5	Q97.5	Prior for β
0.098	0.004	0.187	Normal(0,.01)

Estimate	Q2.5	Q97.5	Prior for β
0.004	-0.015	0.023	Normal(0,.1)
0.122	0.020	0.220	Normal(0,1)

We can take a look at the estimate for the difference between conditions in milliseconds:

```
alpha <- as_draws_df(fit_df_gg05_rc)$b_Intercept
beta1 <- as_draws_df(fit_df_gg05_rc)$b_c_cond
diff1 <- exp(alpha + beta1/2) - exp(alpha - beta1/2)
c(mean = mean(diff1), quantile(diff1, c(.025, .975)))
```

```
##  mean  2.5% 97.5%
## 35.33  1.54 69.35
```

If we repeat this with each model, we can take a look at the effect of the prior on the difference between conditions:

Estimate (ms)	Q2.5	Q97.5	Prior for β
35.33	1.53	69.35	Normal(0,.01)
1.47	-5.07	8.12	Normal(0,.1)
44.05	6.72	82.59	Normal(0,1)

This sensitivity shows us that the posterior changes quite a lot under different priors; depending on what we consider to be a reasonable prior (i.e., depending on our prior beliefs), the inference that we derive from the model and data can vary quite a bit when the data are relatively sparse, as in this case.

Exercise 3

```
data("df_gibsonwu")
df_gibsonwu$cond<-ifelse(df_gibsonwu$type=="obj-ext",0.5,-0.5)

data("df_gibsonwu2")
df_gibsonwu2$cond<-ifelse(df_gibsonwu2$condition=="obj-ext",0.5,-0.5)
```

Analysis of Gibson and Wu 2013 data:

Normal likelihood:

```
priorsgwfullnormal <- c(set_prior("normal(500, 200)", class = "Intercept"),
            set_prior("normal(0,500)", class = "b",
                      coef = "cond"),
            set_prior("normal(0, 500)", class = "sd"),
            set_prior("normal(0, 1000)", class = "sigma"),
            set_prior("lkj(2)", class = "cor"))
```

```
m_gwfullnormal<-brm(rt~cond + (1+cond|subj)+(1+cond|item),
              family=gaussian(), prior=priorsgwfullnormal,
             warmup=1000,
             iter=2000,
             cores=4,
             data = df_gibsonwu)
```

Log-normal likelihood:

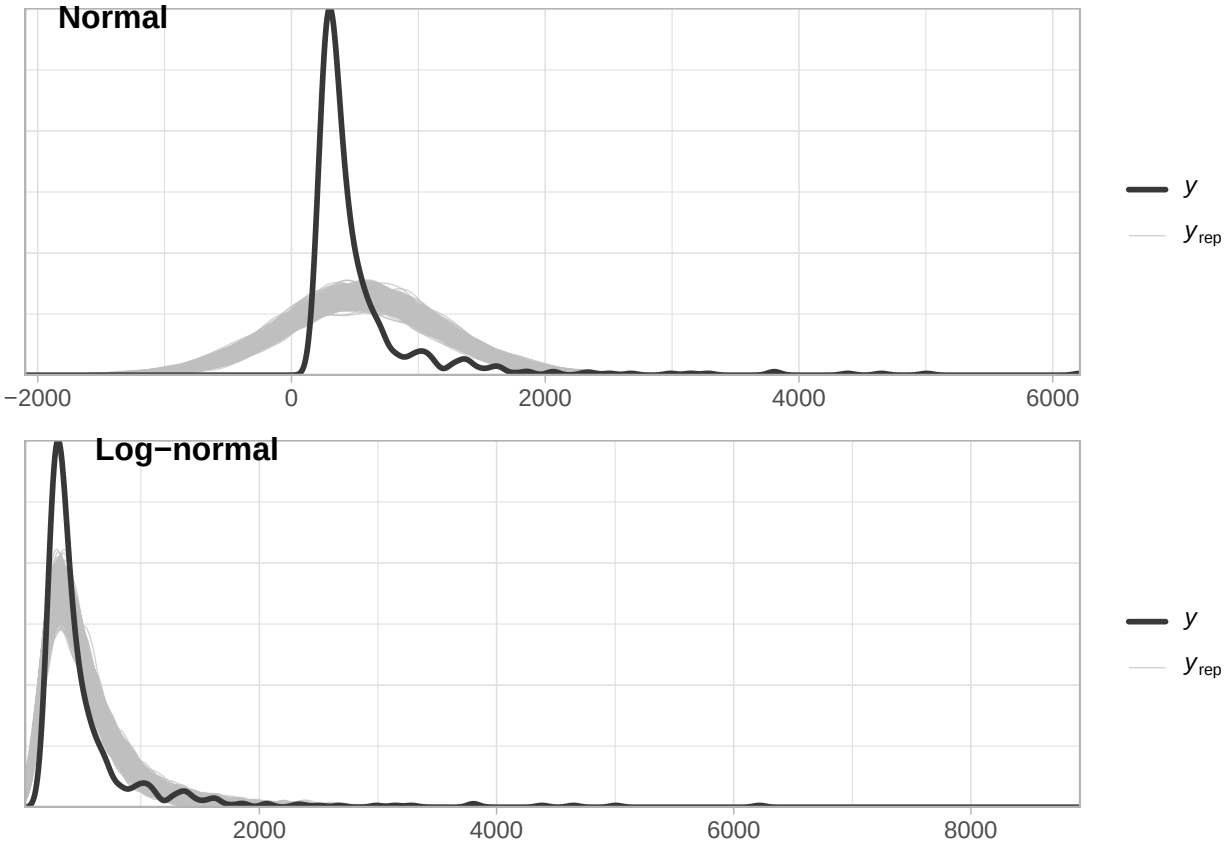
```
priorsgwfulllogn <- c(set_prior("normal(6, 0.6)", class = "Intercept"),
             set_prior("normal(0,1)", class = "b",
                       coef = "cond"),
             set_prior("normal(0, 1)", class = "sd"),
             set_prior("normal(0, 1)", class = "sigma"),
             set_prior("lkj(2)", class = "cor"))
```

```
m_gwfulllogn<-brm(rt~cond + (1+cond|subj)+(1+cond|item),
                  family=lognormal(), prior=priorsgwfulllogn,
                 warmup=1000,
                 iter=2000,
                 cores=4,
                 data = df_gibsonwu,
                 control=list(adapt_delta=0.99, max_treedepth=15))
```

(a) Posterior predictive distributions

Gibson and Wu (2013) data:

```
pp_gwnormal<-pp_check(m_gwfullnormal,ndraws=1000)
pp_gwlogn<-pp_check(m_gwfulllogn,ndraws=1000)
cowplot::plot_grid(pp_gwnormal,pp_gwlogn,labels=c("Normal","Log-normal"),label_size=12,ncol=1)
```

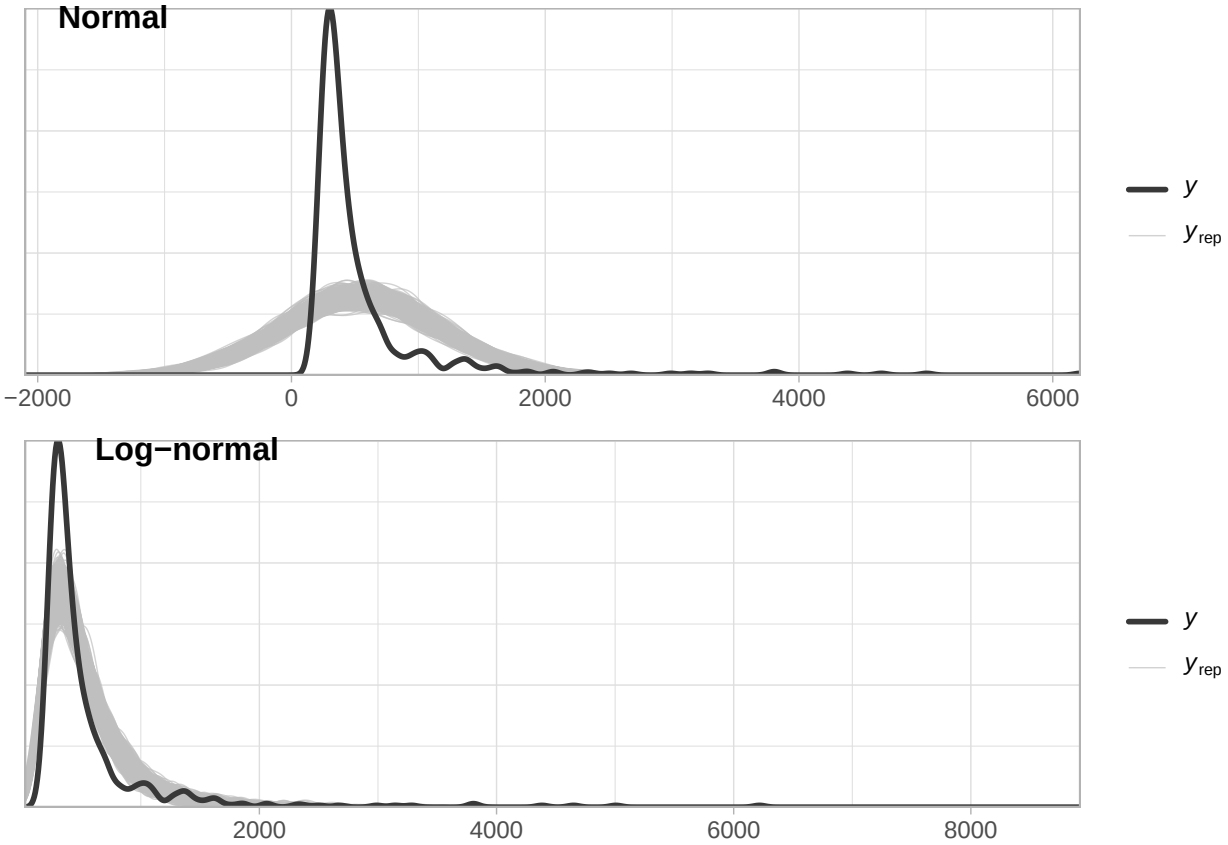


In the Gibson and Wu (2013) data, the normal likelihood leads to a pretty dramatic mismatch between the observed and posterior predictive values; negative values are generated. The log-normal has a better match.

The replication attempt of Gibson and Wu (2013):

```
m_gw2fullnormal<-brm(rt~cond + (1+cond|subj)+(1+cond|item),
                family=gaussian(), prior=priorsgwfullnormal,
                warmup=1000,
                iter=2000,
                cores=4,
                data = df_gibsonwu2)

m_gw2fulllogn<-brm(rt~cond + (1+cond|subj)+(1+cond|item),
                family=lognormal(), prior=priorsgwfulllogn,
                warmup=1000,
                iter=2000,
                cores=4,
                data = df_gibsonwu2)
```

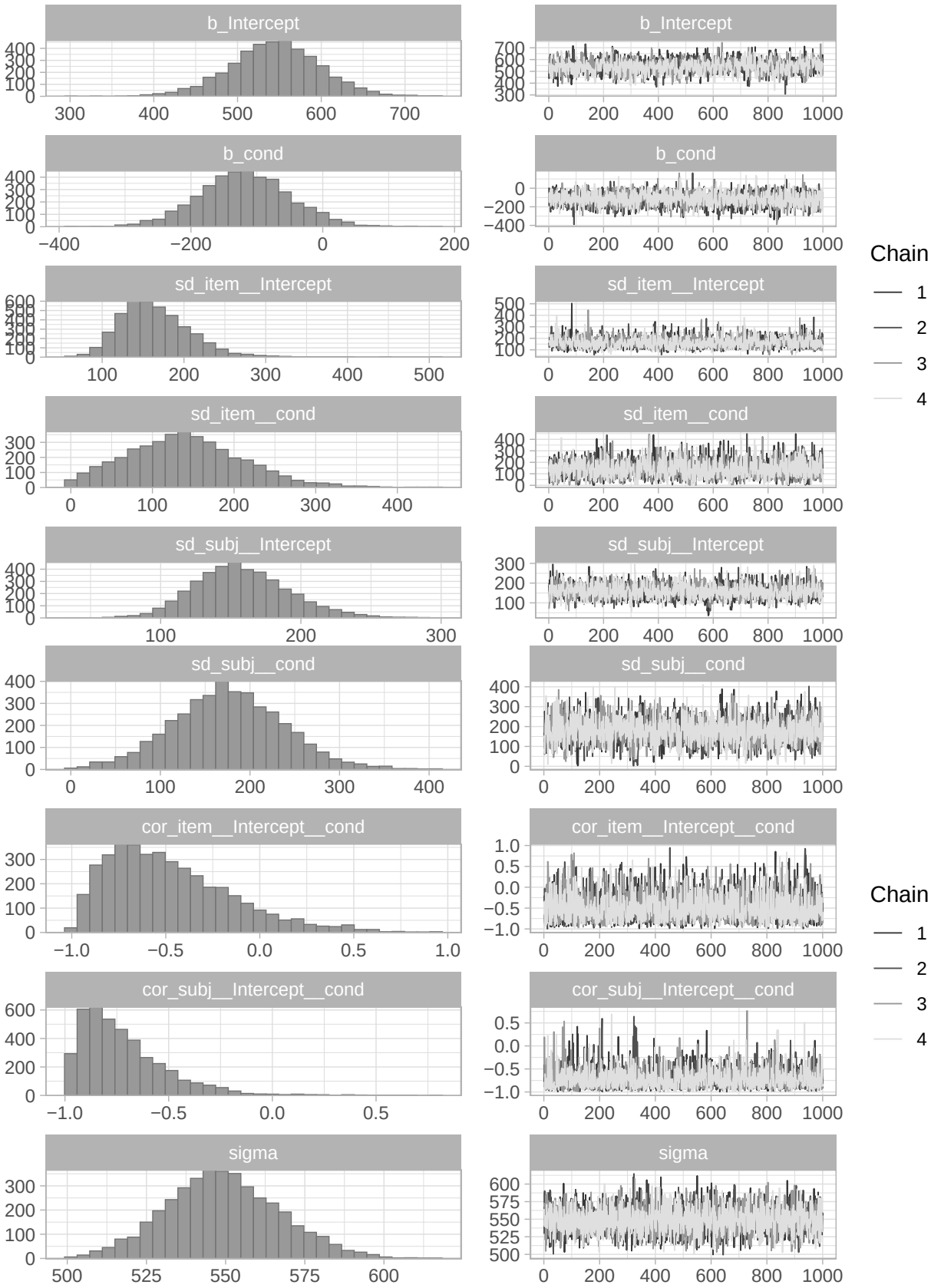
```
pp_gw2normal<-pp_check(m_gw2fullnormal,ndraws=1000)
pp_gw2logn<-pp_check(m_gw2fulllogn,ndraws=1000)
cowplot::plot_grid(pp_gwnormal,pp_gwlogn,labels=c("Normal","Log-normal"),label_size=12,ncol=1)
```



Here we have the same situation as in the first data set: assuming a normal likelihood yields a posterior predictive distribution that has negative values; the log-normal produces more realistic values.

(b) Gibson and Wu (2013) effect estimates:

```
plot(m_gwfullnormal)
```


b_Intercept
b_cond
sd_item__Intercept
sd_item__cond
sd_subj__Intercept
sd_subj__cond
cor_item__Intercept__cond
cor_subj__Intercept__cond
sigma
Chain
1
2
3
4

```
gw_intercept <- as_draws_df(m_gwfullnormal)$b_Intercept
gw_slope <- as_draws_df(m_gwfullnormal)$b_cond

gw_interceptln <- as_draws_df(m_gwfulllogn)$b_Intercept
gw_slopeln <- as_draws_df(m_gwfulllogn)$b_cond

gw_RT_diffnormal <- gw_slope
quantile(gw_RT_diffnormal,prob=c(0.025,0.975))
```

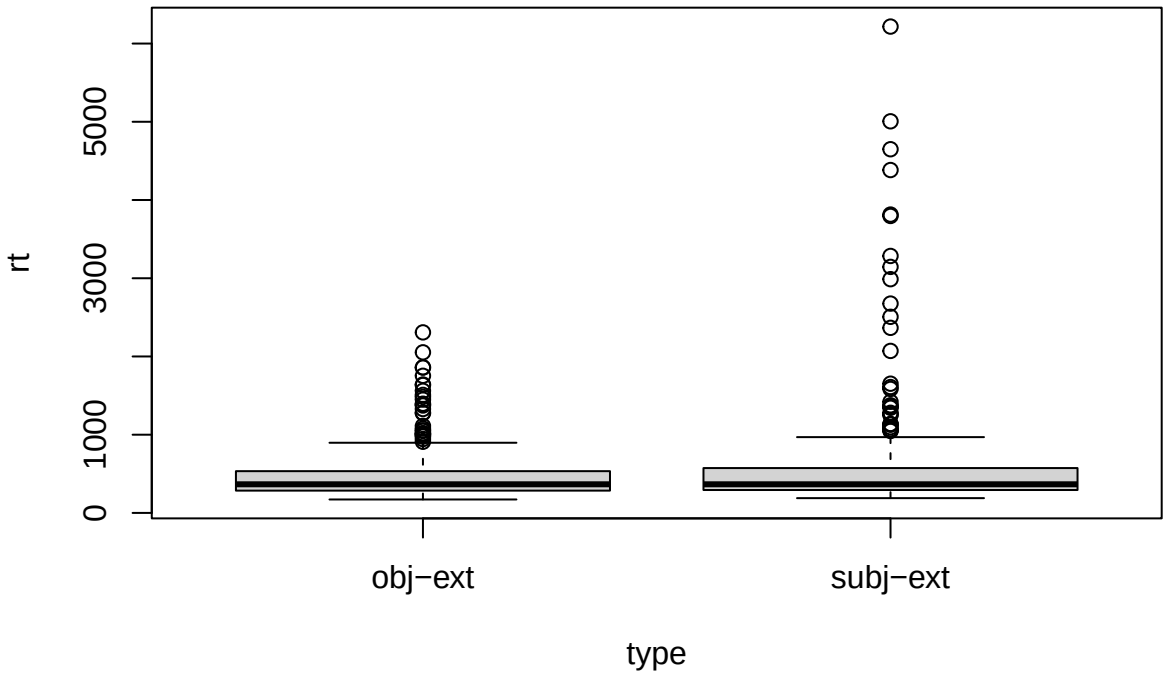
```
##   2.5%  97.5%
## -255.2   18.6
```

```
gw_RT_difflogn <- exp(gw_interceptln + gw_slopeln/2) -
    exp(gw_interceptln - gw_slopeln/2)

quantile(gw_RT_difflogn,prob=c(0.025,0.975))
```

```
##  2.5% 97.5%
## -81.0  16.4
```

The effect estimate is much smaller, and has a smaller 95% credible interval, with the log-normal likelihood. This is because there are a few extreme values in the data that bias the mean effect in subject relatives:

```
boxplot(rt~type,df_gibsonwu)
```



The same thing happens with the replication attempt of the Gibson and Wu design:

```
gw2_intercept <- as_draws_df(m_gw2fullnormal)$b_Intercept
gw2_slope <- as_draws_df(m_gw2fullnormal)$b_cond


gw2_interceptln <- as_draws_df(m_gw2fulllogn)$b_Intercept
gw2_slopeln <- as_draws_df(m_gw2fulllogn)$b_cond

gw2_RT_diffnormal <- gw2_slope
```

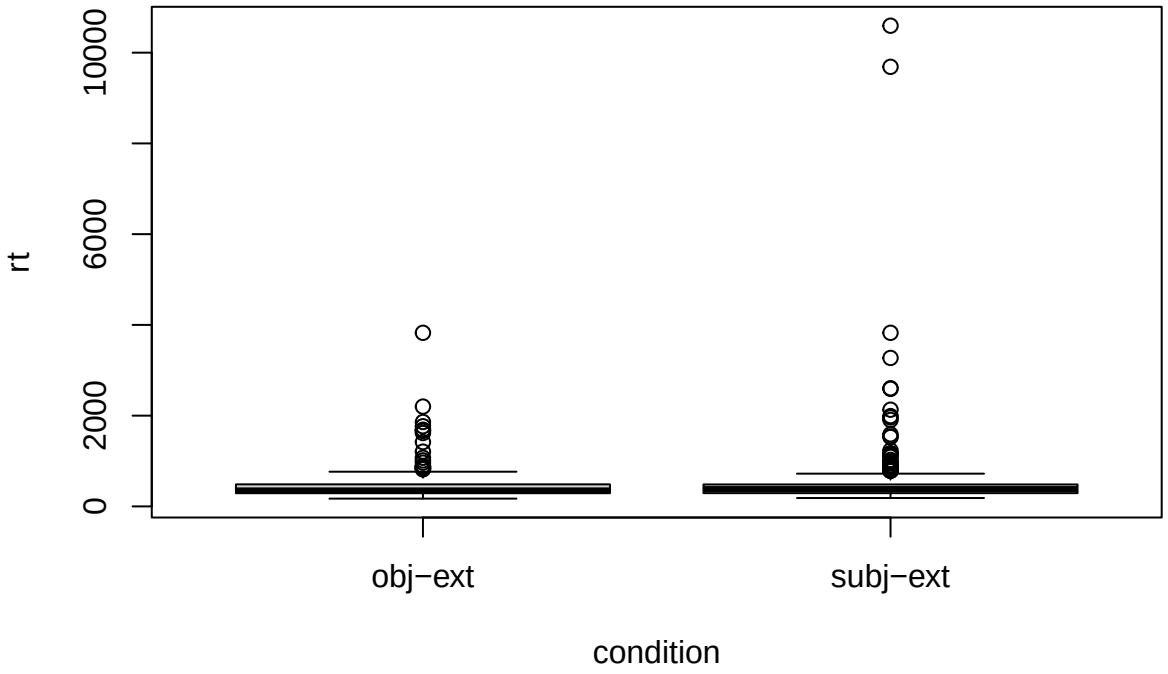
```r
quantile(gw2_RT_diffnormal,prob=c(0.025,0.975))
```

```
##   2.5%  97.5%
## -264.1   53.9
```

```r
gw2_RT_difflogn <- exp(gw2_interceptln + gw2_slopeln/2) -
    exp(gw2_interceptln - gw2_slopeln/2)

quantile(gw2_RT_difflogn,prob=c(0.025,0.975))
```

```
##  2.5% 97.5%
## -90.5  21.0
```

```r
boxplot(rt~condition,df_gibsonwu2)
```



(c) The log-normal is better because it yields more realistic posterior predictive data and the estimate of the effect is not unduly affected by a few extreme values.

Using an informative prior (the posterior of a preceding study) for the next study:

```r
summary(m_gw2fulllogn)
```

```
##  Family: lognormal
##   Links: mu = identity; sigma = identity
## Formula: rt ~ cond + (1 + cond | subj) + (1 + cond | item)
##    Data: df_gibsonwu2 (Number of observations: 595)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Multilevel Hyperparameters:
## ~item (Number of levels: 15)
##                     Estimate Est.Error l-95% CI u-95% CI Rhat
## sd(Intercept)           0.16      0.04     0.10     0.26 1.00
## sd(cond)                0.19      0.07     0.08     0.34 1.00
## cor(Intercept,cond)    -0.41      0.28    -0.86     0.22 1.00
```

```
##                     Bulk_ESS Tail_ESS
## sd(Intercept)           1347     1493
## sd(cond)                1157     1712
## cor(Intercept,cond)     2131     2252
##
## ~subj (Number of levels: 40)
##                     Estimate Est.Error l-95% CI u-95% CI Rhat
## sd(Intercept)           0.25      0.04     0.18     0.32 1.01
## sd(cond)                0.11      0.06     0.01     0.23 1.00
## cor(Intercept,cond)     0.01      0.35    -0.67     0.69 1.00
##                     Bulk_ESS Tail_ESS
## sd(Intercept)           1357     1933
## sd(cond)                 954     1490
## cor(Intercept,cond)     3214     2346
##
## Regression Coefficients:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     6.00      0.06     5.88     6.13 1.00     1006     1454
## cond         -0.08      0.07    -0.22     0.06 1.00     1837     2174
##
## Further Distributional Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.43      0.01     0.40     0.46 1.00     3169     2775
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Based on the posterior from the Gibson and Wu (2013) analysis, the posterior for the effect on the log scale is Normal(-0.08,0.065).

```
priorsgwfulllogninf <- c(set_prior("normal(6, 0.6)", class = "Intercept"),
             set_prior("normal(-0.08,0.065)", class = "b",
                       coef = "cond"),
             set_prior("normal(0, 1)", class = "sd"),
             set_prior("normal(0, 1)", class = "sigma"),
             set_prior("lkj(2)", class = "cor"))
```

```
m_gwfulllogninf<-brm(rt~cond + (1+cond|subj)+(1+cond|item),
              family=lognormal(), prior=priorsgwfulllogninf,
             warmup=1000,
             iter=2000,
             cores=4,
             data = df_gibsonwu2,
             control=list(adapt_delta=0.99, max_treedepth=15))
```

```
summary(m_gwfulllogninf)
```

```
##  Family: lognormal
##   Links: mu = identity; sigma = identity
## Formula: rt ~ cond + (1 + cond | subj) + (1 + cond | item)
##    Data: df_gibsonwu2 (Number of observations: 595)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Multilevel Hyperparameters:
```

```
## ~item (Number of levels: 15)
##                     Estimate Est.Error l-95% CI u-95% CI Rhat
## sd(Intercept)           0.16      0.04     0.10     0.26 1.00
## sd(cond)                0.18      0.06     0.07     0.32 1.00
## cor(Intercept,cond)    -0.42      0.28    -0.87     0.20 1.00
##                     Bulk_ESS Tail_ESS
## sd(Intercept)           1300     2159
## sd(cond)                1293     1324
## cor(Intercept,cond)     2595     2676
##
## ~subj (Number of levels: 40)
##                     Estimate Est.Error l-95% CI u-95% CI Rhat
## sd(Intercept)           0.25      0.04     0.18     0.32 1.00
## sd(cond)                0.10      0.06     0.01     0.23 1.00
## cor(Intercept,cond)     0.02      0.36    -0.71     0.71 1.00
##                     Bulk_ESS Tail_ESS
## sd(Intercept)           1297     2228
## sd(cond)                 999     1532
## cor(Intercept,cond)     4565     2422
##
## Regression Coefficients:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     6.00      0.06     5.88     6.12 1.00     1386     1759
## cond         -0.08      0.04    -0.17     0.00 1.00     3869     3091
##
## Further Distributional Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.43      0.01     0.40     0.46 1.00     4790     2888
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Next, we pool the data and fit the model again:

```
colnames(df_gibsonwu)
```

```
## [1] "subj" "item" "type" "rt"   "cond"
```

```
colnames(df_gibsonwu2)[3]<-"type"
df_gibsonwu2<-df_gibsonwu2[,c(1,2,3,5,7)]

df_gibsonwu$subj<-paste("gw",df_gibsonwu$subj,sep="")
df_gibsonwu2$subj<-paste("gw2",df_gibsonwu2$subj,sep="")
pooled_gw<-rbind(df_gibsonwu,df_gibsonwu2)

priorsgwfulllognuninf <- c(set_prior("normal(6, 0.6)", class = "Intercept"),
             set_prior("normal(0,1)",
                       class = "b",
                       coef = "cond"),
             set_prior("normal(0, 1)", class = "sd"),
             set_prior("normal(0, 1)", class = "sigma"),
             set_prior("lkj(2)", class = "cor"))
```

```
m_gwfulllognpooled<-brm(rt~cond + (1+cond|subj)+(1+cond|item),
               family=lognormal(), prior=priorsgwfulllognuninf,
```

```
              warmup=1000,
              iter=2000,
              cores=4,
              data = pooled_gw,
              control=list(adapt_delta=0.99, max_treedepth=15))
```

```
summary(m_gwfulllognpooled)
```

```
##  Family: lognormal
##   Links: mu = identity; sigma = identity
## Formula: rt ~ cond + (1 + cond | subj) + (1 + cond | item)
##    Data: pooled_gw (Number of observations: 1142)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Multilevel Hyperparameters:
## ~item (Number of levels: 15)
##                     Estimate Est.Error l-95% CI u-95% CI Rhat
## sd(Intercept)           0.18      0.04     0.12     0.28 1.00
## sd(cond)                0.06      0.04     0.00     0.15 1.00
## cor(Intercept,cond)    -0.23      0.40    -0.85     0.62 1.00
##                     Bulk_ESS Tail_ESS
## sd(Intercept)           1233     2311
## sd(cond)                1515     1863
## cor(Intercept,cond)     5916     2764
##
## ~subj (Number of levels: 77)
##                     Estimate Est.Error l-95% CI u-95% CI Rhat
## sd(Intercept)           0.25      0.03     0.20     0.30 1.00
## sd(cond)                0.10      0.05     0.01     0.20 1.00
## cor(Intercept,cond)    -0.32      0.30    -0.83     0.41 1.00
##                     Bulk_ESS Tail_ESS
## sd(Intercept)           1469     2192
## sd(cond)                1161     1625
## cor(Intercept,cond)     4402     2194
##
## Regression Coefficients:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     6.03      0.06     5.92     6.15 1.00     1178     1617
## cond         -0.08      0.04    -0.15    -0.01 1.00     4306     2753
##
## Further Distributional Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.48      0.01     0.46     0.50 1.00     5991     3118
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

We do get similar estimates by incrementally using the posterior from a previous study as a prior for a subsequent study, and from pooling the data.

Exercise 4

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

First we set the contrast coding so that:

$$\mu_{low} = \alpha + 1 \cdot \beta$$
$$\mu_{high} = \alpha + -1 \cdot \beta$$

```
dillonE1 <- dillonE1 %>%
mutate(int_meff = if_else(int == "low", 1, -1))
```

```
fit_dillonE1 <- brm(rt ~ int_meff + (int_meff | subj) + (int_meff | item),
  family = lognormal(),
  prior =
    c(
      prior(normal(6, 1.5), class = Intercept),
      prior(normal(0, 1), class = b),
      prior(normal(0, 1), class = sigma),
      prior(normal(0, 1), class = sd),
      prior(lkj(2), class = cor)
    ),
  data = dillonE1
)
```

```
print(fit_dillonE1)
```

```
##  Family: lognormal
##   Links: mu = identity; sigma = identity
## Formula: rt ~ int_meff + (int_meff | subj) + (int_meff | item)
##    Data: dillonE1 (Number of observations: 2855)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Multilevel Hyperparameters:
## ~item (Number of levels: 48)
##                          Estimate Est.Error l-95% CI u-95% CI Rhat
## sd(Intercept)                0.13      0.02     0.10     0.18 1.00
## sd(int_meff)                 0.05      0.02     0.01     0.09 1.00
## cor(Intercept,int_meff)      0.25      0.28    -0.32     0.75 1.00
##                          Bulk_ESS Tail_ESS
## sd(Intercept)                1723     2807
## sd(int_meff)                  897     1181
## cor(Intercept,int_meff)      2768     2517
##
## ~subj (Number of levels: 40)
##                          Estimate Est.Error l-95% CI u-95% CI Rhat
## sd(Intercept)                0.28      0.04     0.22     0.36 1.00
## sd(int_meff)                 0.06      0.02     0.01     0.09 1.00
## cor(Intercept,int_meff)      0.17      0.26    -0.34     0.68 1.00
##                          Bulk_ESS Tail_ESS
## sd(Intercept)                1241     2338
## sd(int_meff)                 1019     1047
```

```
## cor(Intercept,int_meff)     3853     2809
##
## Regression Coefficients:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     6.51      0.05     6.42     6.61 1.00      856     1522
## int_meff      0.03      0.02    -0.00     0.06 1.00     3937     3298
##
## Further Distributional Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.57      0.01     0.56     0.59 1.00     6100     3014
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
posterior_summary(fit_dillonE1, variable = "b_int_meff")
```

```
##            Estimate Est.Error     Q2.5  Q97.5
## b_int_meff   0.0287    0.0157 -0.00143 0.0601
```

There is some indication that the low interference conditions have slower rt, as predicted.

Exercise 5

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

```
fit_ab <- brm(probe_correct ~ lag + (lag | subj),
  family = bernoulli(),
  prior =
    c(
      prior(normal(6, 1.5), class = Intercept),
      prior(normal(0, 1), class = b),
      prior(normal(0, 1), class = sd),
      prior(lkj(2), class = cor)
    ),
  data = df_ab
)
```

```
fit_ab
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: probe_correct ~ lag + (lag | subj)
##    Data: df_ab (Number of observations: 2101)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Multilevel Hyperparameters:
## ~subj (Number of levels: 29)
##                     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## sd(Intercept)           0.60      0.14     0.37     0.90 1.00     2188
## sd(lag)                 0.08      0.03     0.02     0.15 1.00     1050
## cor(Intercept,lag)      0.34      0.31    -0.27     0.88 1.00     2523
##                     Tail_ESS
## sd(Intercept)           3015
## sd(lag)                 1450
```

```
## cor(Intercept,lag)     2370
##
## Regression Coefficients:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     0.29      0.15     0.02     0.58 1.00     2171     2394
## lag           0.02      0.02    -0.03     0.06 1.00     4345     3119
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

(a) How is the accuracy of the probe identification affected by the lag? Estimate this in log-odds and percentages.

In log odds:

```
posterior_summary(fit_ab, variable = "b_lag")
```

```
##       Estimate Est.Error    Q2.5  Q97.5
## b_lag    0.016    0.0246 -0.0316 0.0646
```

From lag 0 to lag 1 the accuracy increases by

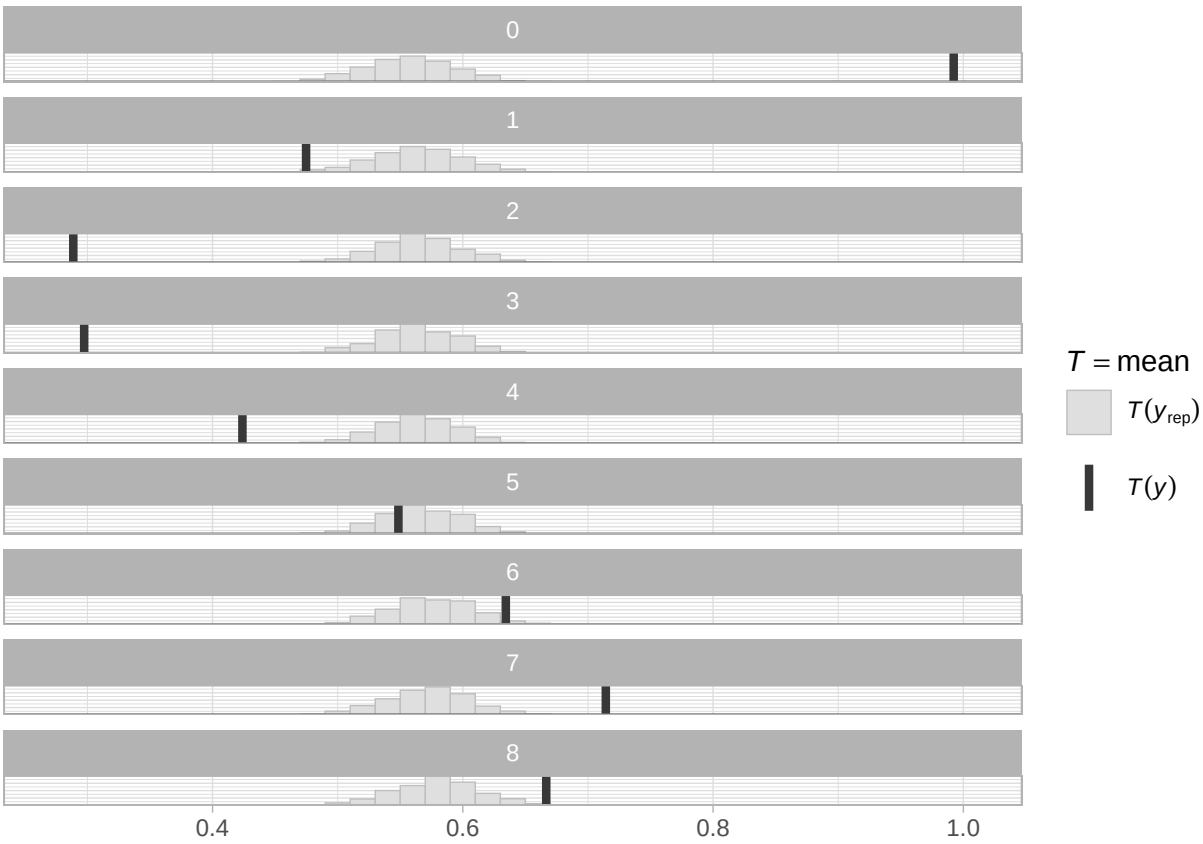
```
alpha_samples <- as_draws_df(fit_ab)$b_Intercept
beta_samples <- as_draws_df(fit_ab)$b_lag
effect_lag1_0 <- plogis(alpha_samples + 1* beta_samples) - plogis(alpha_samples)

c(mean = mean(effect_lag1_0),
quantile(effect_lag1_0, c(0.025, 0.975)))
```

```
##     mean     2.5%    97.5%
##  0.00392 -0.00781  0.01583
```

(b) Is the linear relationship justified? Use posterior predictive checks to verify this.

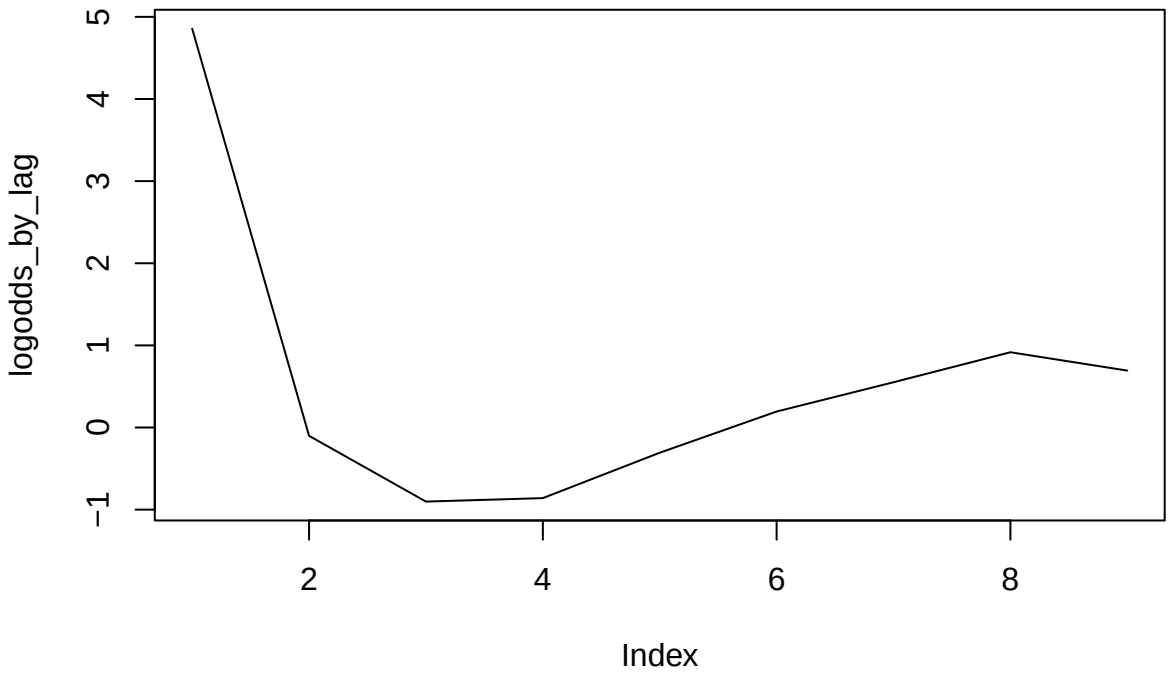
```
pp_check(fit_ab,
         type = "stat_grouped",
         stat = "mean",
         group = "lag",
         facet_args = list(
           ncol = 1, scales = "fixed"
           ),
         binwidth = 0.02
         )
```

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



Computing the mean log odds for each lag by hand:

```
probs_by_lag<-with(df_ab,tapply(probe_correct,lag,mean))

logodds_by_lag<-qlogis(probs_by_lag)

plot(logodds_by_lag,type="l")
```



It seems that below lag 3 (lag 0, lag 1, lag 2), lag has a negative effect on the log odds of detection, and from lag>=3, it reverses the lag increases the log-odds of detection. We see this in the plot (also see below): The average proportion of detections goes from 1 to 0 (in lags 0, 1, 2) in log odds space, and then it starts to increase again. A model that assumes only a monotonically increasing effect of lag can't take that into account.

This was not asked for in the exercise, but one can think about a better relationship between lag and accuracy. Fit a new model and use posterior predictive checks to verify if the fit improved.

The new model introduces a new term with an interaction to account for the U shape relationship (but the model is still linear).

- if lag ≥ 3, the model in log odds is reduced to

$\alpha + \beta_1 \times \text{lag}$

- if lag < 3, the model in log odds is

$\alpha + \beta_1 \times \text{lag} + \beta_2 \times 1 + \beta_3 \times \text{lag} = \alpha + \beta_2 + (\beta_1 + \beta_3) \times \text{lag}$

```
fit_ab2 <- brm(probe_correct ~ lag * I(lag < 3) + (lag * I(lag < 3)  | subj),
  family = bernoulli(link= logit),
  prior =
    c(prior(normal(0, 1.5), class = Intercept),
      prior(normal(0, 0.5), class = b, coef = lag),
      prior(normal(0, 1), class = sd, coef = Intercept, group = subj),
      prior(normal(0, 1), class = sd, coef = lag, group = subj),
      prior(lkj(2), class = cor, group = subj)
    ),
  iter = 2000,
  warmup = 1000,
  control = list(adapt_delta = 0.99),
  data = df_ab
)
```

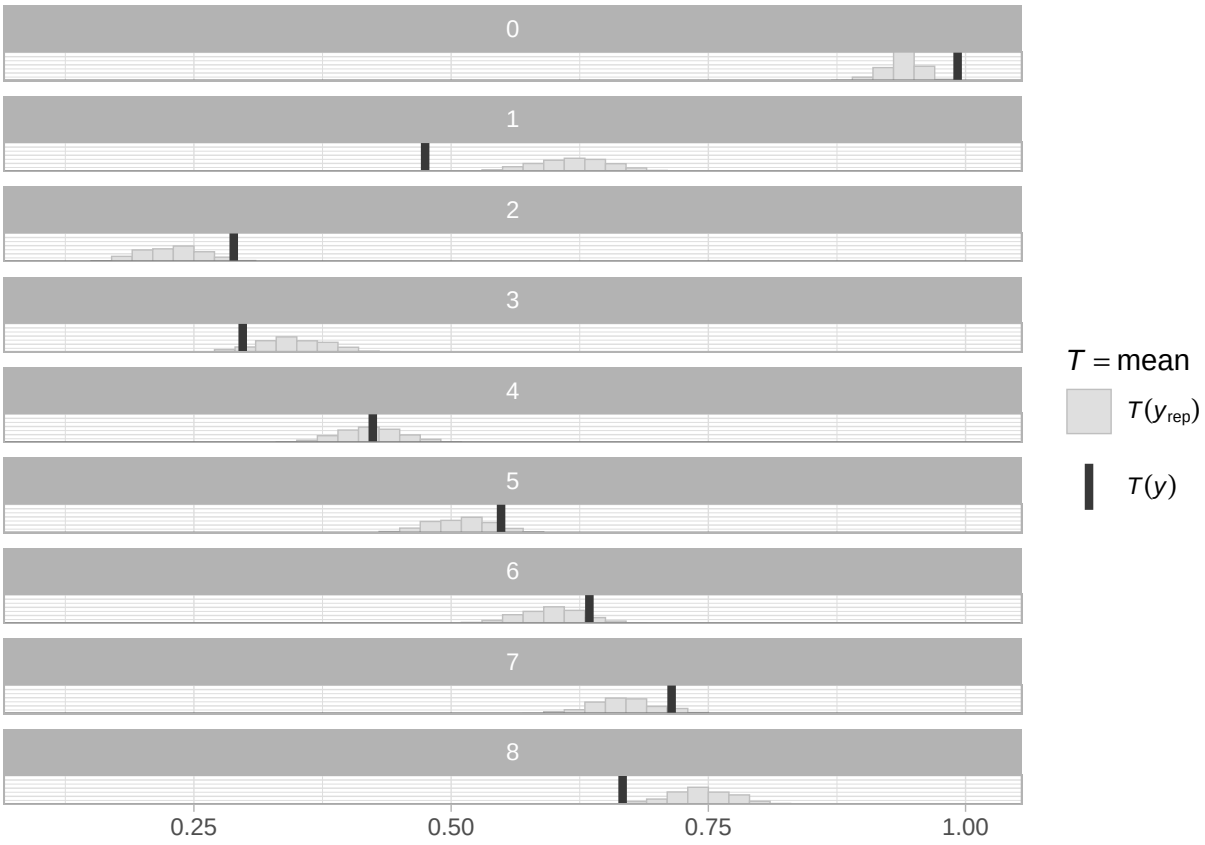
```
fit_ab2
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: probe_correct ~ lag * I(lag < 3) + (lag * I(lag < 3) | subj)
##    Data: df_ab (Number of observations: 2101)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Multilevel Hyperparameters:
## ~subj (Number of levels: 29)
##                                 Estimate Est.Error l-95% CI u-95% CI
## sd(Intercept)                       0.98      0.20     0.63     1.42
## sd(lag)                             0.06      0.04     0.00     0.16
## sd(Ilag<3TRUE)                      0.45      0.30     0.02     1.15
## sd(lag:Ilag<3TRUE)                  0.80      0.23     0.40     1.31
## cor(Intercept,lag)                 -0.10      0.36    -0.72     0.61
## cor(Intercept,Ilag<3TRUE)          -0.41      0.34    -0.89     0.42
## cor(lag,Ilag<3TRUE)                 0.12      0.38    -0.65     0.76
## cor(Intercept,lag:Ilag<3TRUE)       0.26      0.25    -0.23     0.70
## cor(lag,lag:Ilag<3TRUE)            -0.04      0.35    -0.69     0.63
## cor(Ilag<3TRUE,lag:Ilag<3TRUE)     -0.22      0.36    -0.79     0.55
##                                 Rhat Bulk_ESS Tail_ESS
## sd(Intercept)                   1.00     2592     2440
## sd(lag)                         1.00      485     1490
## sd(Ilag<3TRUE)                  1.00     1547     1782
## sd(lag:Ilag<3TRUE)              1.00     1478     2330
## cor(Intercept,lag)              1.00     3004     3100
## cor(Intercept,Ilag<3TRUE)       1.00     3368     3353
## cor(lag,Ilag<3TRUE)             1.00     3062     2996
## cor(Intercept,lag:Ilag<3TRUE)   1.00     2341     2880
## cor(lag,lag:Ilag<3TRUE)         1.00      881     1800
## cor(Ilag<3TRUE,lag:Ilag<3TRUE) 1.00      912     1388
##
## Regression Coefficients:
##                Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## Intercept         -1.92      0.28    -2.48    -1.39 1.00     2225
## lag                0.39      0.04     0.31     0.47 1.00     5163
## Ilag<3TRUE         4.94      0.33     4.31     5.61 1.00     4086
## lag:Ilag<3TRUE    -2.87      0.25    -3.37    -2.42 1.00     3393
##                Tail_ESS
## Intercept          2854
## lag                2376
## Ilag<3TRUE         3174
## lag:Ilag<3TRUE     3108
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
pp_check(fit_ab2,
         type = "stat_grouped",
         stat = "mean",
         group = "lag",
```

```
        facet_args = list(
          ncol = 1, scales = "fixed"
          ),
        binwidth = 0.02
        )
```

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



This seems to account for the observed pattern a bit better.