Computational Biology Group

D BSSE

ETH zürich

Introduction to Bayesian Statistics with R

6: Exercise solutions

Jack Kuipers

29 November 2022

First we load the tidyverse, brms and set a seed.

```
library(tidyverse); options(dplyr.summarise.inform = FALSE) # suppress summarise warnings
library(brms)
set.seed(42)
```

Exercise 6.1 - a fully Bayesian analysis

Take your analysis from Exercise 5.1 (of the ***lung_data.csv*** *from Exercise 1.1) and turn it into a robust t-test. Now to make the analysis fully Bayesian we should select our prior choices.*

- *Check which priors have already been set by default*
- *Input sensible priors, especially for the regression coefficients and intercept of σ.*
- *Check prior predictions*
- *Run the Bayesian analysis and discuss the output of interest.*

We plug the data

```
lung_data <- read.csv("./data/lung_data.csv")
```

straight into the robust t-test `brms` model from the lecture (using the helper function from exercises 5)

```
brmfit_t_ex6_r <- run_model(brm(bf(Lung.function ~ Trial.arm, sigma ~ Trial.arm),
  family = student, lung_data), "./brm_models_exercises/t_test_ex6_r")
```

First we want to check the priors

```
prior_summary(brmfit_t_ex6_r)[, -c(4:5, 7)] # hide some columns for display
```

```
##                   prior     class               coef  dpar lb ub  source
##                  (flat)         b                                   default
##                  (flat)         b Trial.armTreatment               default
##                  (flat)         b                     sigma         default
##                  (flat)         b Trial.armTreatment sigma         default
##  student_t(3, 10.3, 2.5) Intercept                                   default
##     student_t(3, 0, 2.5) Intercept                     sigma         default
##            gamma(2, 0.1)        nu                           1      default
```

and as in the lecture, we're not so keen on having flat priors for the coefficients of *Trial.arm* for both the main effect and the effect on σ, nor on the intercept of σ, with the log-link potentially leading to very small standard deviations. Since *Lung.function* is only measured to the nearest 0.1, we wouldn't expect standard

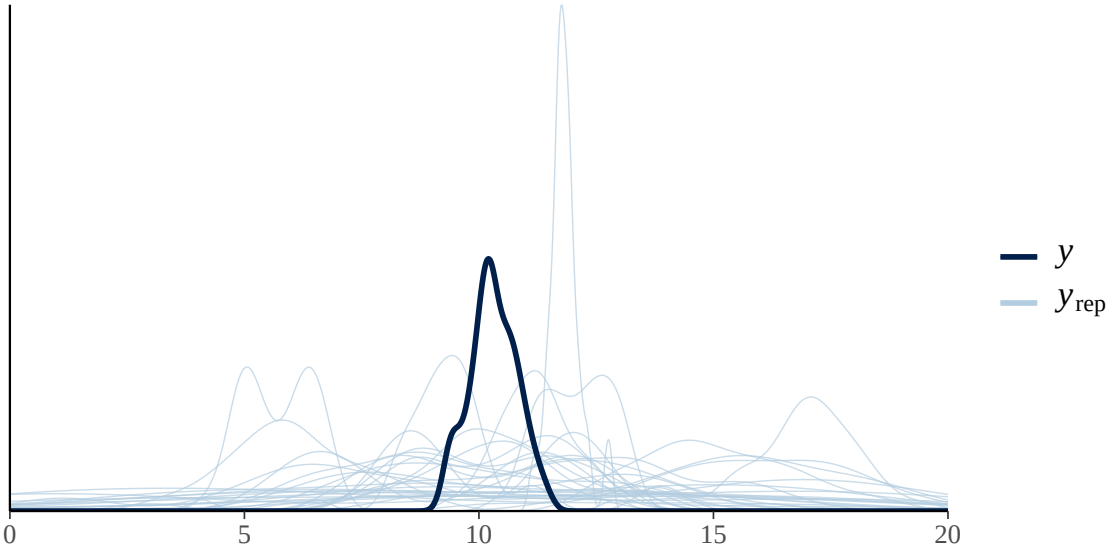
deviations to be small than that, so as a minimum we can impose a lower bound of around -2 (since with the log link $\exp(-2) \approx 0.1$). The drug we might hope to have an effect of around 1, suggesting a prior something like `student_t(3, 0, 2)`. We would expect the standard deviations to be quite similar in the two groups, so we could remove that effect completely (like the equal variance model) or have a narrow prior like `student_t(3, 0, 0.2)` as in the lecture (remember the log-link!). Adding the priors, we first sample from them

```
brmfit_t_ex6_prior <- run_model(brm(bf(Lung.function ~ Trial.arm, sigma ~ Trial.arm),
  prior = prior(student_t(3, 0, 2), class = "b", coef = "Trial.armTreatment") +
  prior(student_t(3, 0, 0.2), class = "b", coef = "Trial.armTreatment", dpar = "sigma") +
  prior(student_t(3, 0, 2.5), class = "Intercept", dpar = "sigma", lb = -2), # bound
  sample_prior = "only", family = student, lung_data),
  "./brm_models_exercises/t_test_ex6_prior")
prior_summary(brmfit_t_ex6_prior)[, -c(4:5, 7)] # hide some columns for display
```

```
##                     prior     class               coef  dpar lb ub  source
##                    (flat)         b                                 default
##       student_t(3, 0, 2)          b Trial.armTreatment                 user
##                    (flat)         b                     sigma       default
##     student_t(3, 0, 0.2)          b Trial.armTreatment sigma           user
##  student_t(3, 10.3, 2.5) Intercept                                   default
##     student_t(3, 0, 2.5) Intercept                     sigma -2        user
##            gamma(2, 0.1)        nu                           1     default
```

Now that the priors have been updated and we have run the model, we can check the prior predictions are vaguely in line with the data

```
pp_check(brmfit_t_ex6_prior, ndraws = 40) + xlim(0, 20) # plot predictive check
```



The spread of each prior sample looks reasonable, but sometimes is much too wide. Let's simply add an upper bound of 2 as well. The bigger problem is that the range of x values is far too wide seeing as all our data is around 10. This we can trace to the spread of *Intercept* values. Let's narrow that (and likewise for our main effect) and define our final model

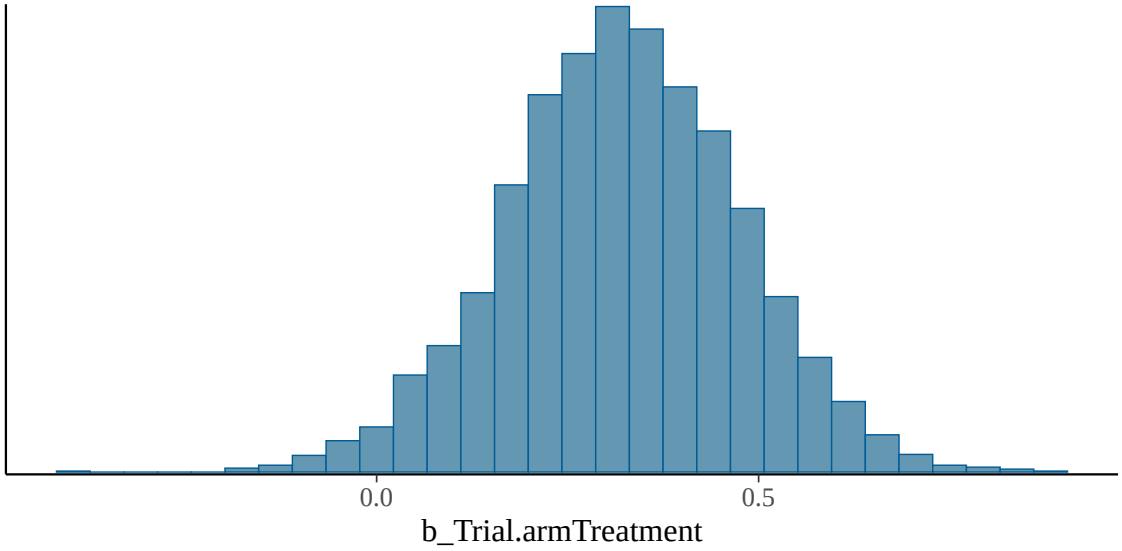
```
brmfit_t_ex6 <- run_model(brm(bf(Lung.function ~ Trial.arm, sigma ~ Trial.arm),
  prior = prior(student_t(3, 0, 1), class = "b", coef = "Trial.armTreatment") + # updated
  prior(student_t(3, 0, 0.2), class = "b", coef = "Trial.armTreatment", dpar = "sigma") +
  prior(student_t(3, 10.3, 1), class = "Intercept") + # new # bound both sides below
  prior(student_t(3, 0, 2.5), class = "Intercept", dpar = "sigma", lb = -2, ub = 2),
```

```
  family = student, lung_data), "./brm_models_exercises/t_test_ex6")
prior_summary(brmfit_t_ex6)[, -c(4:5, 7)] # hide some columns for display
```

```
##                  prior     class                coef  dpar lb ub  source
##                 (flat)         b                                   default
##     student_t(3, 0, 1)         b Trial.armTreatment                  user
##                 (flat)         b                      sigma        default
##   student_t(3, 0, 0.2)         b Trial.armTreatment sigma            user
##  student_t(3, 10.3, 1) Intercept                                     user
##   student_t(3, 0, 2.5) Intercept                      sigma -2  2     user
##          gamma(2, 0.1)        nu                            1     default
```

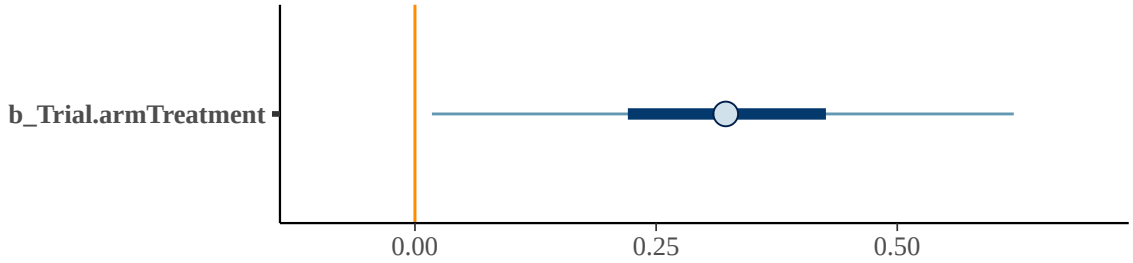
Above we just ran the model to sample from the posterior, and with that we can visualise the output for the coefficient of the `Trial.arm`.

```
mcmc_plot(brmfit_t_ex6, variable = "b_Trial.armTreatment", type = "hist")
```



Like in Exercises 5.1, the distribution looks some way from 0, but let's check the quantiles (setting `prob_outer = 0.95` to get the 95% CI)

```
mcmc_plot(brmfit_t_ex6, variable = "b_Trial.armTreatment", type = "intervals",
  prob_outer = 0.95) + geom_vline(xintercept = 0, color = "darkorange") + xlim(-0.1, 0.7)
```



As before, it just excludes 0.

Bonus Exercise 6.2 - confounding

The data from the previous exercise had unfortunately lost a column, namely the participant's Sex. *Read in the full data* `lung_data_all.csv` *and test for a difference in means between the two groups, adjusting to the participant's sex using* `lm`.

Can you see how to port the `lm` syntax into `brms` and run a Bayesian version of the same analysis?

After we load the complete data

```
lung_data_all <- read.csv("./data/lung_data_all.csv")
```

we can quickly check if the data design is balanced

```
lung_data_all %>% group_by(Trial.arm, Sex) %>%
  summarize(n=n()) %>%
  spread(Sex, n) %>%
  kable(col.names=c("", "Female", "Male"),
        caption = "Number of samples per sex and treatment group.")
```

Table 1: Number of samples per sex and treatment group.

	Female	Male
Control	17	3
Treatment	5	15

This is our first red flag, and when we analyse the data with `lm` we can see the issue more clearly

```
lung_data_all %>% lm(Lung.function ~ Trial.arm + Sex, .) %>% summary() %>% .$coefficients
```

```
##                      Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)        10.0303571 0.08178272 122.646409 6.717004e-50
## Trial.armTreatment -0.1535714 0.14006908  -1.096398 2.799897e-01
## SexM                0.7976190 0.14077472   5.665925 1.779974e-06
```

The *Treatment* is no longer significant, and if anything the effect is harmful with a decrease in *Lung.function*. Instead we see a large baseline difference between the *Sex* levels, which along with the unbalanced design has confounded our previous analyses.

For the Bayesian modelling, we will simplify by having a shared σ across all indications. In the syntax we set `sigma ~ 1` to keep the log link as before. We then try adding the *Sex* to the previous model, and having a common prior across all regression coefficients

```
brmfit_t_ex6_adj <- run_model(brm(bf(Lung.function ~ Trial.arm + Sex, sigma ~ 1),
  prior = prior(student_t(3, 0, 1), class = "b") + # sets for both beta
  prior(student_t(3, 10.3, 1), class = "Intercept") +
  prior(student_t(3, 0, 2.5), class = "Intercept", dpar = "sigma", lb = -2, ub = 2),
  family = student, lung_data_all), "./brm_models_exercises/t_test_ex6_adj")
```

Despite what the prior summary suggests

```
prior_summary(brmfit_t_ex6_adj)[, -c(4:5, 7)] # hide some columns for display
```

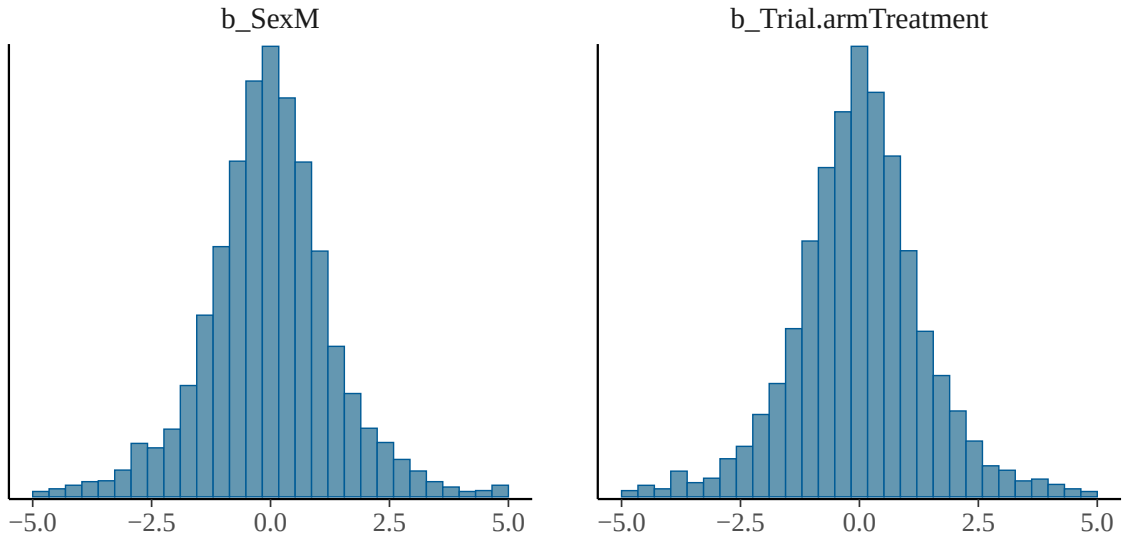
```
##                   prior     class                coef  dpar lb ub  source
##     student_t(3, 0, 1)         b                                      user
##                  (flat)         b               SexM              default
##                  (flat)         b Trial.armTreatment              default
##  student_t(3, 10.3, 1) Intercept                                      user
##   student_t(3, 0, 2.5) Intercept                     sigma -2  2      user
##          gamma(2, 0.1)        nu                           1     default
```

the prior on the regression coefficients has been set to the global class one, which we can check by sampling from the prior as follows

```
brmfit_t_ex6_adj_prior <- run_model(brm(bf(Lung.function ~ Trial.arm + Sex, sigma ~ 1),
  prior = prior(student_t(3, 0, 1), class = "b") + # sets for both beta
  prior(student_t(3, 10.3, 1), class = "Intercept") +
  prior(student_t(3, 0, 2.5), class = "Intercept", dpar = "sigma", lb = -2, ub = 2),
  sample_prior = "only", family = student, lung_data_all),
  "./brm_models_exercises/t_test_ex6_adj_prior")
```

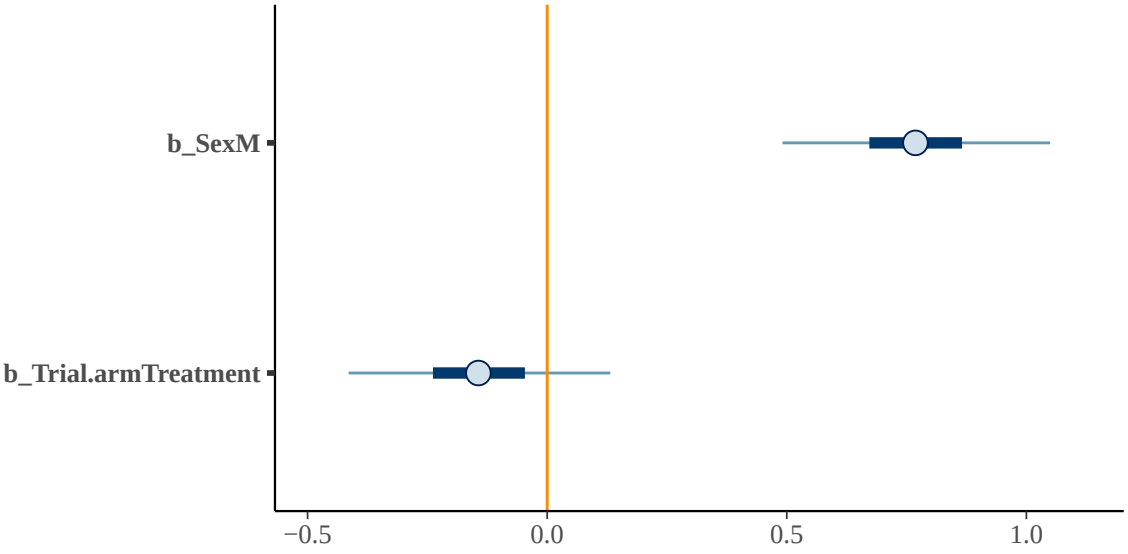
and plotting the prior distribution of the variables we care most about

```
mcmc_plot(brmfit_t_ex6_adj_prior, variable = paste0("b_", c("SexM", "Trial.armTreatment")),
  type = "hist") + xlim(-5, 5)
```



From the posterior, let's look at the same variables

```
mcmc_plot(brmfit_t_ex6_adj, variable = paste0("b_", c("SexM", "Trial.armTreatment")),
  type = "intervals", prob_outer = 0.95) + geom_vline(xintercept = 0, color = "darkorange")
```



and, in line with the `lm` results above, we see a negative or no effect from treatment and a clear effect of *Sex* that we now adjust for.