Assignment 7

Hierarchical model in Stan

anonymous

> ⚠️ **Warning**
>
> Currently, rendering on github is broken, such that the rendered template at https://avehtari.github.io/BDA_course_Aalto/assignments/template7.html looks weird. Rendering should however work on Aalto's JupyterLab, but we will also try to fix rendering on github ASAP.

1 General information

This is the template for assignment 7. You can download the separate model with bad priors and the qmd-file or copy the code from this rendered document after clicking on `</> Code` in the top right corner.

Please replace the instructions in this template by your own text, explaining what you are doing in each exercise.

2 Hierarchical Model: Chicken Data with Stan (6p)

2.1 Choosing a weakly informative prior by intuition

2.2 (a)

2.3 (b)

2.4 (c)

2.5 (d)

2.6 (e)

2.7 Choosing a weakly informative prior using external references

2.8 (f)

2.9 (g)

2.10 (h)

2.11 (i)

2.12 Non-normal priors

2.13 (j)

2.14 Modeling diet effects on chicken weight

> ❗ Data inside, don't peek before you have set your priors!
>
> > ❗ Have you set your priors?
> >
> > ```
> > data("ChickWeight")
> >
> > Chick12 <- ChickWeight |> filter(Time == 12)
> >
> > head(Chick12)
> > ```
> >
> > ```
> > Grouped Data: weight ~ Time | Chick
> >   weight Time Chick Diet
> > 1    106   12     1    1
> > 2    122   12     2    1
> > 3    115   12     3    1
> > 4    102   12     4    1
> > 5    141   12     5    1
> > 6    141   12     6    1
> > ```

2.15 (k)

2.16 (l)

For the figures below, we use the earlier draws for the separate model with bad priors. When you have implemented the pooled and hierarchical models, edit the code below to include draws from your model posterior into the figures.

2.17 (m)

```
ggplot(posterior_mean_diet_4, aes(x = mean_diet_4, y = model_name)) +
  stat_dotsinterval(quantiles = 100, scale = .9) +
  vline_at(diet_means[4], size = 1, linetype = "dashed") +
  # Annotate the vline from above.
  annotate("text", label = "Observation mean", x = diet_means[4] - 5, y = .7,
           hjust = "right", size = 6) +
  # Add title and axis labels. One line to make everything so much more clear!
  labs(
    title = "Mean of diet 4",
    x = "Weight (g)",
    y = "Model"
  )
```

Error in ggplot(posterior_mean_diet_4, aes(x = mean_diet_4, y = model_name)): object 'posterior_mean_d

2.18 (n)

```
ggplot(predicted_weight_diet_4, aes(x = predicted_weight, y = model_name)) +
  stat_dotsinterval(quantiles = 100, scale = .9) +
  vline_at(diet_means[4], size = 1, linetype = "dashed") +
  # Annotate the vline from above.
  annotate("text", label = "Observation mean", x = diet_means[4] - 5, y = .7,
           hjust = "right", size = 6) +
  # Add title and axis labels. One line to make everything so much more clear!
  labs(
    title = "Weigth of a chick with diet 4",
    x = "Weight (g)",
    y = "Model"
  )
```

Error in ggplot(predicted_weight_diet_4, aes(x = predicted_weight, y = model_name)): object 'predicted

2.19 (o)

```
ggplot(posterior_mean_diet_5, aes(x = mean_diet_5, y = model_name)) +
  # Draw the mean of each diet from the data as a dashed vertical line.
  vline_at(diet_means, size = .5, linetype = "dashed") +
  # dotsinterval gives mean, 50%, and 90% intervals + dotsplot with each dot
```

```
    # representing 1% of data (quantiles = 100).
    stat_dotsinterval(quantiles = 100, scale = .9) +
    # Annotate the vline from above.
    annotate(geom = "text", label = "Means of observed diets", y = .7, x = 100,
             hjust = "right", size = 5, family = "sans") +
    # Add title and axis labels. One line to make everything so much more clear!
    labs(title = "Mean of a new diet",
         x = "Weight (g)",
         y = "Model")
```

Error in ggplot(posterior_mean_diet_5, aes(x = mean_diet_5, y = model_name)): object 'posterior_mean_d

2.20 (p)

3 Hierarchical model with BRMS (3p)

3.1 (a)

```
bayesplot::mcmc_scatter(x = fit_hierarchical$draws(variables = c("mean_diet[4]", "sd_diets")),
                        np = nuts_params(fit_hierarchical)) +
    scale_y_log10() +
    labs(x = expression(mean_diet[4]), y = expression(sd_diets)) +
    ylim(c(0,NA))
```

Error: No chains finished successfully. Unable to retrieve the draws.

3.2 (b)

Create a brms model and sample from the posterior

```
brms_fit = brm(
  weight ~ 1 + (1 | Diet),
  data=Chick12,
  prior=c(
    # REPLACE WITH YOUR PRIOR DERIVED in 2)
    prior(normal(0,10), class="Intercept"),
  ),
  backend = "cmdstanr",
  save_pars = save_pars(manual = c("z_1[1,4]"))
)
```

Error in c(prior(normal(0, 10), class = "Intercept"),): argument 2 is empty

3.3 (c)

```
# Draws for mu_4
mu_4 = posterior_epred(brms_fit, newdata = data.frame(Diet=4))
```

Error in posterior_epred(brms_fit, newdata = data.frame(Diet = 4)): object 'brms_fit' not found

```
# Compute the mean, and quantiles. Remember to round your answers accordingly.
# ...
```

3.4 (d)

Due the non-centered parametrization, we need to transform compute the μ_d term as the sum of the population intercept and the group specific deviation from the intercept. You can choose which diet to plot by modifying the `d` integer in `r_Diet[d,Intercept]`.

```
draws = as_draws_df(brms_fit) |>
  posterior::mutate_variables(mean_diet_4 = `r_Diet[4,Intercept]` + b_Intercept)
```

Error in as_draws_df(brms_fit): object 'brms_fit' not found

```
bayesplot::mcmc_scatter(draws,
                        pars = c("mean_diet_4", "sd_Diet__Intercept"),
                        np = nuts_params(brms_fit)) +
  scale_y_log10() +
  xlab(expression(mean_diet[4])) +
  ylab(expression(sd_diets))
```

Error in posterior::is_draws(x): object 'draws' not found

3.5 (e)