Assignment 7

Hierarchical model in Stan

anonymous

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

1.1 Choosing a weakly informative prior by intuition

1.1.1 (a)

Based on the size of chickens and weights when buying groceries I would say a fully grown chicken ca be between 2000 and 10000 grams. Not sure really about the top boundary.

1.1.2 (b)

I don't know anything about chicken growth process but I would say it tend to increase more at the end, so I will adjust heavily my estimate to 500 grams to 3000 grams. I that case I would pick a μ of 1750 grams.

1.1.3 (c)

Following my previous answer in b) I would say the boundaries could be 250 grams and 5000 grams. I'm extending the range to avoid being overconfident and I don't know the data so maybe there are chicken with unexpected values. ### (d) If I think of a plausible effect of 200 grams of difference, then augmenting one order of magnitude would be 2000 grams giving a standard deviation of 2000gr,

1.1.4 (e)

$\mu_0 \; N(1750, 2000)$

1.2 Choosing a weakly informative prior using external references

1.2.1 (f)

This producer claims it has organic farm chickens and for fully grown one it estimates a range of 3 to 7 pounds, that's 1360 grams to 3175. For twelve days we could say something like 340 grams 1000 grams. The adjustment is roughly dividing by 4 and increasing a bit the upper limit to be less confident. We divide by 4 because I'm assuming some kind of exponential growth in the last days.

1.2.2 (g)

If the range provided and hence the adjusted ones are for all the chickens then the mean could be the middle value of the range, being 670 grams.

1.2.3 (h)

Using the proposed logic, we need to find the deviation that, using 670 as the mean will return that the cumulative probability up to the upper boundary 1000 is 99.7%

```r
pnorm(1000, 670, 120)
```

```
[1] 0.9970202
```

We end up with a standard deviation for the prior of 120.

1.2.4 (i)

$\mu_0 \; N(670, 120)$

1.3 Non-normal priors

1.3.1 (j)

Normal priors can make no sense in the case of variables that are bounded (since Normal goes from -inf to inf). For example, the variance prior should probably not follow a normal distribution. Another example could be parameters that need to be from 0 to 1 for example.

1.4 Modeling diet effects on chicken weight

❗ Data inside, don't peek before you have set your priors!

❗ Have you set your priors?

```r
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
```

1.4.1 (k)

Pooled model:

$$w_{ij}\ N(\mu, \sigma)$$

$$\mu\ N(670, 120)$$

$$\sigma\ Uniform(0, 100)$$

Hierarchical Model:

$$w_{ij}\ N(\mu_j, \sigma_j)$$

$$\mu_j\ N(\mu_0, sigma_0)$$

$$\log(\sigma)\ Uniform(0, 100)$$

$$\mu_0\ N(\mu, \tau)$$

$$\mu\ Uniform(0, 800)$$

$$\tau\ Uniform(0, 100)$$

The pooled model states that each chicken weight comes from a common normal distribution for all of them, no matter the diet. That Normal distribution has priors for mu and sigma.

The hierarchical models instead states that each chicken weight comes from a normal distribution that has a different mean per diet. All diets share a common variance. And then we state that each of those group diet means is not estimated independently from the others but we assume that each diet mean comes from a "super population" of means, that is itself also a Normal. So, each diet mean is a sample from this Normal distribution and not completely independent from the others as in the no pooled model. This super population normal has uniform priors for both the mean and the variance.

1.4.2 (I)

I had to play a bit with priors to make it work, specially to have logical results for the pooled and hierarchical model.My priors based on data were really off.
Also, the separate model has been kept with the bad priors.
I tried modifying priors for the separate model and I got convincing results -> mean for diet 4 in the hierarchical model being something in between the mean for the pooled model and the separate one.
With the current bad priors that doesn't happen because the separate model is really shrunk towards the prior.

Pooled Model:

```
data {
  int<lower=0> N_observations;
  int<lower=0> N_diets;
  array[N_observations] int diet_idx; // Pair observations to their diets.
  vector[N_observations] weight;
}

parameters {
  // Average weight of chicks no matter the diet
  real mean_diet;

  // Standard deviation of weights observed among chicks no matter the diet.
  real<lower=0> sd_diet;
}

model {
  // Priors
  // These look bad. I need to think about these again.

    mean_diet ~ normal(670, 120);
    sd_diet ~ uniform(0,100);

  // Likelihood
    weight ~ normal(mean_diet, sd_diet);
  // Best practice would be to write the likelihood without the for loop as:
  // weight ~ normal(mean_diet[diet_idx], sd_diet[diet_idx]);
}

generated quantities {
  real weight_pred;
  real mean_five;

  // Sample from the (posterior) predictive distribution
  weight_pred = normal_rng(mean_diet, sd_diet);
}
```

Hierarchical Model:

```
data {
  int<lower=0> N_observations;
  int<lower=0> N_diets;
  array[N_observations] int diet_idx; // Pair observations to their diets.
  vector[N_observations] weight;
}

parameters {
  // Average weight of chicks with a given diet.
  vector[N_diets] mean_diet;

  // Standard deviation of weights observed among chicks sharing a diet. Shared deviation for all diet
  real<lower=0> sd_diet;

  real<lower=0> mean_general;
  real<lower=0> sd_general;

}

model {
  // Priors
  // These look bad. I need to think about these again.

    mean_general ~ uniform(0,800);
    sd_general ~ uniform(0,100);
    sd_diet ~ uniform(0,100);

  for (diet in 1:N_diets) {
    mean_diet[diet] ~ normal(mean_general, sd_general);
  }

  // Likelihood
  for (obs in 1:N_observations) {
    weight[obs] ~ normal(mean_diet[diet_idx[obs]], sd_diet);
  }

  // Best practice would be to write the likelihood without the for loop as:
  // weight ~ normal(mean_diet[diet_idx], sd_diet[diet_idx]);
}

generated quantities {
  real weight_pred;
  real mean_five;

  // Sample from the (posterior) predictive distribution of the fourth diet.
  weight_pred = normal_rng(mean_diet[4], sd_diet);

  // Construct samples of the mean of the fifth diet.
  // We only have the prior...
  mean_five = normal_rng(0, 10);
}
```

1.4.3 (m)

Ideally the hierarchical model should be somewhere in between the separate and pooled model but it's not the case because of the bad priors on the separate model. Apart from that we can see greater dispersion in the posterior for the separate model, a bit less in the hierarchical one and much less in the pooled one.
The pooled model is not as close as the hierarchical one to the actual mean diet, which makes sense based on the pooling (other diets play a larger role).
The separate model is clearly wrong as it should be centered close to the actual mean but again, the priors are too strong for the data avaialable.

```r
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"
  )
```

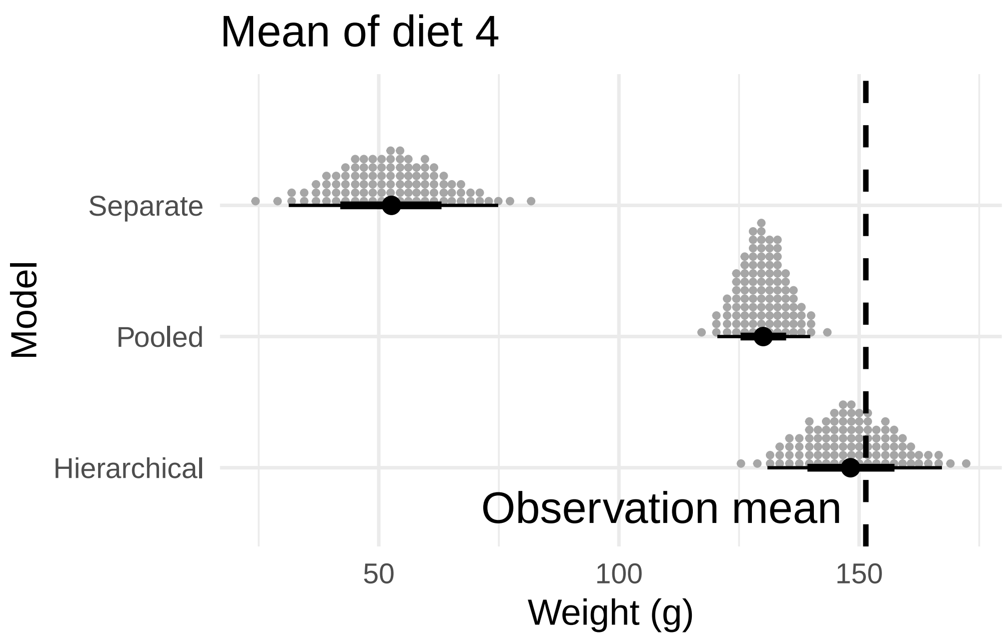


Figure 1: Posterior distribution of the mean weight of chicks consuming diet 4.

1.4.4 (n)

The predictive distribution for the diet 4 overlaps much more the different posteriors. The mean is the same as in the previous plot but we can see much larger dispersion on the possible values. Even the separate model has some draws close to the actual mean, but also some negative draws…
The distributions have larger tails admitting some "far" away draws but most of

the mass for both the pooled and hierarchical model are in a decent range between ~100 and 200 grams and no negative values.

```r
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"
  )
```

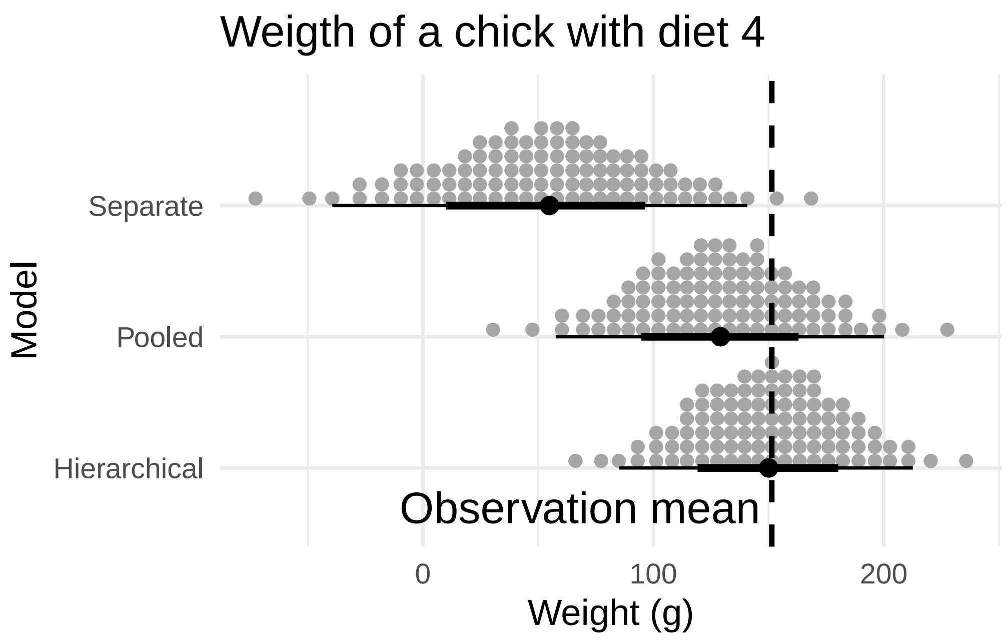


Figure 2: The (posterior) predictive distribution of the weigth of a chick consuming diet 4.

1.4.5 (o)

The separate model has no data to estimate a new fifth diet and so we only can sample from the prior. In this case, the bad prior, a Normal(0,10) as it is shown in the plot.
For the pooled model we sample from the pooled distribution of means, the same we did for diet 4. All diets share a distribution.
For the hierarchical model we can sample a new diet mean from the super population.
It's a pretty wide distribution. The mode and most part of the mass is above 100 grams which is fine but the tails are heavy and there are extreme value, including a few negative draws.

```r
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")
```

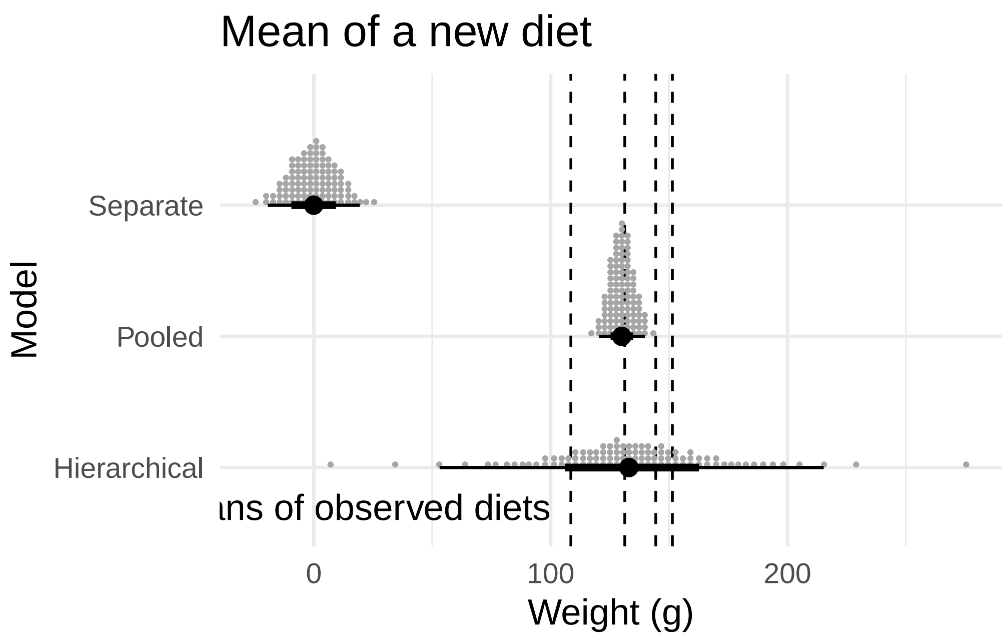


Figure 3: Posterior distribution of the mean weight of chicks consuming the new diet 5 not seen before.

1.4.6 (p)

Separate model For the separate model, the 90% credible interval of the mean of the diet 4 is the following:

```r
model  = posterior_mean_diet_4 %>% filter(model_name == "Separate" )

quantile(model$mean_diet_4,probs=c(0.05, 0.95))
```

```
      5%      95%
34.44233 70.96840
```

Pooled model For the pooled model, the 90% credible interval of the mean of the diet 4 is the following:

```r
model  = posterior_mean_diet_4 %>% filter(model_name == "Pooled" )

quantile(model$mean_diet_4,probs=c(0.05, 0.95))
```

```
      5%      95%
121.9527 138.4072
```

Hierarchical model For the Hierarchical model, the 90% credible interval of the mean of the diet 4 is the following:

```r
model  = posterior_mean_diet_4 %>% filter(model_name == "Hierarchical" )

quantile(model$mean_diet_4,probs=c(0.05, 0.95))
```

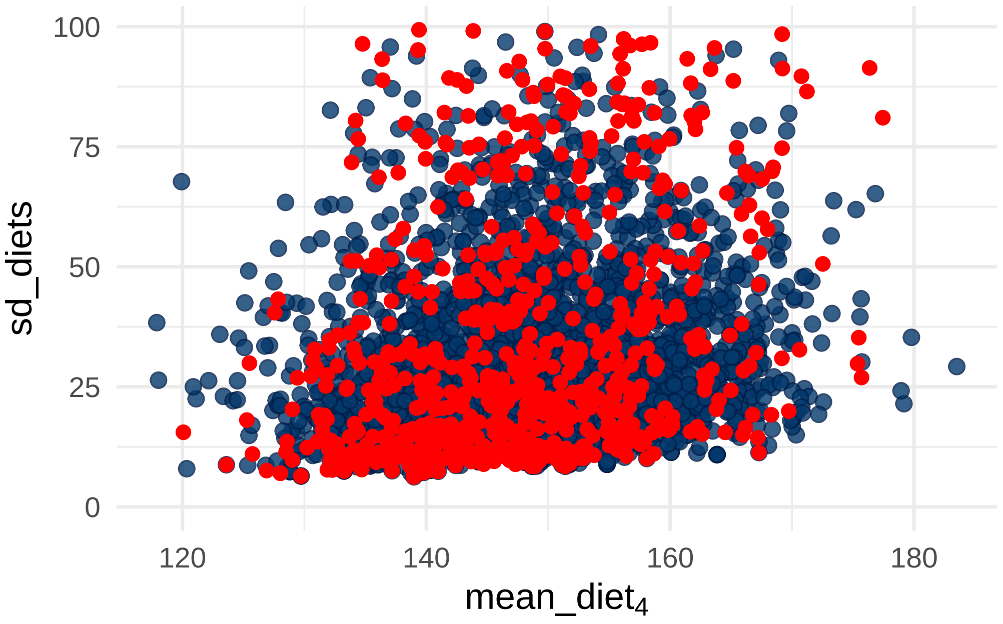
```
      5%      95%
133.2475 164.2916
```

2 Hierarchical model with BRMS (3p)

2.0.1 (a)

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

```
Scale for y is already present.
Adding another scale for y, which will replace the existing scale.
```



There is plenty of possible divergent transitions, all over the scatterplot and not in a particular region.

2.0.2 (b)

Because `brms` is a bit chatty, suppress it's output in the PDF using the block above, but copy the code you executed into the code block below, which doesn't execute:

```r
brms_fit = brm(
  weight ~ 1 + (1 | Diet),
  data=Chick12,
  prior=c(
    # REPLACE WITH YOUR PRIOR FOR THE INTERCEPT
    prior(uniform(0,800), class="Intercept"), # prior for mu_0
    # REPLACE WITH YOUR PRIOR FOR SIGMA
    prior(uniform(0,100), class="sigma"),     # prior for sigma
    # REPLACE WITH YOUR PRIOR FOR SD
    prior(uniform(0,100), class="sd")         # prior for tau
  ),
  backend = "cmdstanr",
  save_pars = save_pars(manual = c("z_1[1,4]"))
)
```

2.0.3 (c)

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

# To display accurate digits
# MCSE for the mean
print(posterior::mcse_mean(mu_4))
```

```
[1] 0.1865902
```

```r
# To display accurate digits
# MCSE for the quantiles
print(posterior::mcse_quantile(mu_4,probs=c(0.05, 0.95)))
```

```
 mcse_q5 mcse_q95
0.517795 0.277440
```

```r
# Compute the mean, and quantiles. Remember to round your answers accordingly.
# ...
print(paste0("mean: ", round(mean(mu_4),0)))
```

```
[1] "mean: 148"
```

```r
print("90% credible interval:")
```

```
[1] "90% credible interval:"
```

```r
print(round(quantile(mu_4,probs=c(0.05, 0.95),0)))
```

```
 5% 95%
133 163
```

The results are virtually the same. I got divergent chains even with brms. Maybe it's a problem with the priors used? Not sure really.

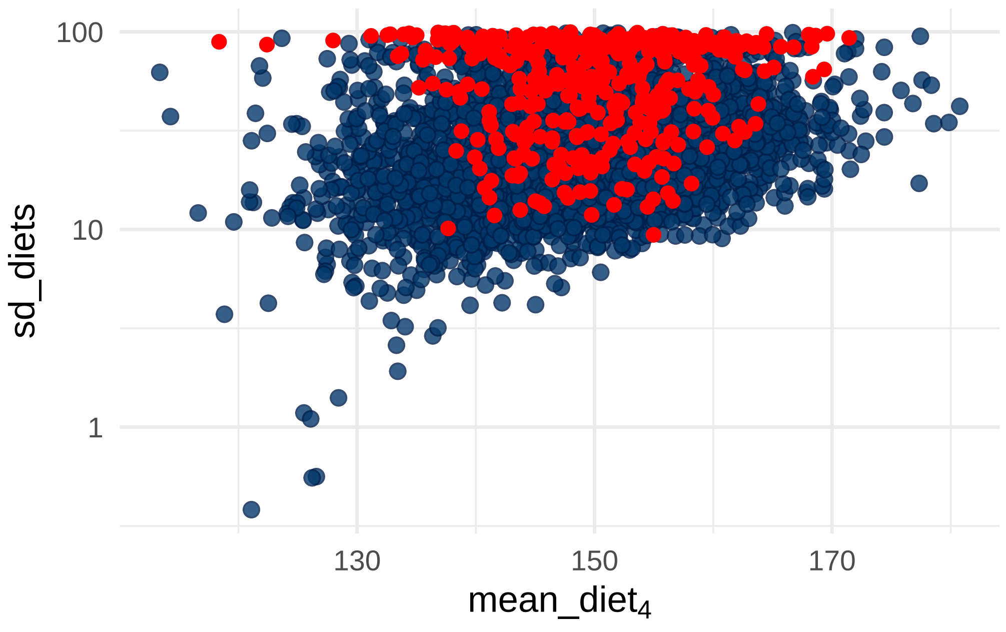
2.0.4 (d)

2.0.5 Scatterplot for non-centered parametrization

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]`.

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

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))
```



Still a lot of divergences although they seem more localized in the upper bound of the standard deviations. Still across all the values of the mean diet.

2.0.6 (e)

In the current run , plain Stan has fewer divergent transitions but I think that's pretty variable. I saw different values quite different, from 16% to 40%.

In brms the divergent transitions occur much more in the higher values of Tau while in Stan they occurred more spread but with highest chance in the lower values of Tau.

The centered parametrization had problems sampling all over the parameter space although specially in the lower values of Tau.
The non-centered parametrization has issues in the highest values of Tau but in the rest it seems to be more stable.