Assignment 8

LOO-CV model comparison

anonymous

1 General information

This is the template for assignment 8. You can download 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 A hierarchical model for chicken weight time series

2.1 Exploratory data analysis

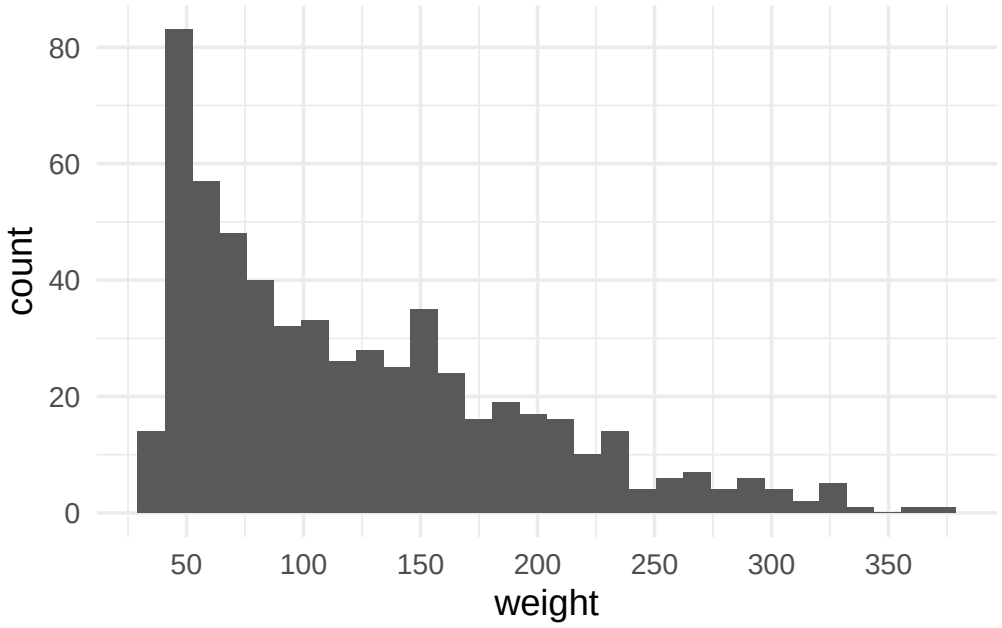
2.1.1 (a)

```
head(ChickWeight,10)
```

```
Grouped Data: weight ~ Time | Chick
   weight Time Chick Diet
1      42    0     1    1
2      51    2     1    1
3      59    4     1    1
4      64    6     1    1
5      76    8     1    1
6      93   10     1    1
7     106   12     1    1
8     125   14     1    1
9     149   16     1    1
10    171   18     1    1
```

```
ggplot(data=ChickWeight) +
  geom_histogram(aes(x=weight)) +
  scale_x_continuous(breaks=c(0,50,100,150,200,250,300,350))
```

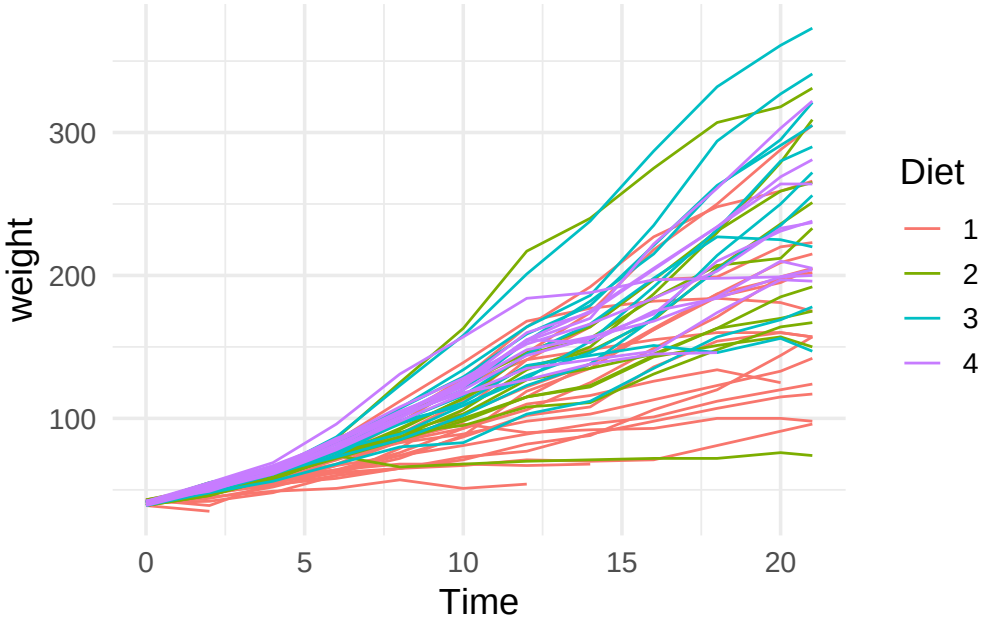
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



The weights go from 0 to 375 grams approximately. We can see that the mode is around 50 grams. Then it decreases pretty much monotonically up to 375 grams. Below 50 we have a small amount too.

2.1.2 (b)

```
ggplot(data= ChickWeight) +
  geom_line(aes(x=Time, y=weight, group=Chick, color=Diet))
```



The chickens seem to have similar weight until day 5 approximately, then we start to see some divergences. At day 20 chicken weight is between 75 and 375 depending on the chicken and diet.

Apparently Diet 1 is associated with lower weights and Diets 3 and 4 have consistently high values.
Also we can see some chicken don't make it to day 20.

2.2 Linear regression

2.2.1 (c)

In `brms`, a regression can be specified as below, see also below (#m) or the last template. Fill in the appropriate variables, data, and likelihood family. Specify the priors, then run the model (by removing `#| eval: false` below).

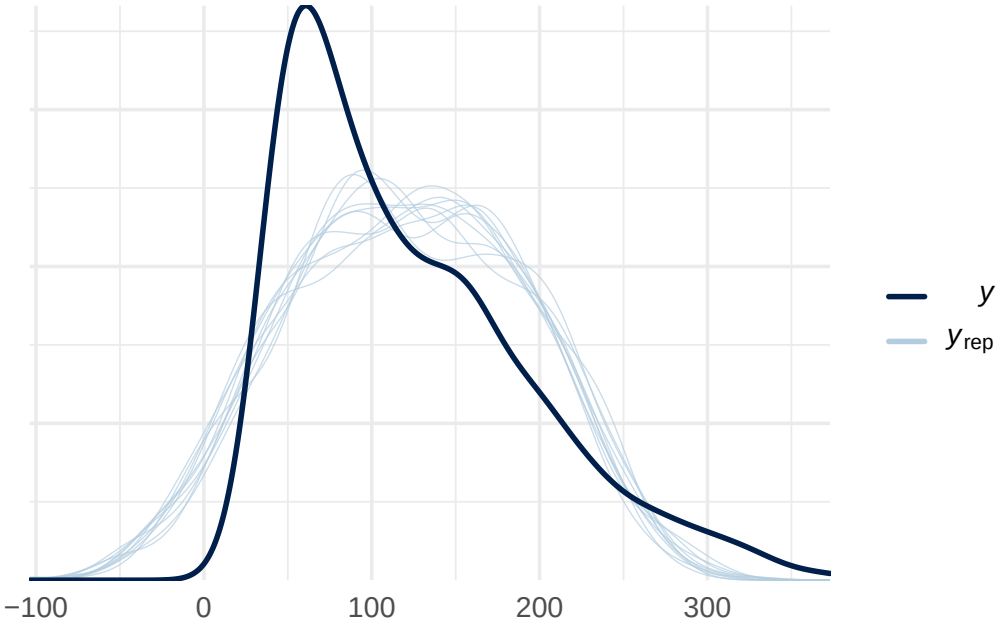
```
priors <- c(
  brms::prior(normal(0, 50), coef = "Time"),
  brms::prior(normal(0, 100 ), coef = "Diet2"),
  brms::prior(normal(0, 100), coef = "Diet3"),
  brms::prior(normal(0, 100), coef = "Diet4")
)

f1 <- brms::brm(
  # This specifies the formula
  weight ~ 1 + Time + Diet,
  # This specifies the dataset
  data = ChickWeight,
  # This specifies the likelihood / the observation family
  family = gaussian(),
  # This passes the priors specified above to brms
  prior = priors,
  # This causes brms to cache the results
  file = "f1"
)
```

2.2.2 (d)

```
brms::pp_check(f1)
```

```
Using 10 posterior draws for ppc type 'dens_overlay' by default.
```

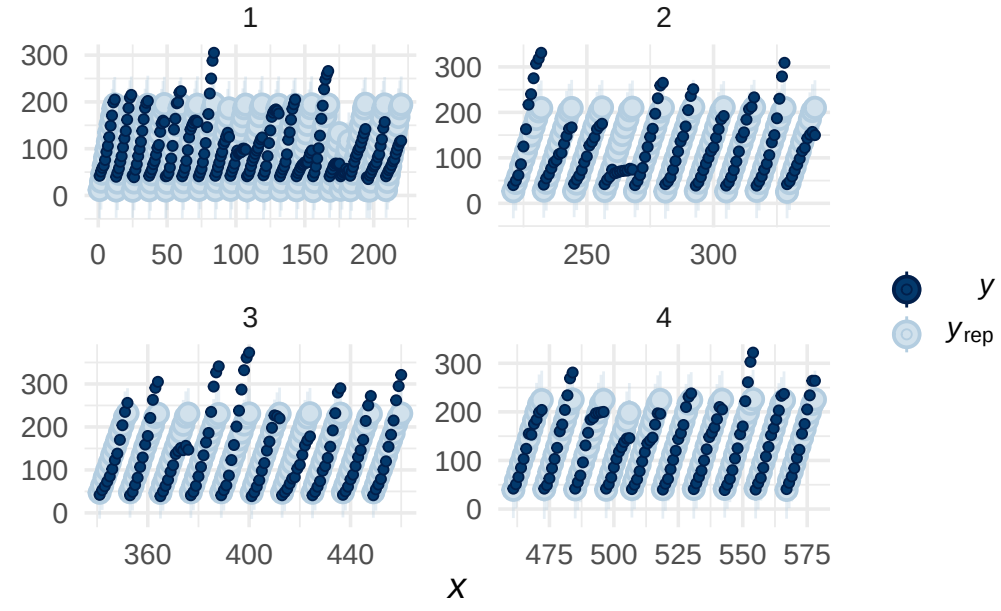


We can see that the predicted data is not similar to the real data.
1) Real data has only positive values, peaks around 50 and then decreases up to 375 grams. Predicted data can take negative values, does not have a clear peak but a plain from 75 to 200 grams and decreases faster and with a smaller tail 2) On top of said above, the predicted data doesn't have any region with as much values as the real data has around 50 grams. The model clearly misses something about the data generation process.

2.2.3 (e)

```
brms::pp_check(f1, type='intervals_grouped', group='Diet')
```

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



We can see that each diet presents similar ranges for all chickens when we know that's not true. For ex. Diet 1 has many chickens where the predicted values go way above the real data. Diet 3 has many where real data goes above the predicted values.
Maybe we can calculate a different slope for Time for each Diet if we keep the pooled format.

2.3 Log-normal linear regression

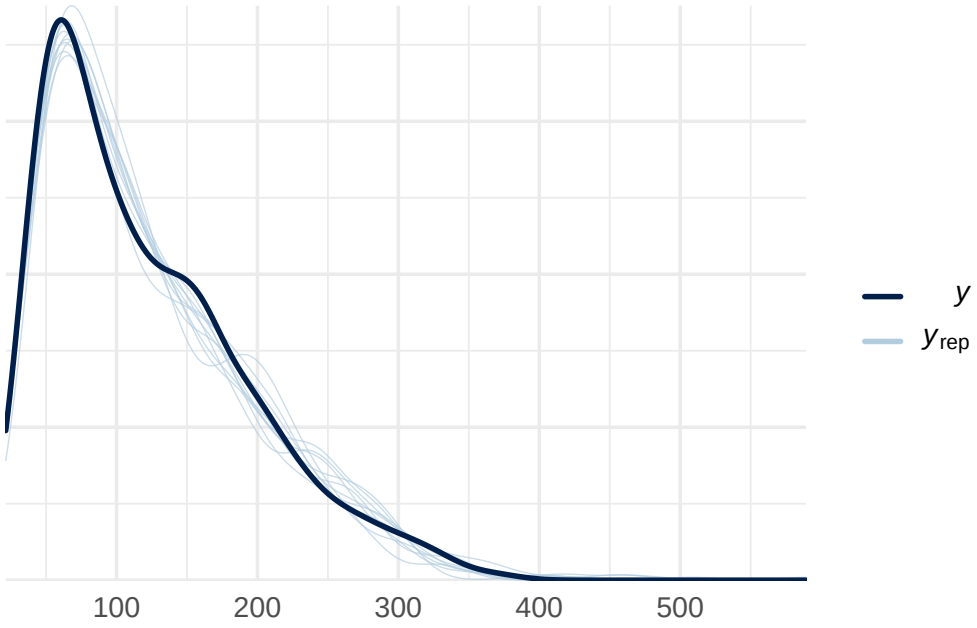
2.3.1 (f)

```
log_priors <- c(
  brms::prior(normal(0, log(3)), coef = "Time"),
  brms::prior(normal(0, log(5)), coef = "Diet2"),
  brms::prior(normal(0, log(5)), coef = "Diet3"),
  brms::prior(normal(0, log(5)), coef = "Diet4")
)
```

```
f2 <- brms::brm(
  # This specifies the formula
  weight ~ 1 + Time + Diet,
  # This specifies the dataset
  data = ChickWeight,
  # This specifies the likelihood / the observation family
  family = lognormal(),
  # This passes the priors specified above to brms
  prior = log_priors,
  # This causes brms to cache the results
  file = "f2"
)
```

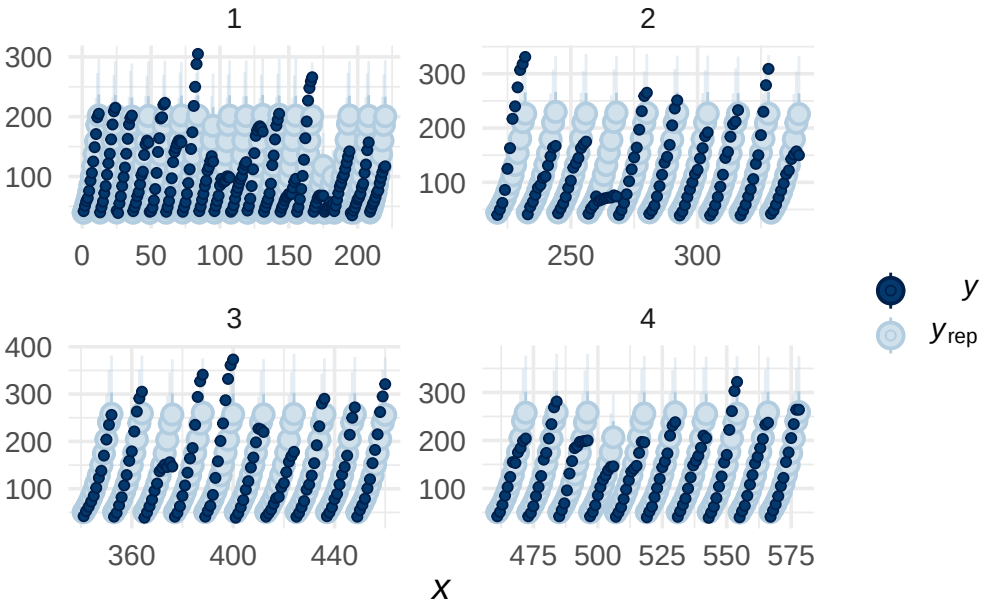
```
brms::pp_check(f2)
```

```
Using 10 posterior draws for ppc type 'dens_overlay' by default.
```




```
brms::pp_check(f2, type='intervals_grouped', group='Diet')
```

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



At the aggregated level we can see a much better fit in terms of predicted vs real obseravations, being the predicted ones following much more closer the real distribution and peaking at 50 grams.
However if we look at the chicken level by diet we can still see severe differences between real and predicted values over time.

2.4 Hierarchical log-normal linear regression

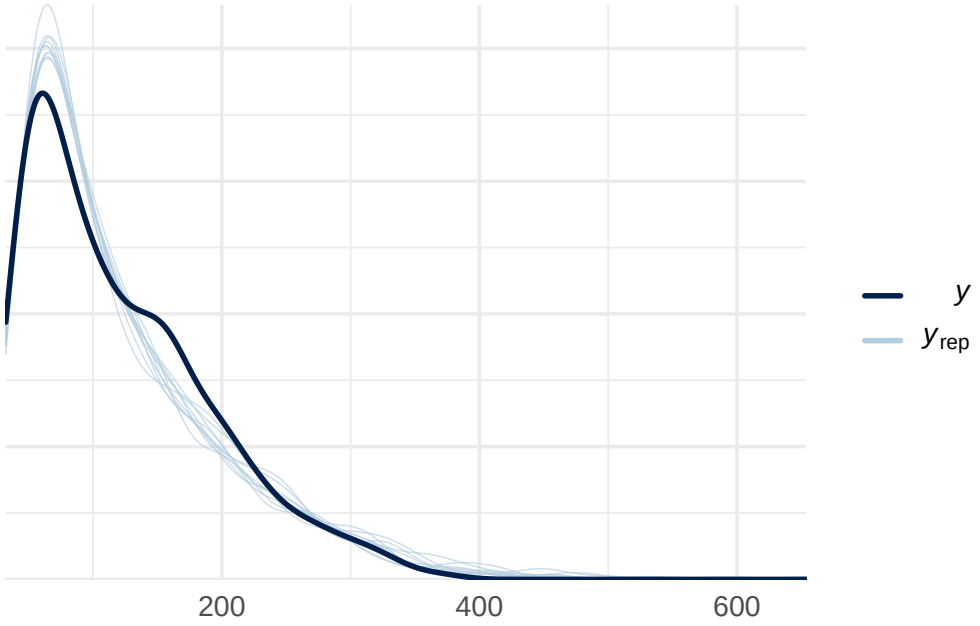
2.4.1 (g)

```
log_priors_hierarchical <- c(
  # prior(normal(0, log(3)), coef = "Time", group="Chick", class="sd"),
  brms::prior(normal(0, log(5)), coef = "Diet2"),
  brms::prior(normal(0, log(5)), coef = "Diet3"),
  brms::prior(normal(0, log(5)), coef = "Diet4")
)
```

```
f3 <- brms::brm(
  # This specifies the formula
  weight ~ 1 + (Time|Chick) + Diet,
  # This specifies the dataset
  data = ChickWeight,
  # This specifies the likelihood / the observation family
  family = lognormal(),
  # This passes the priors specified above to brms
  prior = log_priors_hierarchical,
  # This causes brms to cache the results
  file = "f3"
)
```

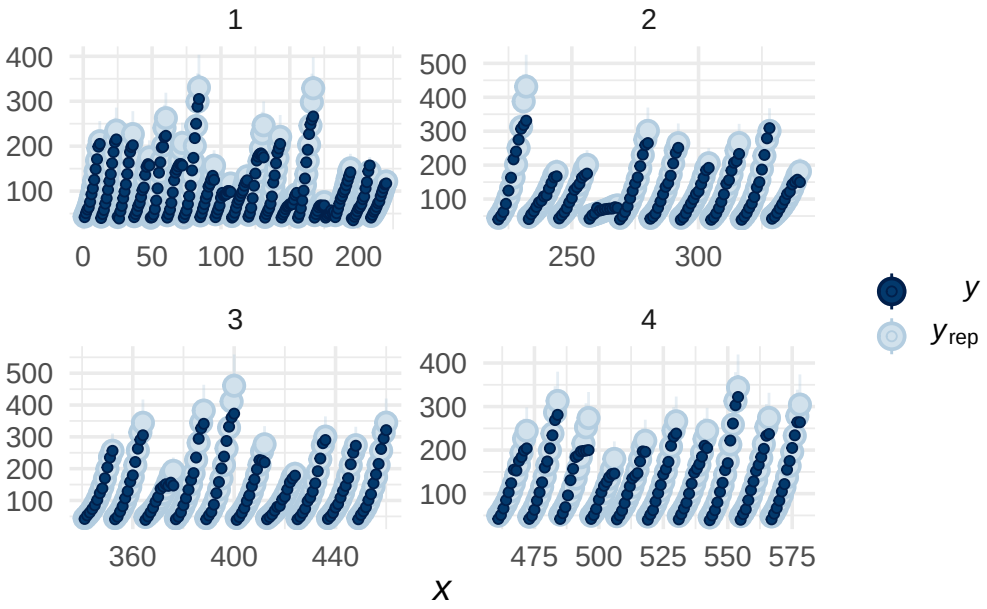
```
brms::pp_check(f3)
```

```
Using 10 posterior draws for ppc type 'dens_overlay' by default.
```




```
brms::pp_check(f3, type='intervals_grouped', group='Diet')
```

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



The first chart seems less accurate, the real line does not go through the predicted ones, we can see clear differences. However, at the chicken level in the second plot we can see a much better fit not having any real observation outside the range of the predictions for each chicken.
I'm honestly a bit confused about the first plot, I thought it would be still accurate as the previous one, specially if at the chicken level we have better results.

2.4.2 (h)

In this run, 5% of the transitions ended up in divergence.

2.5 Model comparison using the ELPD

2.5.1 (i)

```
loo(f1)
```

```
Computed from 4000 by 578 log-likelihood matrix

         Estimate   SE
elpd_loo  -2895.6 25.8
p_loo         7.4  0.9
looic      5791.2 51.7
------
Monte Carlo SE of elpd_loo is 0.0.

All Pareto k estimates are good (k < 0.5).
See help('pareto-k-diagnostic') for details.
```

```
loo(f2)
```

```
Computed from 4000 by 578 log-likelihood matrix

         Estimate   SE
elpd_loo  -2649.3 30.8
p_loo         7.0  1.1
looic      5298.7 61.7
------
Monte Carlo SE of elpd_loo is 0.0.

All Pareto k estimates are good (k < 0.5).
See help('pareto-k-diagnostic') for details.
```

```
loo(f3)
```

```
Computed from 4000 by 578 log-likelihood matrix

         Estimate   SE
elpd_loo  -2254.4 26.9
p_loo        81.5  6.9
looic      4508.7 53.7
------
Monte Carlo SE of elpd_loo is NA.

Pareto k diagnostic values:
                         Count Pct.    Min. n_eff
(-Inf, 0.5]   (good)     563   97.4%   92
 (0.5, 0.7]   (ok)        11    1.9%   19
   (0.7, 1]   (bad)        4    0.7%   33
   (1, Inf)   (very bad)   0    0.0%   <NA>
See help('pareto-k-diagnostic') for details.
```

```
f1 = add_criterion(f1, "loo")
f2 = add_criterion(f2, "loo")
f3 = add_criterion(f3, "loo")
```

2.5.1.1 Loo comparison

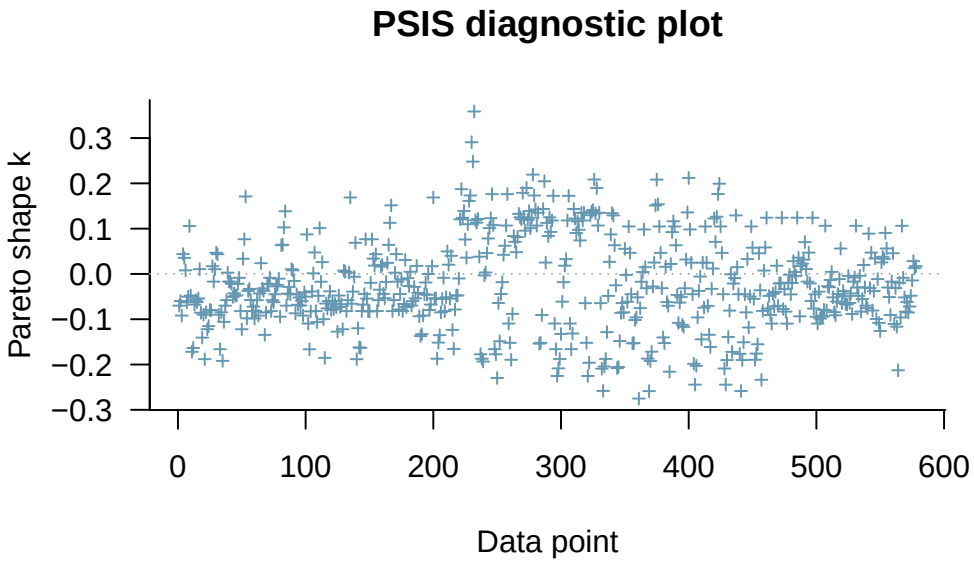
```
loo_compare(f1, f2, f3, criterion="loo")
```

```
   elpd_diff se_diff
f3    0.0       0.0
f2 -395.0      30.6
f1 -641.2      31.7
```

The hierarchical model has the best predictive performance and the standard deviations of the differences are really small compared to the actual difference so they don't influence the decision.

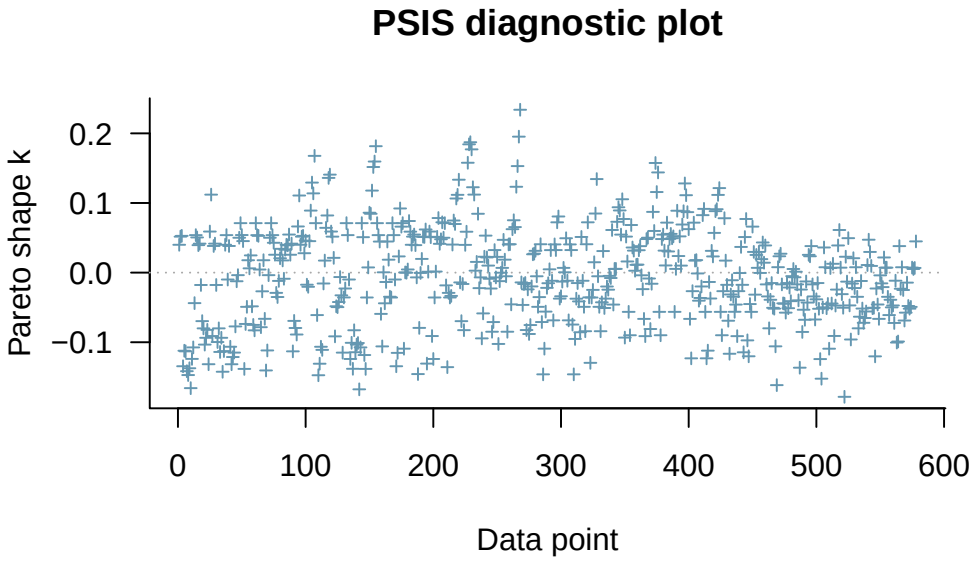
2.5.2 (j)

```
plot(loo(f1), label_points = TRUE)
```



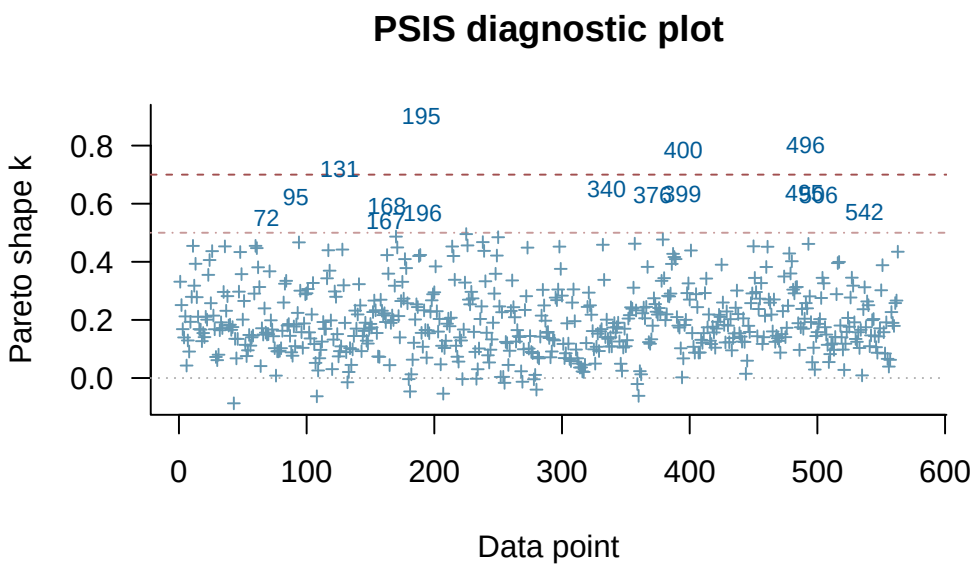

```
# Useful functions: plot(loo(...), label_points = TRUE)
```

```
plot(loo(f2), label_points = TRUE)
```




```
# Useful functions: plot(loo(...), label_points = TRUE)
```

```
plot(loo(f3), label_points = TRUE)
```




```
# Useful functions: plot(loo(...), label_points = TRUE)
```

The hierarchical model has around 2.5% observations with K statistic higher than 0.5 while the pooled models have none. Looking at the chart these higher k statistic observations seem to be just a few distributed across the different diets. Quickly looking they seem to be the more extreme values (highest and lowest weights)

2.5.3 (k)

```
high_chicken_rows = c(131,195,400,496)
ChickWeight[high_chicken_rows,]
```

```
Grouped Data: weight ~ Time | Chick
    weight Time Chick Diet
131    175   21    11    1
195     39    0    18    1
400    373   21    35    3
496    200   21    43    4
```

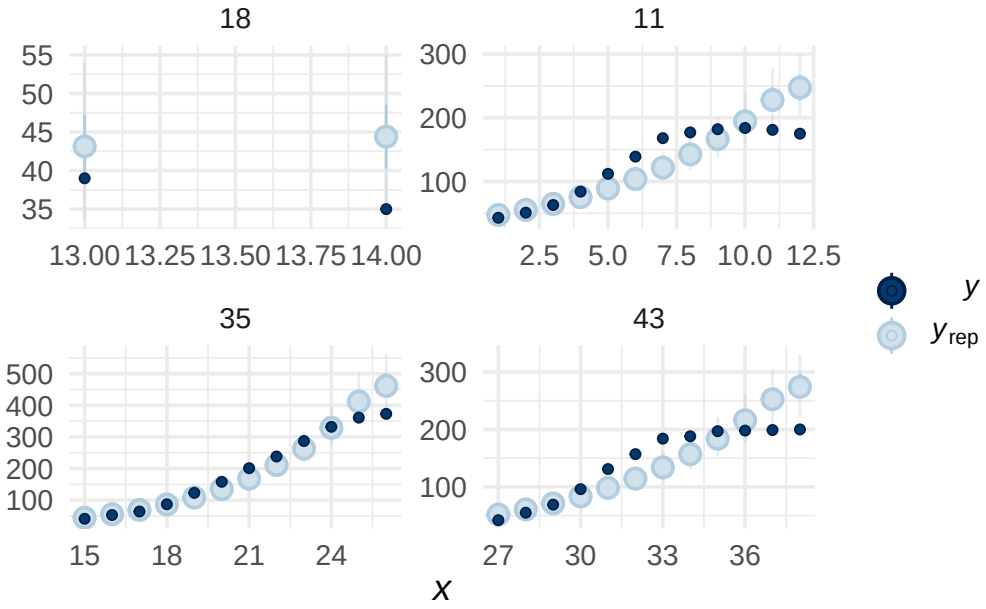
```
k_values = loo(f3)$diagnostics$pareto_k
low_chicken_rows = order(k_values)[0:4]
ChickWeight[low_chicken_rows,]
```

```
Grouped Data: weight ~ Time | Chick
    weight Time Chick Diet
43     102   12     4    1
110     52    4    10    1
368     77    6    33    3
214     73   10    20    1
```

2.5.4 Chickens with highest K

```
# HIGH K
chicken_idxs = c(11,18,35,43)
# Create this plot for your hierarchical model for selected chickens
brms::pp_check(
  f3, type = "intervals_grouped", group = "Chick",
  newdata=ChickWeight %>% filter(Chick %in% chicken_idxs)
)
```

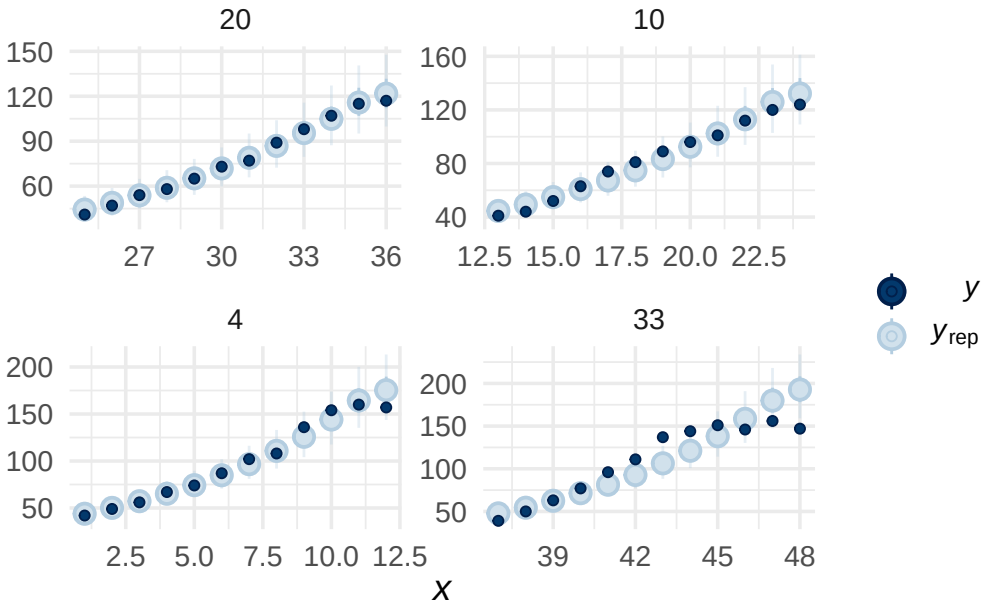
```
Using all posterior draws for ppc type 'intervals_grouped' by default.
```



2.5.5 Chickens with lowest K

```
# LOW  K
chicken_idxs = c(4,10,33,20)
# Create this plot for your hierarchical model for selected chickens
brms::pp_check(
  f3, type = "intervals_grouped", group = "Chick",
  newdata=ChickWeight %>% filter(Chick %in% chicken_idxs)
)
```

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



We can see that the chickens with highest K are the ones with some extreme values (chicken 35 has the highest weight at time 21) and where the weight clearly has some kind of sigmoid shape, going up and then being more stable breaking the linear relationship.
Chicken 18 in the other hand only has two observations..

Among the lowest K, chicken 4, 10 and 20 follow pretty well the linear relationship and the predicted Y are quite similar to the really y in many times.
Chicken 33 differs but in the first observations is quite accurate and that's one of the lowest K observations.

2.6 Model comparison using the RMSE

2.6.1 (I)

```
rmse_f1 = rmse(f1)
rmse_f2 = rmse(f2)
rmse_f3 = rmse(f3)
```

```
[1] "RMSE on the train data"
```

```
[1] 35.84708
```

```
[1] 34.71348
```

```
[1] 15.8592
```

```
rmse_loo_f1 = rmse(f1, use_loo = TRUE)
```

```
Running PSIS to compute weights
```

```
  rmse_loo_f2 = rmse(f2, use_loo = TRUE)
```

```
Running PSIS to compute weights
```

```
  rmse_loo_f3 = rmse(f3, use_loo = TRUE)
```

```
Running PSIS to compute weights
```

```
[1] "RMSE usgin LOO"
```

```
[1] 36.15745
```

```
[1] 35.10708
```

```
[1] 20.23473
```

We can see that systematically the have a greater RMSE in LOO than in the full training data. The reason is that for LOO we don't use the evaluated observation for training so the posterior doesn't know about that particular value. In the other hand, the full RMSE has seen all the data before calculating the RMSE so it's not a good estimation of RMSE on future observations.