My title*

My subtitle if needed

Hannah Yu

April 9, 2024

First sentence. Second sentence. Third sentence. Fourth sentence.

```
ces2020 <-  read_parquet(here::here("data/analysis_data/cleaned_ces2020.parquet"))
```

1 Introduction

You can and should cross-reference sections and sub-sections. We use R Core Team (2023) and Wickham et al. (2019).

The remainder of this paper is structured as follows. Section 2….

2 Data

Some of our data is of penguins (**?@fig-bills**), from Horst, Hill, and Gorman (2020).

Talk more about it.

And also planes (**?@fig-planes**). (You can change the height and width, but don't worry about doing that until you have finished every other aspect of the paper - Quarto will try to make it look nice and the defaults usually work well once you have enough text.)

*Code and data are available at: LINK.

3 Model

The goal of our modelling strategy is twofold. Firstly,...

Here we briefly describe the Bayesian analysis model used to investigate... Background details and diagnostics are included in Appendix B.

3.1 Model set-up

Define y_i as the number of seconds that the plane remained aloft. Then β_i is the wing width and γ_i is the wing length, both measured in millimeters.

$$y_i|\mu_i, \sigma \sim \text{Normal}(\mu_i, \sigma) \tag{1}$$
$$\mu_i = \alpha + \beta_i + \gamma_i \tag{2}$$
$$\alpha \sim \text{Normal}(0, 2.5) \tag{3}$$
$$\beta \sim \text{Normal}(0, 2.5) \tag{4}$$
$$\gamma \sim \text{Normal}(0, 2.5) \tag{5}$$
$$\sigma \sim \text{Exponential}(1) \tag{6}$$

We run the model in R (R Core Team 2023) using the `rstanarm` package of Goodrich et al. (2022). We use the default priors from `rstanarm`.

3.1.1 Model justification

We expect a positive relationship between the size of the wings and time spent aloft. In particular...

We can use maths by including latex between dollar signs, for instance θ.

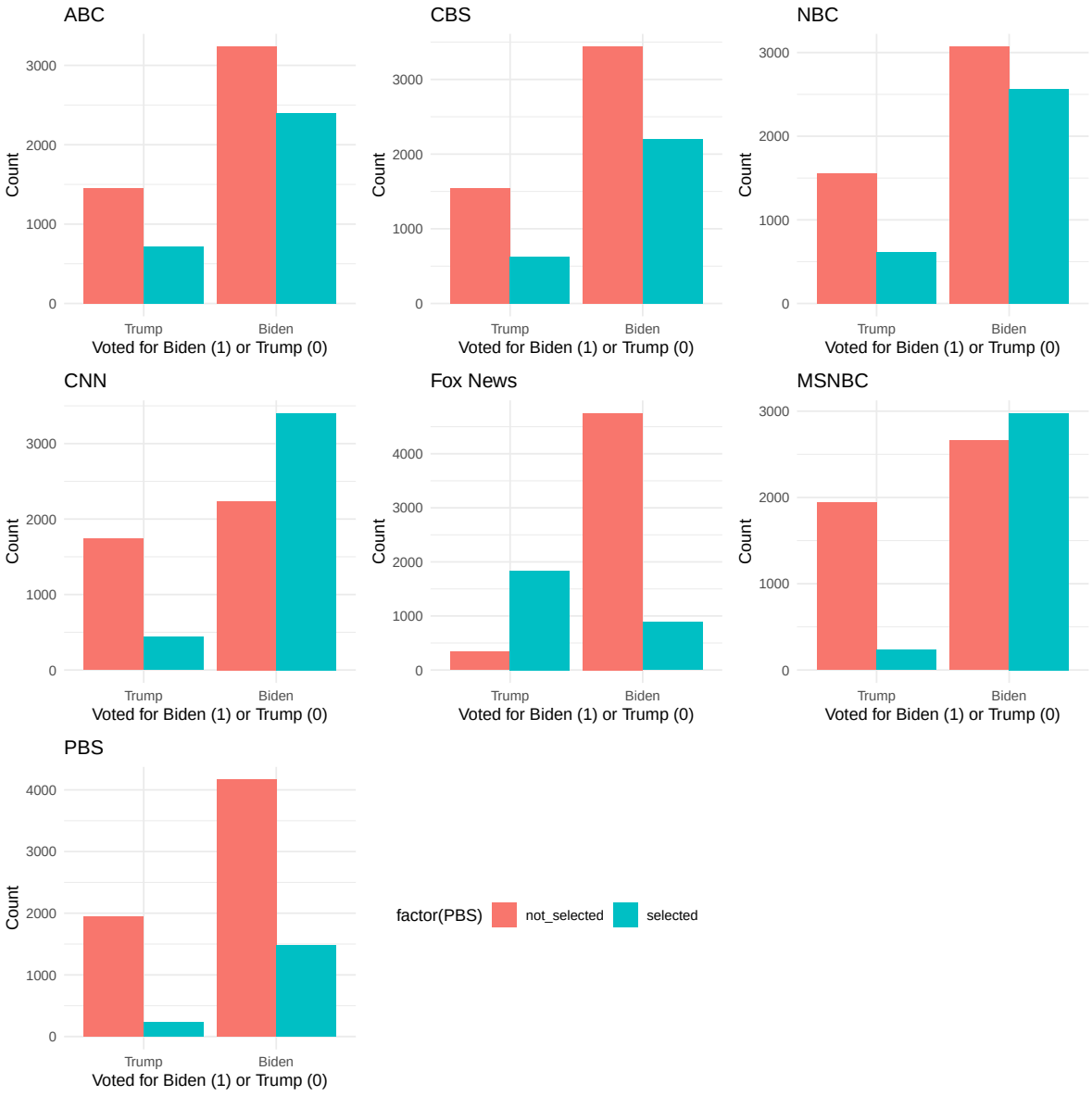
4 Results

```
# Create individual plots for each news channel
plot_ABC <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(ABC))) +
  geom_bar(position = "dodge") +
  labs(title = "ABC", x = "Voted for Biden (1) or Trump (0)", y = "Count") +
  theme_minimal() +
 # facet_wrap(~ ., ncol = 4) +
  guides(fill = FALSE)
```

```
Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
of ggplot2 3.3.4.
```

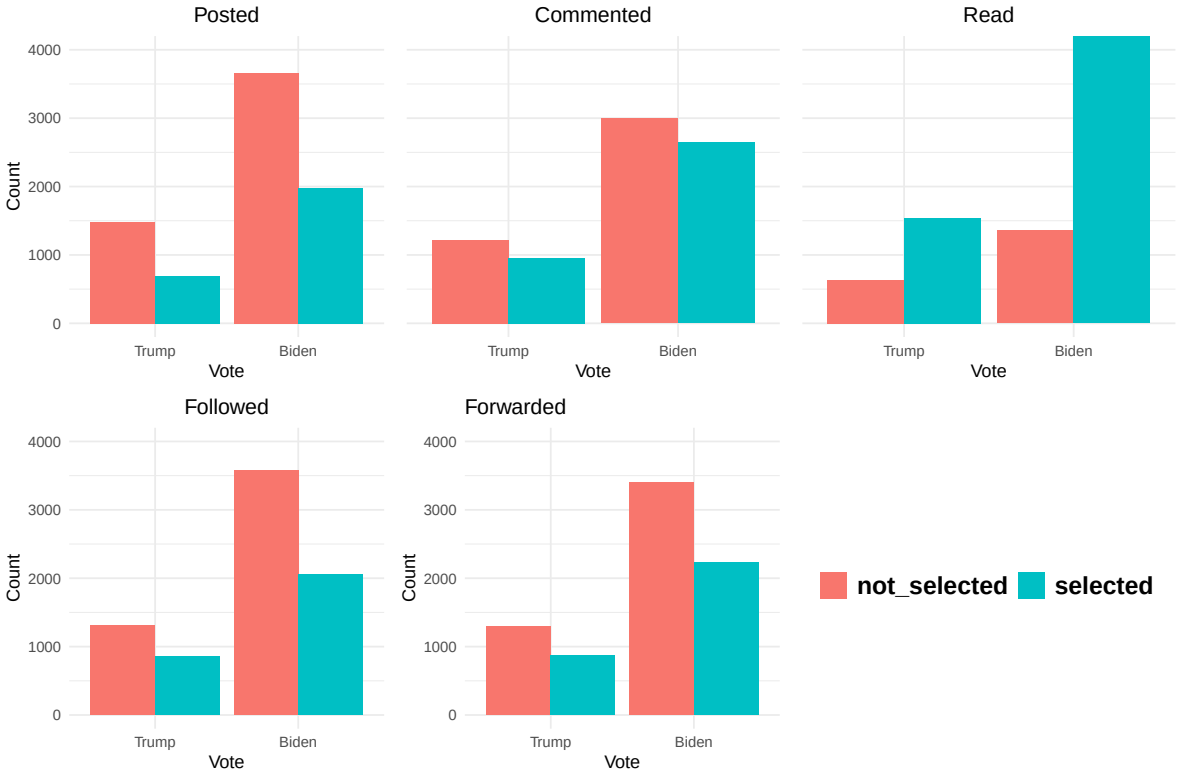
```r
plot_CBS <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(CBS))) +
  geom_bar(position = "dodge") +
  labs(title = "CBS", x = "Voted for Biden (1) or Trump (0)", y = "Count") +
  theme_minimal() +
 # facet_wrap(~ ., ncol = 4) +
  guides(fill = FALSE)

plot_NBC <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(NBC))) +
  geom_bar(position = "dodge") +
  labs(title = "NBC", x = "Voted for Biden (1) or Trump (0)", y = "Count") +
  theme_minimal() +
 # facet_wrap(~ ., ncol = 4) +
  guides(fill = FALSE)

plot_CNN <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(CNN))) +
  geom_bar(position = "dodge") +
  labs(title = "CNN", x = "Voted for Biden (1) or Trump (0)", y = "Count") +
  theme_minimal() +
 # facet_wrap(~ ., ncol = 4) +
  guides(fill = FALSE)

plot_Fox_News <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(Fox_News))) +
  geom_bar(position = "dodge") +
  labs(title = "Fox News", x = "Voted for Biden (1) or Trump (0)", y = "Count") +
  theme_minimal() +
#  facet_wrap(~ ., ncol = 4) +
  guides(fill = FALSE)

plot_MSNBC <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(MSNBC))) +
  geom_bar(position = "dodge") +
  labs(title = "MSNBC", x = "Voted for Biden (1) or Trump (0)", y = "Count") +
  theme_minimal() +
#  facet_wrap(~ ., ncol = 4) +
  guides(fill = FALSE)

plot_PBS <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(PBS))) +
  geom_bar(position = "dodge") +
  labs(title = "PBS", x = "Voted for Biden (1) or Trump (0)", y = "Count") +
```

```
  theme_minimal()

legend <- g_legend(plot_PBS + theme(legend.position='bottom'))

grid.arrange(plot_ABC, plot_CBS, plot_NBC, plot_CNN, plot_Fox_News, plot_MSNBC, plot_PBS +
```



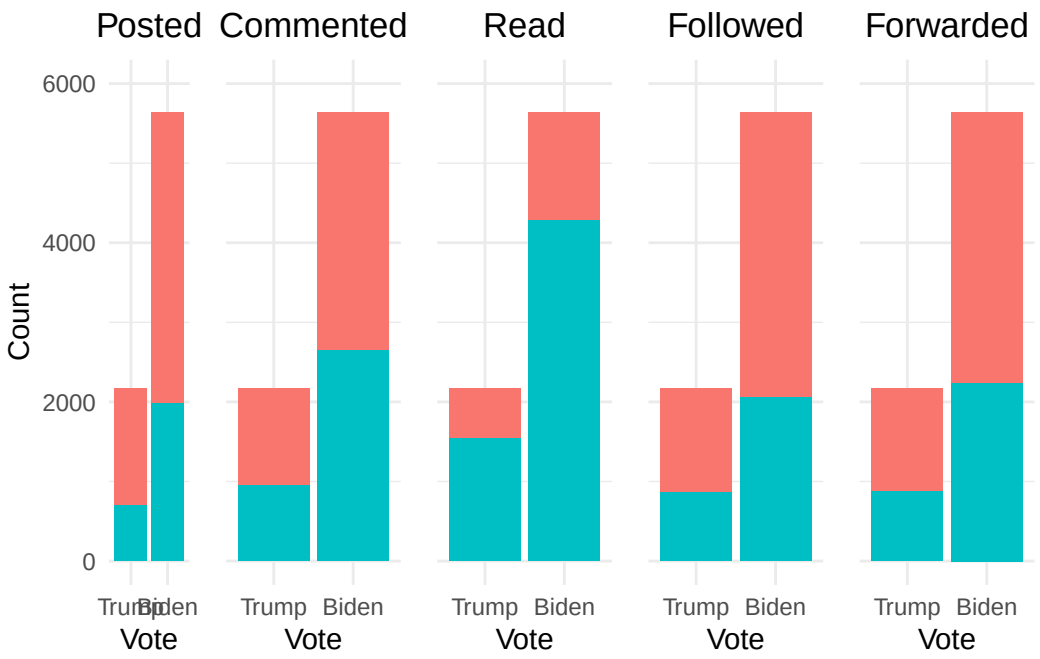
```r
# Create individual plots for each news channel
plot_post <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(post))) +
  geom_bar(position = "dodge") +
  labs(title = "Posted", x = "Vote", y = "Count") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5)) +
  coord_cartesian(ylim = c(0, 4000)) +
  guides(fill = FALSE)

plot_comment <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(comment))) +
  geom_bar(position = "dodge") +
  labs(title = "Commented", x = "Vote", y = "Count") +
  theme_minimal() +
  coord_cartesian(ylim = c(0, 4000)) +
  theme(axis.title.y = element_blank(), axis.text.y = element_blank(), axis.ticks.y = elem
 # facet_wrap(~ ., ncol = 4) +
  guides(fill = FALSE)

plot_read <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(read))) +
  geom_bar(position = "dodge") +
  labs(title = "Read", x = "Vote", y = "Count") +
  theme_minimal() +
  coord_cartesian(ylim = c(0, 4000)) +
  theme(axis.title.y = element_blank(), axis.text.y = element_blank(), axis.ticks.y = elem
 # facet_wrap(~ ., ncol = 4) +
  guides(fill = FALSE)

plot_follow <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(follow))) +
  geom_bar(position = "dodge") +
  labs(title = "Followed", x = "Vote", y = "Count") +
  theme_minimal() +
  coord_cartesian(ylim = c(0, 4000)) +
  theme(plot.title = element_text(hjust = 0.5)) +
 # facet_wrap(~ ., ncol = 4) +
  guides(fill = FALSE)

plot_forward <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(forward))) +
  geom_bar(position = "dodge") +
  labs(title = "Forwarded", x = "Vote", y = "Count") +
  coord_cartesian(ylim = c(0, 4000)) +
  theme(axis.title.y = element_blank(), axis.text.y = element_blank(), axis.ticks.y = elem
```

```
  theme_minimal()
 # facet_wrap(~ ., ncol = 4) +


legend <- g_legend(plot_forward + theme(legend.position='bottom',
                                        legend.text = element_text(size = 15, face="bold")
                                        legend.title = element_blank()))

grid.arrange(plot_post, plot_comment, plot_read, plot_follow, plot_forward + theme(legend.
```




```
# Create individual plots for each news channel
plot_post <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(post, levels = c("no
  geom_bar(position = "stack", stat = "count") +
  labs(title = "Posted", x = "Vote", y = "Count") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5)) +
  coord_cartesian(ylim = c(0, 6000)) +
  guides(fill = FALSE)
```

```r
plot_comment <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(comment, levels =
  geom_bar(position = "stack", stat = "count") +
  labs(title = "Commented", x = "Vote", y = "Count") +
  theme_minimal() +
  coord_cartesian(ylim = c(0, 6000)) +
  theme(axis.title.y = element_blank(), axis.text.y = element_blank(), axis.ticks.y = elem
  guides(fill = FALSE)

plot_read <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(read, levels = c("no
  geom_bar(position = "stack", stat = "count") +
  labs(title = "Read", x = "Vote", y = "Count") +
  theme_minimal() +
  coord_cartesian(ylim = c(0, 6000)) +
  theme(axis.title.y = element_blank(), axis.text.y = element_blank(), axis.ticks.y = elem
  guides(fill = FALSE)

plot_follow <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(follow, levels = c
  geom_bar(position = "stack", stat = "count") +
  labs(title = "Followed", x = "Vote", y = "Count") +
  theme_minimal() +
  coord_cartesian(ylim = c(0, 6000)) +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme(axis.title.y = element_blank(), axis.text.y = element_blank(), axis.ticks.y = elem
  guides(fill = FALSE)

plot_forward <- ggplot(ces2020, aes(x = factor(voted_for), fill = factor(forward, levels =
  geom_bar(position = "stack", stat = "count") +
  labs(title = "Forwarded", x = "Vote", y = "Count") +
  coord_cartesian(ylim = c(0, 6000)) +
  theme_minimal() +
  theme(axis.title.y = element_blank(), axis.text.y = element_blank(), axis.ticks.y = elem
 # facet_wrap(~ ., ncol = 4) +


#legend <- g_legend(plot_forward + theme(legend.position='bottom',
                                         #legend.text = element_text(size = 15, face="bold"
                                        # legend.title = element_blank())

grid.arrange(plot_post, plot_comment, plot_read, plot_follow, plot_forward + theme(legend.
```



Our results are summarized in **?@tbl-modelresults**.

```
modelsummary(
  list(
    "Support Biden" = political_preferences
  ),
  statistic = "mad"
  )
```

```
Warning:
`modelsummary` uses the `performance` package to extract goodness-of-fit
statistics from models of this class. You can specify the statistics you wish
to compute by supplying a `metrics` argument to `modelsummary`, which will then
push it forward to `performance`. Acceptable values are: "all", "common",
"none", or a character vector of metrics names. For example: `modelsummary(mod,
metrics = c("RMSE", "R2")` Note that some metrics are computationally
expensive. See `?performance::performance` for details.
 This warning appears once per session.
```

```
# Load necessary libraries
library(car)
```

	Support Biden
(Intercept)	1.416
	(0.258)
postselected	0.012
	(0.197)
commentselected	−0.107
	(0.187)
readselected	0.523
	(0.177)
followselected	0.044
	(0.158)
forwardselected	−0.227
	(0.182)
ABCselected	0.105
	(0.154)
CBSselected	0.027
	(0.166)
NBCselected	0.167
	(0.157)
CNNselected	1.771
	(0.164)
Fox_Newsselected	−3.770
	(0.162)
MSNBCselected	1.914
	(0.183)
PBSselected	0.362
	(0.190)
Otherselected	−1.246
	(0.216)
TV_typeNational Newscast	−0.225
	(0.175)
Newspaper_typeOnline	0.168
	(0.186)
Newspaper_typePrint	−0.602
	(0.227)
Num.Obs.	2500
R2	0.594
Log.Lik.	−669.964
ELPD	−686.9
ELPD s.e.	28.9
LOOIC	1373.9
LOOIC s.e.	57.8
WAIC	1373.8
RMSE	0.28

```
Loading required package: carData


Attaching package: 'car'

The following object is masked from 'package:rstanarm':

    logit

The following object is masked from 'package:dplyr':

    recode

The following object is masked from 'package:purrr':

    some
```

```r
# Check for multicollinearity using VIF
vif(political_preferences)
```

```
                  GVIF Df GVIF^(1/(2*Df))
post          1.648485  1        1.283933
comment       1.632211  1        1.277580
read          1.269848  1        1.126875
follow        1.253147  1        1.119440
forward       1.516867  1        1.231611
ABC           1.175825  1        1.084355
CBS           1.208830  1        1.099468
NBC           1.164232  1        1.078996
CNN           1.282863  1        1.132635
Fox_News      1.381391  1        1.175326
MSNBC         1.116343  1        1.056571
PBS           1.087766  1        1.042960
Other         1.069697  1        1.034261
TV_type       1.183873  1        1.088059
Newspaper_type 1.153136  2        1.036263
```

```r
# View the model summary
summary(political_preferences)
```

```
Model Info:
 function:     stan_glm
 family:       binomial [logit]
 formula:      voted_for_binary ~ post + comment + read + follow + forward +
       ABC + CBS + NBC + CNN + Fox_News + MSNBC + PBS + Other +
       TV_type + Newspaper_type
 algorithm:    sampling
 sample:       4000 (posterior sample size)
 priors:       see help('prior_summary')
 observations: 2500
 predictors:   17

Estimates:
                            mean   sd   10%   50%   90%
(Intercept)                 1.4    0.3  1.1   1.4   1.7
postselected                0.0    0.2 -0.2   0.0   0.3
commentselected            -0.1    0.2 -0.3  -0.1   0.1
readselected                0.5    0.2  0.3   0.5   0.7
followselected              0.0    0.2 -0.2   0.0   0.3
forwardselected            -0.2    0.2 -0.5  -0.2   0.0
ABCselected                 0.1    0.2 -0.1   0.1   0.3
CBSselected                 0.0    0.2 -0.2   0.0   0.2
NBCselected                 0.2    0.2  0.0   0.2   0.4
CNNselected                 1.8    0.2  1.6   1.8   2.0
Fox_Newsselected           -3.8    0.2 -4.0  -3.8  -3.6
MSNBCselected               1.9    0.2  1.7   1.9   2.1
PBSselected                 0.4    0.2  0.1   0.4   0.6
Otherselected              -1.2    0.2 -1.5  -1.2  -1.0
TV_typeNational Newscast   -0.2    0.2 -0.4  -0.2   0.0
Newspaper_typeOnline        0.2    0.2 -0.1   0.2   0.4
Newspaper_typePrint        -0.6    0.2 -0.9  -0.6  -0.3

Fit Diagnostics:
           mean   sd   10%   50%   90%
mean_PPD 0.7    0.0  0.7   0.7   0.7

The mean_ppd is the sample average posterior predictive distribution of the outcome variable

MCMC diagnostics
                          mcse Rhat n_eff
(Intercept)               0.0  1.0  4953
postselected              0.0  1.0  6355
```

```
commentselected          0.0  1.0  5827
readselected             0.0  1.0  6973
followselected           0.0  1.0  6592
forwardselected          0.0  1.0  6424
ABCselected              0.0  1.0  7021
CBSselected              0.0  1.0  6894
NBCselected              0.0  1.0  7991
CNNselected              0.0  1.0  5949
Fox_Newsselected         0.0  1.0  4575
MSNBCselected            0.0  1.0  5632
PBSselected              0.0  1.0  6951
Otherselected            0.0  1.0  6510
TV_typeNational Newscast 0.0  1.0  6150
Newspaper_typeOnline     0.0  1.0  5665
Newspaper_typePrint      0.0  1.0  5021
mean_PPD                 0.0  1.0  4398
log-posterior            0.1  1.0  1792

For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective
```

5 Discussion

5.1 First discussion point

If my paper were 10 pages, then should be be at least 2.5 pages. The discussion is a chance to show off what you know and what you learnt from all this.

5.2 Second discussion point

5.3 Third discussion point

5.4 Weaknesses and next steps

Weaknesses and next steps should also be included.

Appendix

A Additional data details

B Model details

B.1 Posterior predictive check

In **?@fig-ppcheckandposteriorvsprior-1** we implement a posterior predictive check. This shows…

In **?@fig-ppcheckandposteriorvsprior-2** we compare the posterior with the prior. This shows…

Examining how the model fits, and is affected by, the data

Figure 1: **?(caption)**

B.2 Diagnostics

?@fig-stanareyouokay-1 is a trace plot. It shows… This suggests…

?@fig-stanareyouokay-2 is a Rhat plot. It shows… This suggests…

Checking the convergence of the MCMC algorithm

Figure 2: **?(caption)**

References

Goodrich, Ben, Jonah Gabry, Imad Ali, and Sam Brilleman. 2022. "Rstanarm: Bayesian Applied Regression Modeling via Stan." https://mc-stan.org/rstanarm/.

Horst, Allison Marie, Alison Presmanes Hill, and Kristen B Gorman. 2020. *Palmerpenguins: Palmer Archipelago (Antarctica) Penguin Data*. https://doi.org/10.5281/zenodo.3960218.

R Core Team. 2023. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the tidyverse." *Journal of Open Source Software* 4 (43): 1686. https://doi.org/10.21105/joss.01686.