Assignment 2 for DSSE - Bayesian modeling

Anuinder Sekhon, V01022326
Sri Surya Jitendra Palaparty, V01022218

Submitted to: Neil Ernst

June 9, 2023

THIS ASSIGNMENT INVOLVED demonstrated the concepts learned from the lectures by developing a Bayesian workflow analysis to evaluate the software engineering result derived from conventional means. The data used in this analysis has been partly published before[1]. The idea was to use COCOMO (Constructive Cost Model) model to understand how well the predictors, like project type and size, will estimate the eventual project costs like salaries and effort.

THE DATASET USED is taken from the link provided in the problem statement of the assignment[2]. It describes the project characteristics of the NASA software projects. The dimensions of the dataset are (93,8), which implies that it has 93 rows and 8 attributes, including the standard COCOMO model discrete attributes in the range from Very_Low to Very_High [3]. To summarize, the attributes are effort required to complete the project (*act_effort*) in person months, record number (*recordNumber*), categories (*cat2*), project complexity (*cplx*), year (*year*), project category (*pcat*), analyst capability (*acap*), programmer capability (*pcap*). The aim is to predict *act_effort* using the predictors *cplx*, *year*, *pcat*, *acap*, *pcap*.

```
       X         recordnumber       cat2               year
 Min.   : 1   Min.   :  1.00   Length:93          Min.   :1971
 1st Qu.:24   1st Qu.: 24.00   Class :character   1st Qu.:1979
 Median :47   Median : 47.00   Mode  :character   Median :1980
 Mean   :47   Mean   : 47.75                      Mean   :1981
 3rd Qu.:70   3rd Qu.: 70.00                      3rd Qu.:1983
 Max.   :93   Max.   :101.00                      Max.   :1987
     cplx               acap               pcap             act_effort
 Length:93          Length:93          Length:93          Min.   :   8.4
 Class :character   Class :character   Class :character   1st Qu.:  70.0
 Mode  :character   Mode  :character   Mode  :character   Median : 252.0
                                                          Mean   : 624.4
                                                          3rd Qu.: 600.0
                                                          Max.   :8211.0
```

```
       X         recordnumber         cat2             year           cplx
 Min.   : 1   Min.   :  1.00   Min.   : 1.000   Min.   :1971   Min.   :1.000
 1st Qu.:24   1st Qu.: 24.00   1st Qu.: 3.000   1st Qu.:1979   1st Qu.:1.000
 Median :47   Median : 47.00   Median : 6.000   Median :1980   Median :1.000
 Mean   :47   Mean   : 47.75   Mean   : 6.032   Mean   :1981   Mean   :2.011
 3rd Qu.:70   3rd Qu.: 70.00   3rd Qu.: 8.000   3rd Qu.:1983   3rd Qu.:3.000
 Max.   :93   Max.   :101.00   Max.   :14.000   Max.   :1987   Max.   :5.000
      acap            pcap          act_effort
 Min.   :1.000   Min.   :1.000   Min.   :   8.4
 1st Qu.:1.000   1st Qu.:1.000   1st Qu.:  70.0
 Median :1.000   Median :2.000   Median : 252.0
 Mean   :1.559   Mean   :1.688   Mean   : 624.4
 3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.: 600.0
 Max.   :3.000   Max.   :3.000   Max.   :8211.0
```

Figure 1. The descriptive statistics of the dataset before (left) and after (right) preprocessing

The R file demonstrating all the work performed on the dataset has been attached to this report. The preprocessing steps performed were:

- The dataset does not contain any null values.
- The conversion of categorical attributes, namely, *cat2*, *cplx*, *acap*, *pcap*, to the numeric values for the purpose of analyzing.
- The conversion of the data type of the *recordNumber*, *year*, *act_effort attributes* from integer to numeric type.
- Various plots were created to get more insight into the data and study the relationships and correlations between the dataset's attributes.

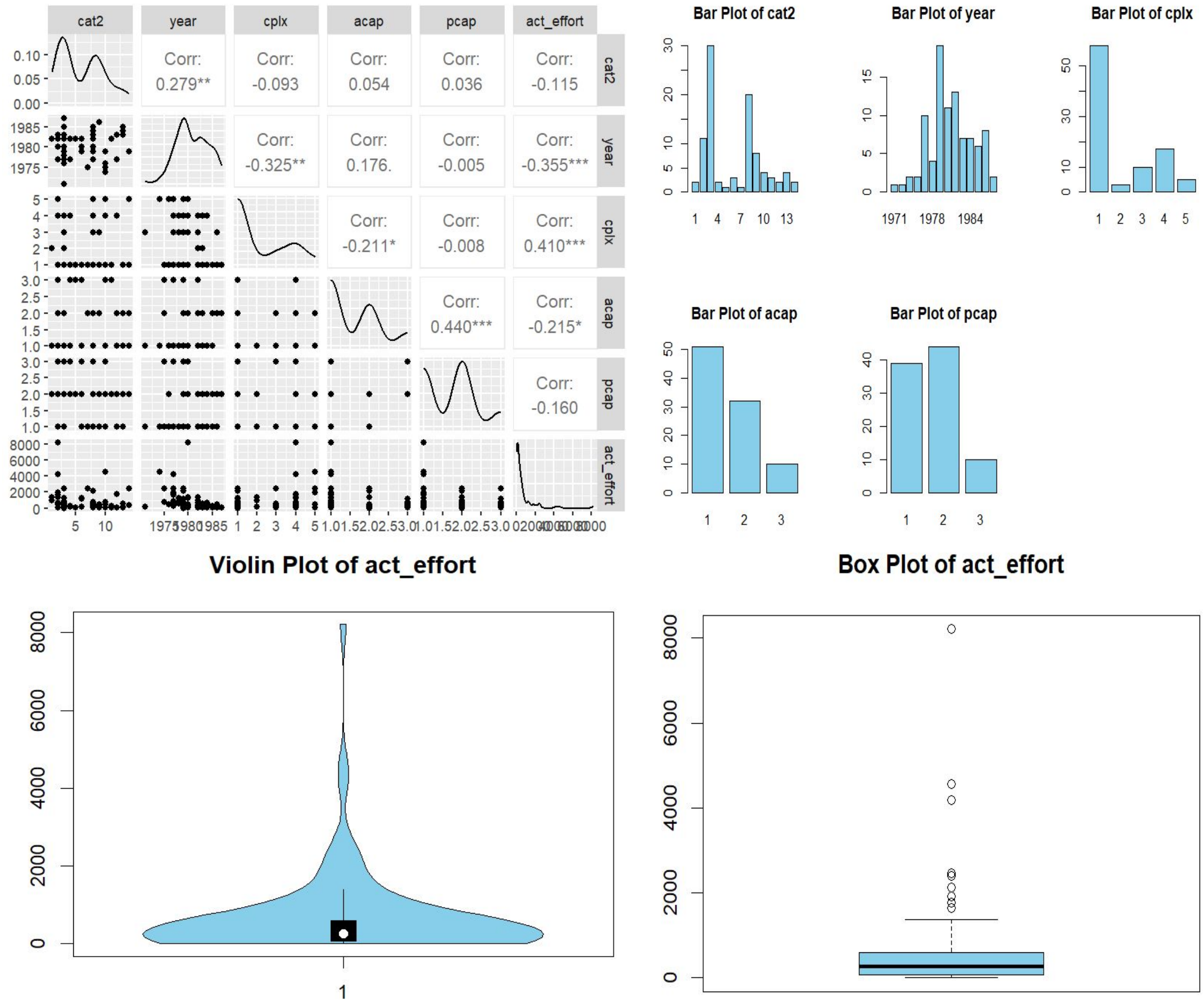


Figure 2. Scatter Plot Matrix, Bar Plots, Violin Plot and Box Plot

Observations: The Violin Plot and the Box Plot of the *act_effort* shows that the data is heavy-tailed. This clearly implies that this distribution of the *act_effort* is in such as way that there are some projects that require a lot of effort, but they cannot be considered outliers as they are in significant numbers,

- No outlier removal was performed, as all the data is useful.
- The process of normalization was not performed as the descriptive analysis shows that the data is overall within a similar scale.

BAYESIAN ANALYSIS was applied to the dataset. Bayesian analysis can be helpful as it allows to integrate the knowledge of the prior information in the estimation process, the uncertainty around the estimated effort schedule, and the cost can be obtained by the posterior distribution, updating and refining the model iteratively as the new data becomes available and supporting the probabilistic project planning.

The likelihood chosen has a significant effect on the results and the complexity of the Bayesian modeling for the estimation process. In our assignment, *ulam* was used to fit different distributions with different likelihoods. As demonstrated in our analysis, the different values of the likelihoods handle the model's sensitivity to the outliers differently; the heavy-tailed likelihood function can cause the posterior distributions, which indicate more uncertainty in the estimates and the value of likelihood function that matches the data distribution leads to better fit. Along with the model was analysis by experimenting with the different values of the prior. The choice of prior can impact the posterior distribution and can be used to introduce the bias toward certain parameter values. Default prior was used - Normal ~ (0,10).

	mean <dbl>	sd <dbl>	5.5% <dbl>	94.5% <dbl>	n_eff <dbl>	Rhat4 <dbl>
intercept	-0.64	0.76	-1.72	0.36	2	11.48
beta1	-0.06	0.01	-0.08	-0.05	2	13.65
beta2	0.06	0.10	0.00	0.24	2	343.33
beta3	0.11	0.57	-0.88	0.44	2	225.87
beta4	-0.43	0.13	-0.65	-0.34	2	9.77
beta5	-0.60	0.54	-1.53	-0.28	2	11.45

	mean <dbl>	sd <dbl>	5.5% <dbl>	94.5% <dbl>	n_eff <dbl>	Rhat4 <dbl>
intercept	4.78	9.47	-10.36	19.69	941	1.02
beta1	-0.03	0.03	-0.07	0.01	9075	1.00
beta2	0.00	0.00	-0.01	0.01	943	1.01
beta3	0.24	0.06	0.14	0.34	8043	1.00
beta4	-0.31	0.17	-0.57	-0.04	6488	1.00
beta5	-0.08	0.16	-0.33	0.17	7000	1.00
log_scale	6.69	0.19	6.40	6.99	8702	1.00

	mean <dbl>	sd <dbl>	5.5% <dbl>	94.5% <dbl>	n_eff <dbl>	Rhat4 <dbl>
intercept	6.04	9.67	-9.44	21.55	4740	1
beta1	-0.03	0.03	-0.08	0.03	8297	1
beta2	0.00	0.00	-0.01	0.01	4750	1
beta3	0.41	0.09	0.27	0.57	7907	1
beta4	-0.10	0.22	-0.44	0.24	6246	1
beta5	-0.13	0.19	-0.42	0.19	7216	1
log_phi	-0.31	0.13	-0.52	-0.11	8691	1

Figure 3. Statistics of Poisson, Gamma, Negative Binomial (top to bottom) with default prior

THE CHOICE OF MODEL was based on the statistics explained in this paragraph. Based on the analysis and the information provided above, it was concluded that the Negative Binomial best fits the data and has an Rhat below 1.01. Poisson cannot be used because the mean and variance (624 and 1290333) are quite different (Possion assumes that mean and variance are similar). Normal cannot be used because it seems to be a heavy-tailed distribution. Negative Binomial can be used as it best fits the data and has an Rhat below 1.01. The random max log normal values were 4.855616e+17, 7287821605, 6397519, and 4059.888 for (0,10), (0,5), (0,3) and (0,2), respectively. Normal ~ (0,3) was picked because it is closer to the actual max values. Additionally, the statistics from Figure 5 imply that the models perform (lower Rhat values) better when we change the prior from Normal ~ (0,10) to Normal ~ (0,3). Hence, Normal ~ (0,3) was chosen for the final model.

$E \sim \text{NegativeBionomial}(\mu, \varphi)$ $\text{logit}(\mu) = \alpha + (\beta1 * \text{cat2} + \beta2 * \text{year} +$ $\beta3 * \text{cplx} + \beta4 * \text{acap} + \beta5 * \text{pcap})$ $\text{logit}(\varphi) = \log\varphi$ $\alpha \sim \text{Normal}(0, 3)$ $\beta1 \sim \text{Normal}(0, 3)$ $\beta2 \sim \text{Normal}(0, 3)$ $\beta3 \sim \text{Normal}(0, 3)$ $\beta4 \sim \text{Normal}(0, 3)$ $\beta5 \sim \text{Normal}(0, 3)$ $\log\varphi \sim \text{Normal}(0, 3)$	$E \sim \text{Poisson}(\lambda)$ $\text{logit}(\lambda) = \alpha + (\beta1 * \text{cat2} + \beta2 * \text{year} +$ $\beta3 * \text{cplx} + \beta4 * \text{acap} + \beta5 * \text{pcap})$ $\alpha \sim \text{Normal}(0, 3)$ $\beta1 \sim \text{Normal}(0, 3)$ $\beta2 \sim \text{Normal}(0, 3)$ $\beta3 \sim \text{Normal}(0, 3)$ $\beta4 \sim \text{Normal}(0, 3)$ $\beta5 \sim \text{Normal}(0, 3)$	$E \sim \text{Gamma}(a, x)$ $\text{logit}(a) = \alpha + (\beta1 * \text{cat2} + \beta2 * \text{year} +$ $\beta3 * \text{cplx} + \beta4 * \text{acap} + \beta5 * \text{pcap})$ $\text{logit}(x) = \log x$ $\alpha \sim \text{Normal}(0, 3)$ $\beta1 \sim \text{Normal}(0, 3)$ $\beta2 \sim \text{Normal}(0, 3)$ $\beta3 \sim \text{Normal}(0, 3)$ $\beta4 \sim \text{Normal}(0, 3)$ $\beta5 \sim \text{Normal}(0, 3)$ $\log x \sim \text{Normal}(0, 3)$

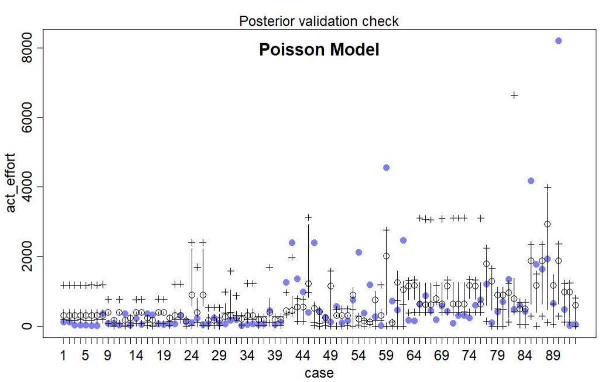
Figure 4. Mathematical notations for the Negative Binomial, Poisson and Gamma Distribution models

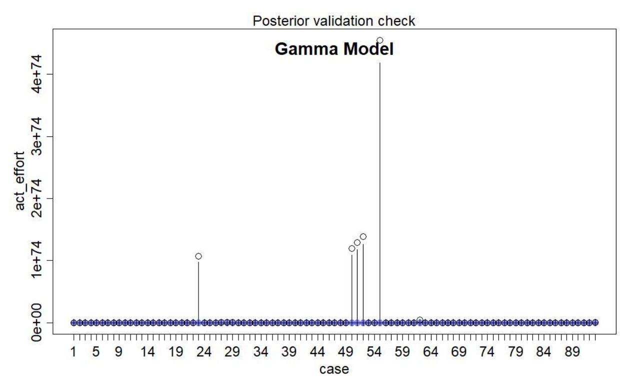
	mean <dbl>	sd <dbl>	5.5% <dbl>	94.5% <dbl>	n_eff <dbl>	Rhat4 <dbl>
intercept	5.25	9.86	-0.83	30.13	3	3.94
beta1	-0.07	0.13	-0.09	-0.04	14	1.28
beta2	0.00	0.01	-0.01	0.00	3	3.83
beta3	0.37	0.32	-0.26	0.53	13	1.38
beta4	-0.06	0.50	-0.36	0.87	2	25.85
beta5	-0.62	0.57	-1.84	-0.28	2	11.94

	mean <dbl>	sd <dbl>	5.5% <dbl>	94.5% <dbl>	n_eff <dbl>	Rhat4 <dbl>
intercept	0.58	2.57	-3.18	4.81	12	1.15
beta1	0.46	0.84	-0.06	1.91	2	39.56
beta2	0.02	0.03	0.00	0.07	2	27.55
beta3	-0.20	0.76	-1.51	0.33	2	15.32
beta4	-0.58	0.48	-1.37	-0.07	2	3.50
beta5	0.21	0.53	-0.31	1.10	2	4.16
log_scale	5.43	2.17	1.69	6.94	2	14.75

	mean <dbl>	sd <dbl>	5.5% <dbl>	94.5% <dbl>	n_eff <dbl>	Rhat4 <dbl>
intercept	0.56	3.03	-4.31	5.42	6999	1
beta1	-0.03	0.04	-0.08	0.03	7937	1
beta2	0.00	0.00	0.00	0.01	7020	1
beta3	0.42	0.09	0.27	0.57	7027	1
beta4	-0.10	0.22	-0.45	0.24	6704	1
beta5	-0.13	0.19	-0.42	0.17	6978	1
log_phi	-0.32	0.13	-0.52	-0.11	7087	1

Figure 5. Statistics of the Possion, Gamma, Negative Binomial (top to bottom) with N ~ (0,3) priors





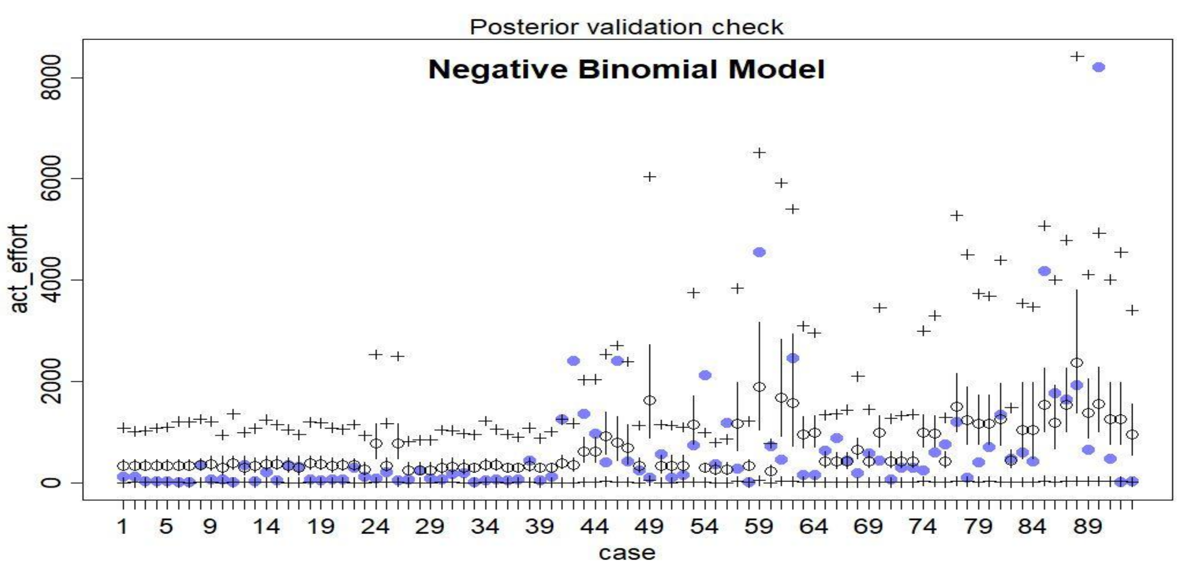


Figure 6. Posterior validation checks of all three models

POSTERIOR CHECKS have also been performed to defend our argument further. Figure 6 shows that the Gamma model does not fit the data correctly, and Poisson also does not fit the model well. Overall, the Negative Binomial model best fits the data as it models the distribution better.

	PSIS	SE	dPSIS	dSE	pPSIS	weight
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
m_negbinom_f	1.347894e+03	3.012428e+01	0.000000e+00	NA	6.949740e+00	1
m_poisson_f	5.000696e+05	1.153220e+05	4.987217e+05	1.153065e+05	2.364693e+05	0
m_gamma_f	1.331255e+81	1.218349e+78	1.331255e+81	1.218349e+78	6.656273e+80	0

	WAIC	SE	dWAIC	dSE	pWAIC	weight
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
m_negbinom_f	1.347424e+03	2.983574e+01	0.000000e+00	NA	6.714634e+00	1
m_poisson_f	2.581051e+08	1.339830e+08	2.581038e+08	134709227	1.290390e+08	0
m_gamma_f	1.999842e+157	NaN	1.999842e+157	Inf	9.999212e+156	0

Figure 7. LOO (top) and WAIC (bottom) model comparisons

MODEL COMPARISON has been performed using both LOO and WAIC; the Negative Binomial model has the lowest value of PSIS and the highest weight. It also has the lowest value of WAIC and the highest weight. In contrast, the Poisson and Gamma models have much higher PSIS and WAIC values with much lower weights. These factors reinforce the fact that Negative binomial is the best-fitting model for the data.

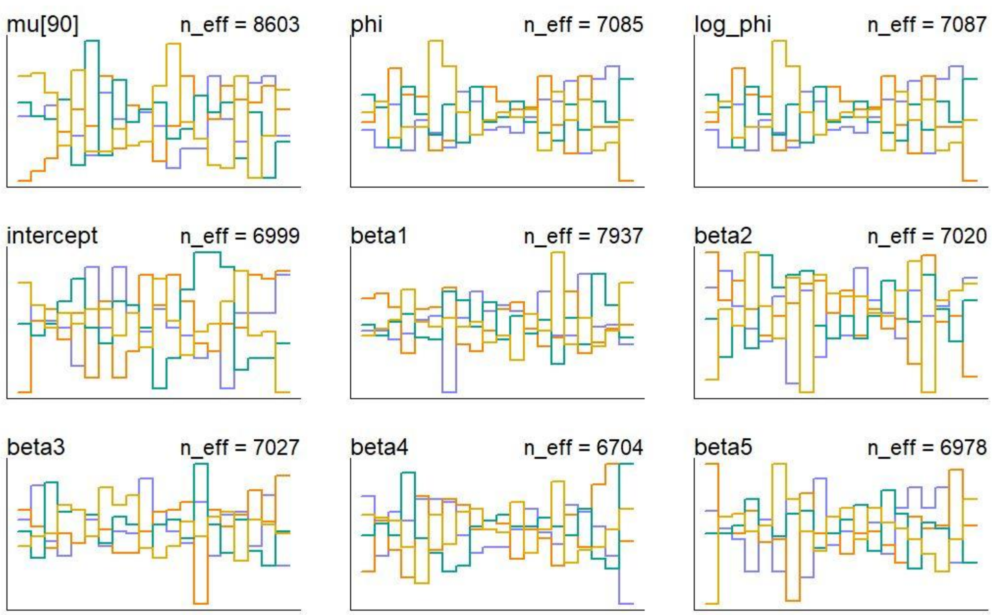


Figure 8. Trank plots of the final model (Negative Binomial)

The trank plots of the final Negative Binomial Model appear to be good as chains in the center of the plot are well mixed; this indicates good convergence and mixing of the samples. It also suggests that the chains have explored the posterior distribution effectively and are providing reliable estimates.

THE CAUSAL GRAPH drawn (figure 9) based on our understanding of the features, represented in the form of a DAG (directed acyclic graph), shows that all 5 features (category, year, complexity, analyst capability, programmer capability) have a direct causal implication on the actual effort required. It also shows other interesting relationships, such as the category of a project having a causal implication on its complexity.

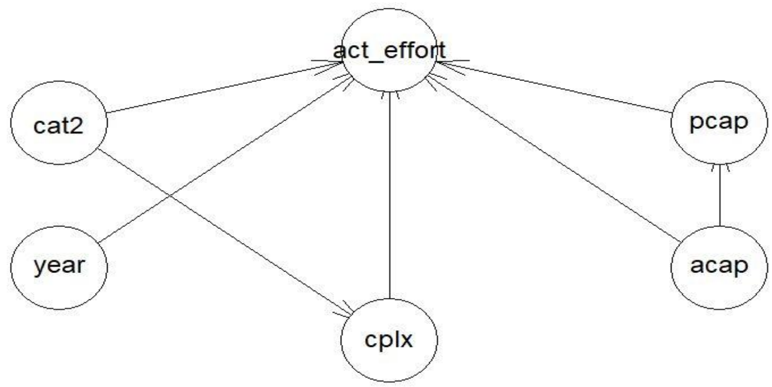


Figure 9. Causal graph of the features

IN CONCLUSION, the Bayesian analysis compared three different models: Negative Binomial, Poisson, and Gamma. The results consistently favored the Negative Binomial model over the other two. This suggests that the Negative Binomial distribution is better suited for modeling the effort required in software projects based on the given dataset. This was evidenced by several factors, including the posterior validation checks showing a better fit for the Negative Binomial model compared to the Poisson and Gamma models, lower PSIS and WAIC values for the Negative Binomial model, and higher weights assigned to it during model comparison.

Several predictors were considered, including project category, year, complexity, analyst capability, and programmer capability. By incorporating these predictors into the model, the analysis accounted for the influence of experience and other factors on project effort estimation. This allows for a more comprehensive and accurate estimation process, as it takes into account the varying levels of expertise and project characteristics.

PRACTICAL IMPLICATIONS of the results suggest that using the Negative Binomial distribution, along with the identified predictors, can lead to more reliable and accurate estimates of project effort in software engineering. This knowledge can help practitioners and project managers make informed decisions regarding resource allocation, scheduling, and budgeting. Organizations can improve their project planning and control processes by considering the factors that influence effort estimation, leading to more successful outcomes.

References:

[1] Afzal, W., Ghazi, A.N., Itkonen, J. et al. An experiment on the effectiveness and efficiency of exploratory testing. Empir Software Eng 20, 844–878 (2015). https://doi.org/10.1007/s10664-014-9301-4

[2] https://github.com/UVic-Data-Science-for-SE/course/blob/main/Assignments/a2-files/nasa93_subset.csv

[3] Tim Menzies. (2008). nasa93 [Data set]. Zenodo. https://doi.org/10.5281/zenodo.268419

Appendix:
Source Code is attached below

Bayesian Data Analytics on NASA software projects dataset (For Data Science and Software Engineering (CSC 578A) - Assignment 2

Anuinder Sekhon (V01022326) and Sri Surya Jitendra Palaparty (V01022218)

Version: 2023-06-10 05:31:02.492128

Initial setup

Load necessary libraries

```
library(RWeka)
library(rethinking)
library(posterior)
library(rstanarm)
library(rstan)
library(tidyverse)
library(ggplot2)
library(bayesplot)
library(foreign)
library(here)
library(coda)
library(mvtnorm)
library(devtools)
library(loo)
library(dagitty)
library(cmdstanr)
library(dplyr)
library(tidyverse)
library(GGally)
library(vioplot)
library(MASS)
library(ggdag)
library(bnlearn)
```

Question 1:

Reading csv

```
nasa_df <- read.csv("nasa93_subset.csv")
```

Descriptive statistics

```
# Before conversion
str(nasa_df)
```

```
## 'data.frame':    93 obs. of  8 variables:
##  $ X           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ recordnumber: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ cat2        : chr  "avionicsmonitoring" "avionicsmonitoring" "avionicsmonitoring" "avionicsmonito
##  $ year        : int  1979 1979 1979 1979 1979 1979 1979 1982 1980 1980 ...
##  $ cplx        : chr  "h" "h" "h" "h" ...
##  $ acap        : chr  "n" "n" "n" "n" ...
##  $ pcap        : chr  "n" "n" "n" "n" ...
##  $ act_effort  : num  117.6 117.6 31.2 36 25.2 ...
```

```
head(nasa_df)
```

```
##   X recordnumber               cat2 year cplx acap pcap act_effort
## 1 1            1 avionicsmonitoring 1979    h    n    n      117.6
## 2 2            2 avionicsmonitoring 1979    h    n    n      117.6
## 3 3            3 avionicsmonitoring 1979    h    n    n       31.2
## 4 4            4 avionicsmonitoring 1979    h    n    n       36.0
## 5 5            5 avionicsmonitoring 1979    h    n    n       25.2
## 6 6            6 avionicsmonitoring 1979    h    n    n        8.4
```

```
summary(nasa_df)
```

```
##        X       recordnumber        cat2                year     
##  Min.   : 1   Min.   :  1.00   Length:93          Min.   :1971  
##  1st Qu.:24   1st Qu.: 24.00   Class :character   1st Qu.:1979  
##  Median :47   Median : 47.00   Mode  :character   Median :1980  
##  Mean   :47   Mean   : 47.75                      Mean   :1981  
##  3rd Qu.:70   3rd Qu.: 70.00                      3rd Qu.:1983  
##  Max.   :93   Max.   :101.00                      Max.   :1987  
##      cplx               acap               pcap             act_effort    
##  Length:93          Length:93          Length:93          Min.   :   8.4  
##  Class :character   Class :character   Class :character   1st Qu.:  70.0  
##  Mode  :character   Mode  :character   Mode  :character   Median : 252.0  
##                                                           Mean   : 624.4  
##                                                           3rd Qu.: 600.0  
##                                                           Max.   :8211.0
```

```
# Convert categorical variables into numeric
nasa_df <- nasa_df %>%
  mutate(
    cat2 = as.numeric(as.factor(cat2)),
    cplx = as.numeric(as.factor(cplx)),
    acap = as.numeric(as.factor(acap)),
    pcap = as.numeric(as.factor(pcap))
  )
```

```r
# Convert to numeric
nasa_df$year <- as.numeric(nasa_df$year)
nasa_df$X <- as.numeric(nasa_df$X)
nasa_df$recordnumber <- as.numeric(nasa_df$recordnumber)

# After conversion
str(nasa_df)
```

```
## 'data.frame':    93 obs. of  8 variables:
##  $ X           : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ recordnumber: num  1 2 3 4 5 6 7 8 9 10 ...
##  $ cat2        : num  3 3 3 3 3 3 3 3 8 8 ...
##  $ year        : num  1979 1979 1979 1979 1979 ...
##  $ cplx        : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ acap        : num  2 2 2 2 2 2 2 2 1 1 ...
##  $ pcap        : num  2 2 2 2 2 2 2 2 1 3 ...
##  $ act_effort  : num  117.6 117.6 31.2 36 25.2 ...
```

```r
head(nasa_df)
```

```
##   X recordnumber cat2 year cplx acap pcap act_effort
## 1 1            1    3 1979    1    2    2      117.6
## 2 2            2    3 1979    1    2    2      117.6
## 3 3            3    3 1979    1    2    2       31.2
## 4 4            4    3 1979    1    2    2       36.0
## 5 5            5    3 1979    1    2    2       25.2
## 6 6            6    3 1979    1    2    2        8.4
```

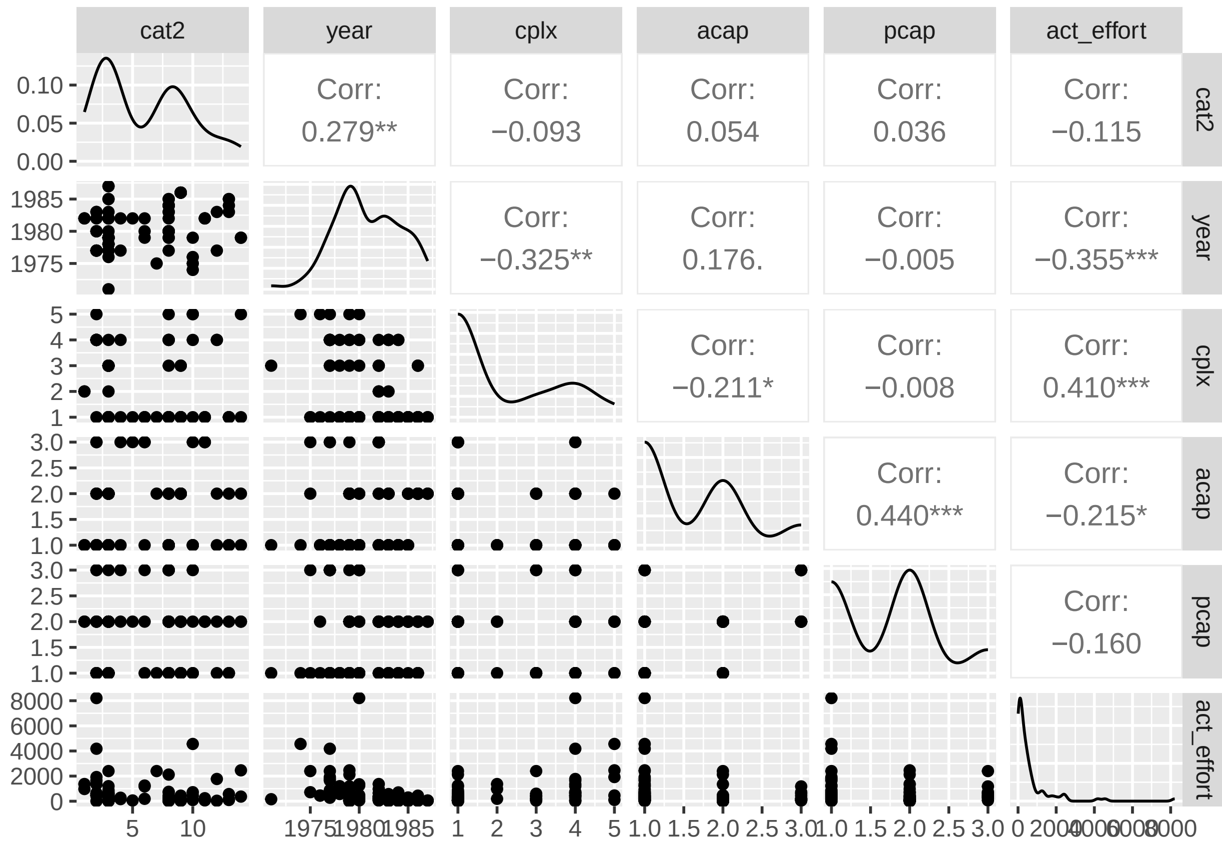
```r
summary(nasa_df)
```

```
##        X       recordnumber         cat2             year           cplx      
##  Min.   : 1   Min.   :  1.00   Min.   : 1.000   Min.   :1971   Min.   :1.000  
##  1st Qu.:24   1st Qu.: 24.00   1st Qu.: 3.000   1st Qu.:1979   1st Qu.:1.000  
##  Median :47   Median : 47.00   Median : 6.000   Median :1980   Median :1.000  
##  Mean   :47   Mean   : 47.75   Mean   : 6.032   Mean   :1981   Mean   :2.011  
##  3rd Qu.:70   3rd Qu.: 70.00   3rd Qu.: 8.000   3rd Qu.:1983   3rd Qu.:3.000  
##  Max.   :93   Max.   :101.00   Max.   :14.000   Max.   :1987   Max.   :5.000  
##       acap            pcap         act_effort    
##  Min.   :1.000   Min.   :1.000   Min.   :   8.4  
##  1st Qu.:1.000   1st Qu.:1.000   1st Qu.:  70.0  
##  Median :1.000   Median :2.000   Median : 252.0  
##  Mean   :1.559   Mean   :1.688   Mean   : 624.4  
##  3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.: 600.0  
##  Max.   :3.000   Max.   :3.000   Max.   :8211.0
```

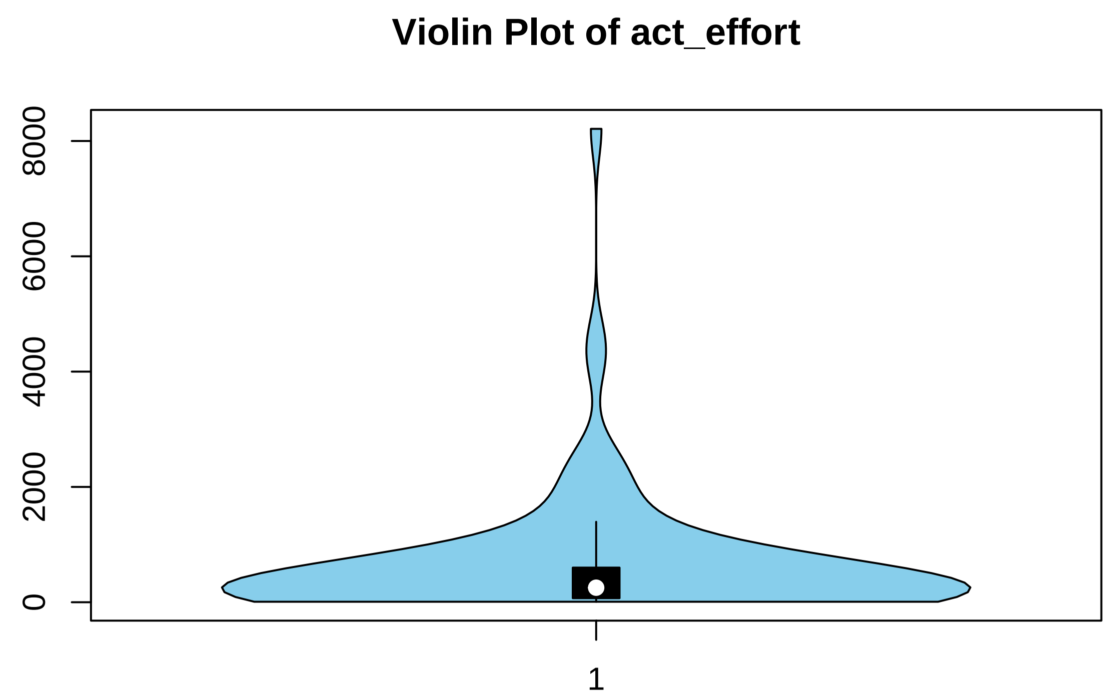
```r
# Did not normalize because the values are relatively in similar ranges
```

Data Visualization

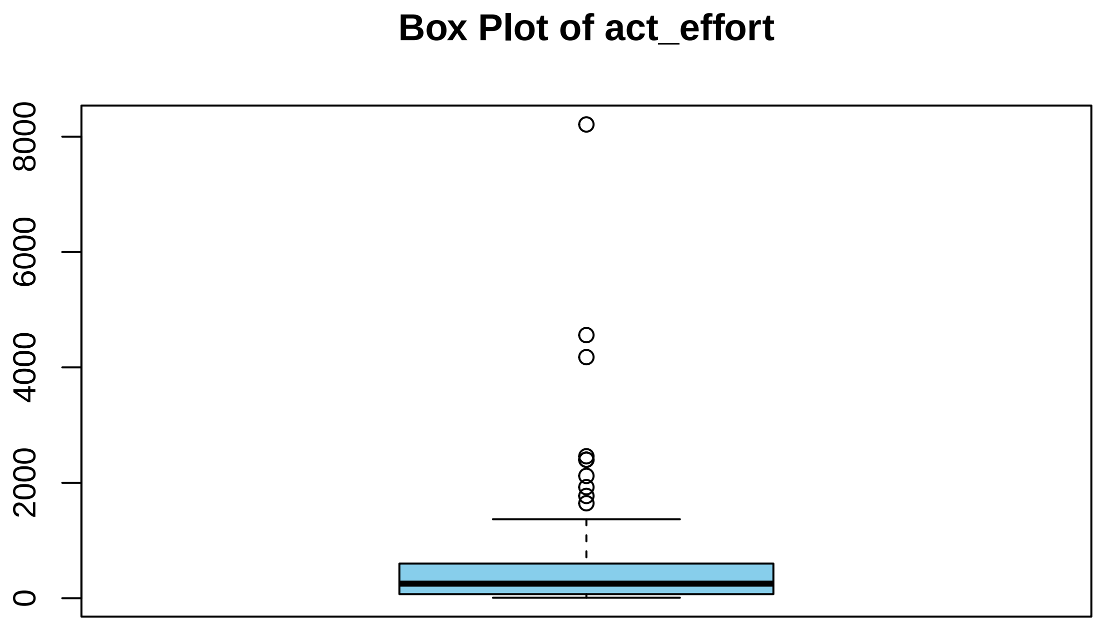
```r
# Scatter plot and correlation matrix
ggpairs(nasa_df[, c("cat2","year", "cplx", "acap", "pcap", "act_effort")])
```



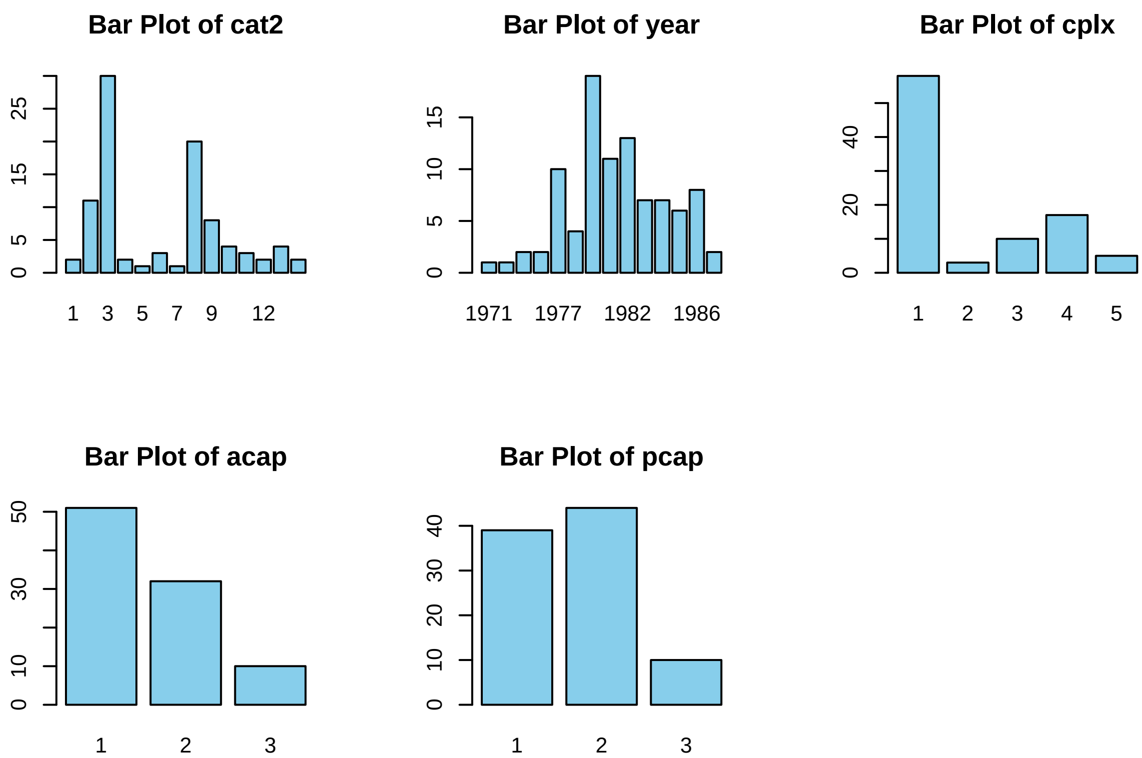

```r
# Violin plot for 'act_effort'
vioplot(nasa_df$act_effort, col = "skyblue", main = "Violin Plot of act_effort")
```




```r
# Box plot for 'act_effort'
boxplot(nasa_df$act_effort, col = "skyblue", main = "Box Plot of act_effort")
```




```r
# Create a 2x3 grid of plots
par(mfrow = c(2, 3))

# Bar plot for "cat2"
cat2_counts <- table(nasa_df$cat2)
barplot(cat2_counts, col = "skyblue", main = "Bar Plot of cat2")

# Bar plot for "year"
year_counts <- table(nasa_df$year)
barplot(year_counts, col = "skyblue", main = "Bar Plot of year")

# Bar plot for "cplx"
cplx_counts <- table(nasa_df$cplx)
barplot(cplx_counts, col = "skyblue", main = "Bar Plot of cplx")

# Bar plot for "acap"
acap_counts <- table(nasa_df$acap)
barplot(acap_counts, col = "skyblue", main = "Bar Plot of acap")

# Bar plot for "pcap"
pcap_counts <- table(nasa_df$pcap)
barplot(pcap_counts, col = "skyblue", main = "Bar Plot of pcap")
```



Question 2:

Bayesian Analysis

Likelihoods

```
# Using ulam to fit different distributions with different likelihoods

m_normal <- ulam(
  alist(
    act_effort ~ normal(mu, sigma),
    mu <- intercept + beta1 * cat2 + beta2 * year + beta3 * cplx + beta4 * acap + beta5 * pcap,
    intercept ~ normal(0, 10),
    beta1 ~ normal(0, 10),
    beta2 ~ normal(0, 10),
    beta3 ~ normal(0, 10),
    beta4 ~ normal(0, 10),
    beta5 ~ normal(0, 10),
    sigma ~ exponential(1)
  ),
  data = nasa_df,
  cores = 4,
  chains = 4,
```

```
  cmdstan = TRUE,
  log_lik = TRUE,
  iter = 5e3
)

m_poisson <- ulam(
  alist(
    act_effort ~ poisson(lambda),
    log(lambda) <- intercept + beta1 * cat2 + beta2 * year + beta3 * cplx + beta4 * acap + beta5 * pcap
    intercept ~ normal(0, 10),
    beta1 ~ normal(0, 10),
    beta2 ~ normal(0, 10),
    beta3 ~ normal(0, 10),
    beta4 ~ normal(0, 10),
    beta5 ~ normal(0, 10)
  ),
  data = nasa_df,
  cores = 4,
  chains = 4,
  cmdstan = TRUE,
  log_lik = TRUE,
  iter = 5e3
)

m_gamma <- ulam(
  alist(
    act_effort ~ gamma(shape, scale),
    log(shape) <- intercept + beta1 * cat2 + beta2 * year + beta3 * cplx + beta4 * acap + beta5 * pcap,
    log(scale) <- log_scale,
    intercept ~ normal(0, 10),
    beta1 ~ normal(0, 10),
    beta2 ~ normal(0, 10),
    beta3 ~ normal(0, 10),
    beta4 ~ normal(0, 10),
    beta5 ~ normal(0, 10),
    log_scale ~ normal(0, 10)
  ),
  data = nasa_df,
  cores = 4,
  chains = 4,
  cmdstan = TRUE,
  log_lik = TRUE,
  iter = 5e3
)

m_negbinom <- ulam(
  alist(
    act_effort ~ neg_binomial_2(mu, phi),
    log(mu) <- intercept + beta1 * cat2 + beta2 * year + beta3 * cplx + beta4 * acap + beta5 * pcap,
    log(phi) <- log_phi,
    intercept ~ normal(0, 10),
    beta1 ~ normal(0, 10),
    beta2 ~ normal(0, 10),
```

```r
    beta3 ~ normal(0, 10),
    beta4 ~ normal(0, 10),
    beta5 ~ normal(0, 10),
    log_phi ~ normal(0, 10)
  ),
  data = nasa_df,
  cores = 4,
  chains = 4,
  cmdstan = TRUE,
  log_lik = TRUE,
  iter = 5e3
)

# Model Statistics

precis(m_normal)
precis(m_poisson)
precis(m_gamma)
precis(m_negbinom)

# Pick Negative Binomial as it best fits the data and has an Rhat below 1.01
```

```r
# Calculate mean and variance

var(nasa_df$act_effort)
```

```
## [1] 1290333
```

```r
mean(nasa_df$act_effort)
```

```
## [1] 624.4118
```

```r
# Cannot pick Poisson because the mean and variance are quite different
# Cannot pick Normal because it seems to be a heavy tailed distribution
# Pick Negative Binomial as it best fits the data and has an Rhat below 1.01
```

Question 3:

Priors

```r
# Default prior (very high)
max(rlnorm(1e5, 0, 10))
```

```
## [1] 4.855616e+17
```

```r
# Normal (0,5) (still high)
max(rlnorm(1e5, 0, 5))
```

```
## [1] 7287821605
```

```r
# Normal (0,3) (reasonable)
max(rlnorm(1e5, 0, 3))
```

```
## [1] 6397519
```

```r
# Normal (0,2) (very low)
max(rlnorm(1e5, 0, 2))
```

```
## [1] 4059.888
```

```r
# Intercept only

# Normal (0,3)

m_negbinom_p1 <- ulam(
  alist(
    act_effort ~ neg_binomial_2(mu, phi),
    log(mu) <- intercept,
    log(phi) <- log_phi,
    intercept ~ normal(0, 3),
    log_phi ~ normal(0, 3)
  ),
  data = nasa_df,
  cores = 4,
  chains = 4,
  cmdstan = TRUE,
  log_lik = TRUE,
  iter = 5e3
)

# Normal (0,10)

m_negbinom_p2 <- ulam(
  alist(
    act_effort ~ neg_binomial_2(mu, phi),
    log(mu) <- intercept,
    log(phi) <- log_phi,
    intercept ~ normal(0, 10),
    log_phi ~ normal(0, 10)
  ),
  data = nasa_df,
  cores = 4,
  chains = 4,
  cmdstan = TRUE,
  log_lik = TRUE,
  iter = 5e3
)

precis(m_negbinom_p1)
precis(m_negbinom_p2)
```

```r
# Both perform similarly but we pick Normal (0,3) as its closer to max values
```

Question 4 and 5:

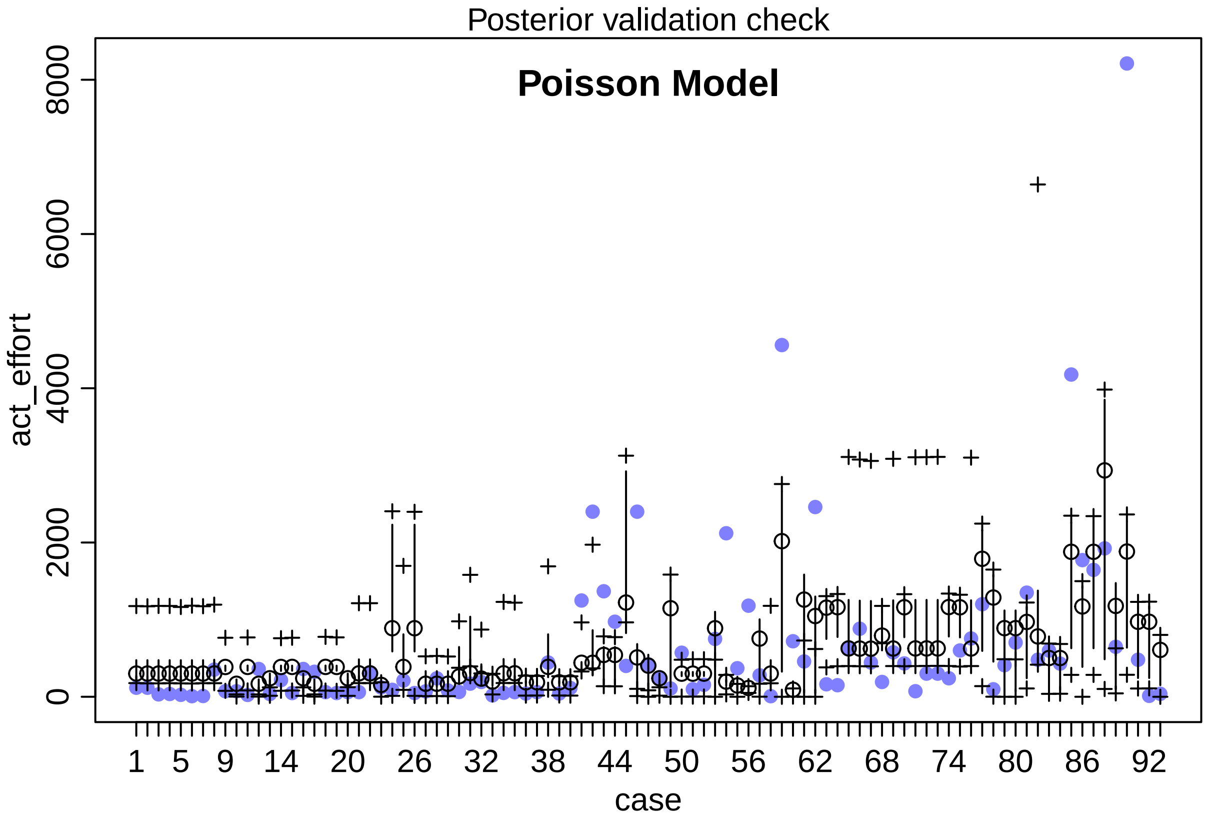
Calculating the posterior

```r
# Testing models with the selected priors

m_poisson_f <- ulam(
  alist(
    act_effort ~ poisson(lambda),
    log(lambda) <- intercept + beta1 * cat2 + beta2 * year + beta3 * cplx + beta4 * acap + beta5 * pcap
    intercept ~ normal(0, 3),
    beta1 ~ normal(0, 3),
    beta2 ~ normal(0, 3),
    beta3 ~ normal(0, 3),
    beta4 ~ normal(0, 3),
    beta5 ~ normal(0, 3)
  ),
  data = nasa_df,
  cores = 4,
  chains = 4,
  cmdstan = TRUE,
  log_lik = TRUE,
  iter = 5e3
)

m_gamma_f <- ulam(
  alist(
    act_effort ~ gamma(shape, scale),
    log(shape) <- intercept + beta1 * cat2 + beta2 * year + beta3 * cplx + beta4 * acap + beta5 * pcap,
    log(scale) <- log_scale,
    intercept ~ normal(0, 3),
    beta1 ~ normal(0, 3),
    beta2 ~ normal(0, 3),
    beta3 ~ normal(0, 3),
    beta4 ~ normal(0, 3),
    beta5 ~ normal(0, 3),
    log_scale ~ normal(0, 3)
  ),
  data = nasa_df,
  cores = 4,
  chains = 4,
  cmdstan = TRUE,
  log_lik = TRUE,
  iter = 5e3
)

m_negbinom_f <- ulam(
  alist(
```

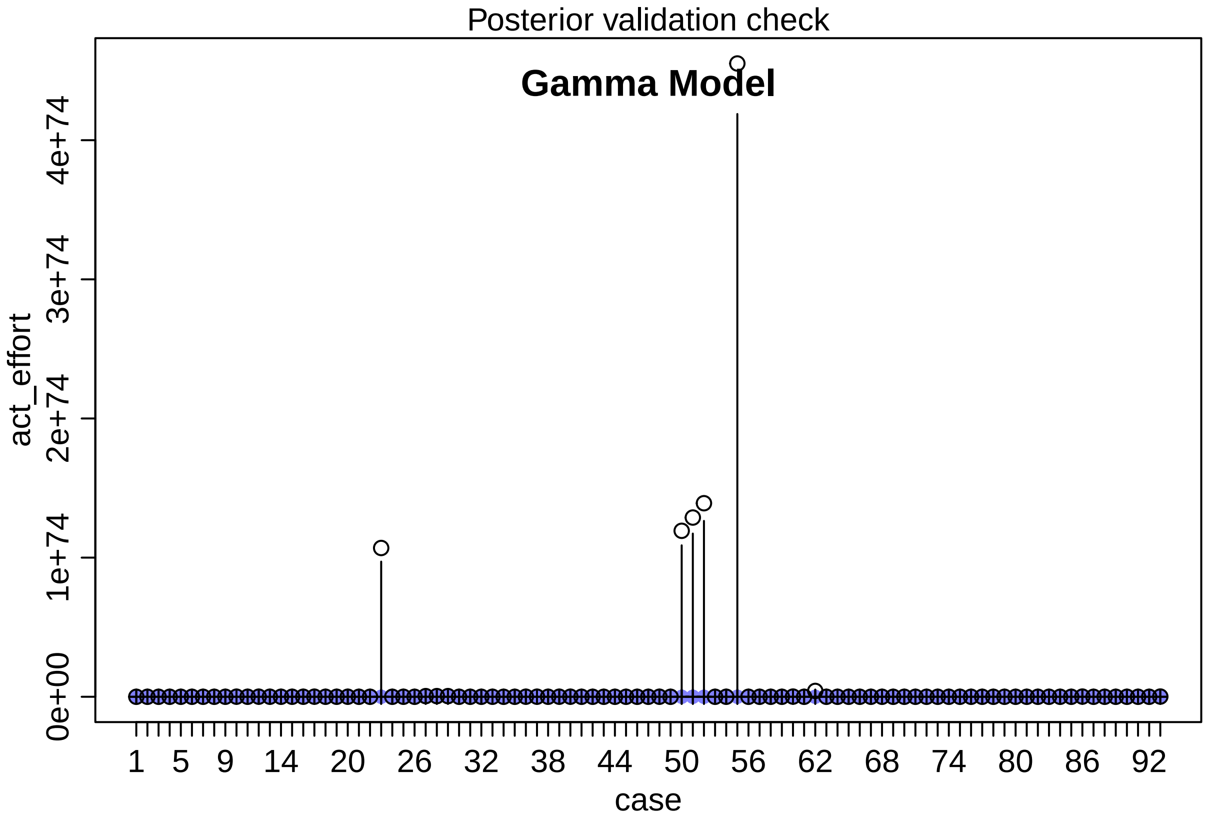
```r
    act_effort ~ neg_binomial_2(mu, phi),
    log(mu) <- intercept + beta1 * cat2 + beta2 * year + beta3 * cplx + beta4 * acap + beta5 * pcap,
    log(phi) <- log_phi,
    intercept ~ normal(0, 3),
    beta1 ~ normal(0, 3),
    beta2 ~ normal(0, 3),
    beta3 ~ normal(0, 3),
    beta4 ~ normal(0, 3),
    beta5 ~ normal(0, 3),
    log_phi ~ normal(0, 3)
  ),
  data = nasa_df,
  cores = 4,
  chains = 4,
  cmdstan = TRUE,
  log_lik = TRUE,
  iter = 5e3
)

# Model Statistics

precis(m_poisson_f)
precis(m_gamma_f)
precis(m_negbinom_f)
```

Sanity check of the posterior

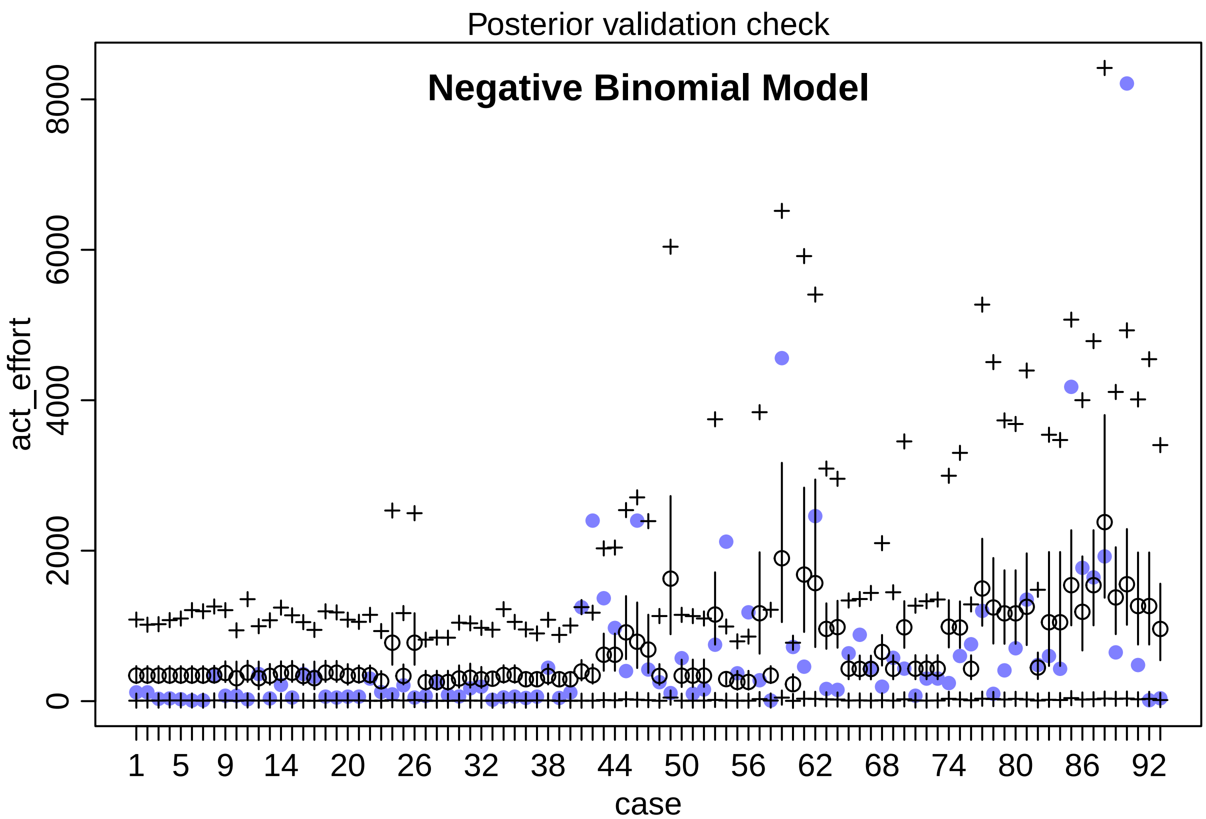
```r
postcheck(m_poisson_f, window=94)
title("Poisson Model", line = -1.5)
```




```r
postcheck(m_gamma_f, window=94)
title("Gamma Model", line = -1.5)
```




```
postcheck(m_negbinom_f, window=94)
title("Negative Binomial Model", line = -1.5)
```



Model comparisons

```r
# Compare the models using LOO
loo_est <- rethinking::compare(m_poisson_f, m_gamma_f, m_negbinom_f, func=LOO)

# Compare the models using WAIC
waic_est <- rethinking::compare(m_poisson_f, m_gamma_f, m_negbinom_f, func=WAIC)

# Print the comparison results
print(loo_est)
```

```
##                        PSIS           SE        dPSIS          dSE        pPSIS
## m_negbinom_f 1.347894e+03 3.012428e+01 0.000000e+00           NA 6.949740e+00
## m_poisson_f  5.000696e+05 1.153220e+05 4.987217e+05 1.153065e+05 2.364693e+05
## m_gamma_f    1.331255e+81 1.218349e+78 1.331255e+81 1.218349e+78 6.656273e+80
##              weight
## m_negbinom_f      1
## m_poisson_f       0
## m_gamma_f         0
```

```r
print(waic_est)
```

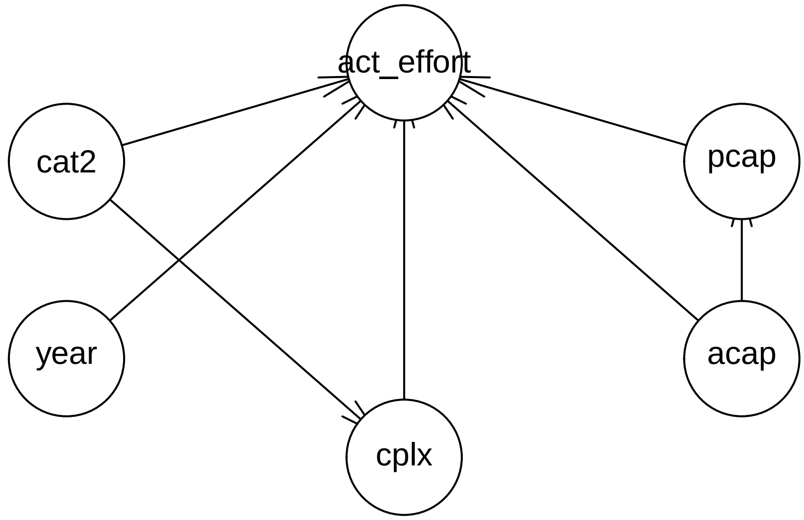
```
##                      WAIC           SE          dWAIC       dSE          pWAIC
## m_negbinom_f  1.347424e+03 2.983574e+01  0.000000e+00        NA  6.714634e+00
## m_poisson_f   2.581051e+08 1.339830e+08  2.581038e+08 134709227  1.290390e+08
## m_gamma_f    1.999842e+157          NaN 1.999842e+157       Inf 9.999212e+156
##              weight
## m_negbinom_f      1
## m_poisson_f       0
## m_gamma_f         0
```

```
# Negative Binomial is the best model from these scores
```

Question 6:

Causal Graph

```
# Create a Bayesian network structure
dag <- empty.graph(nodes = c("cat2", "year", "cplx", "acap", "pcap", "act_effort"))

# Specify the directed edges
arcs(dag) <- matrix(c("cat2", "act_effort",
                      "year", "act_effort",
                      "cplx", "act_effort",
                      "acap", "act_effort",
                      "pcap", "act_effort",
                      "cat2", "cplx",
                      "acap", "pcap"), ncol = 2, byrow = TRUE)

# Plot the causal graph
plot(dag)
```



Question 7:

Model Diagnostics

```r
# Summary statistics

summary(m_negbinom_f)
```

```
## Inference for Stan model: ulam_cmdstanr_57474790a21da4c6acb423357c9e2910-202306100549-1-378428.
## 4 chains, each with iter=5000; warmup=2500; thin=1;
## post-warmup draws per chain=2500, total post-warmup draws=10000.
##
##                 mean se_mean   sd  2.5%   25%   50%   75% 97.5%
## intercept       0.56    0.04 3.03 -5.41 -1.46  0.55  2.60  6.56
## beta1          -0.03    0.00 0.04 -0.10 -0.05 -0.03 -0.01  0.04
## beta2           0.00    0.00 0.00  0.00  0.00  0.00  0.00  0.01
## beta3           0.42    0.00 0.09  0.24  0.35  0.41  0.48  0.60
## beta4          -0.10    0.00 0.22 -0.52 -0.25 -0.10  0.04  0.33
## beta5          -0.13    0.00 0.19 -0.48 -0.25 -0.13  0.00  0.25
## log_phi        -0.32    0.00 0.13 -0.57 -0.40 -0.31 -0.23 -0.06
## log_lik[1]     -6.25    0.00 0.11 -6.47 -6.32 -6.24 -6.18 -6.05
## log_lik[2]     -6.25    0.00 0.11 -6.47 -6.32 -6.24 -6.18 -6.05
## log_lik[3]     -5.71    0.00 0.15 -6.02 -5.80 -5.70 -5.60 -5.43
## log_lik[4]     -5.76    0.00 0.14 -6.05 -5.85 -5.75 -5.66 -5.49
```

```
## log_lik[5]    -5.64    0.00  0.16  -5.97  -5.74  -5.63  -5.53  -5.34
## log_lik[6]    -5.30    0.00  0.23  -5.80  -5.45  -5.29  -5.14  -4.89
## log_lik[7]    -5.37    0.00  0.21  -5.82  -5.50  -5.35  -5.22  -4.98
## log_lik[8]    -7.07    0.00  0.08  -7.23  -7.12  -7.07  -7.01  -6.92
## log_lik[9]    -6.07    0.00  0.14  -6.36  -6.16  -6.07  -5.98  -5.82
## log_lik[10]   -5.94    0.00  0.21  -6.39  -6.07  -5.92  -5.79  -5.59
## log_lik[11]   -5.69    0.00  0.18  -6.08  -5.81  -5.68  -5.56  -5.36
## log_lik[12]   -7.17    0.00  0.15  -7.57  -7.22  -7.14  -7.07  -6.96
## log_lik[13]   -5.74    0.00  0.17  -6.10  -5.85  -5.73  -5.62  -5.43
## log_lik[14]   -6.66    0.00  0.10  -6.86  -6.72  -6.65  -6.59  -6.48
## log_lik[15]   -5.92    0.00  0.15  -6.24  -6.02  -5.91  -5.82  -5.65
## log_lik[16]   -7.10    0.00  0.09  -7.29  -7.16  -7.10  -7.04  -6.95
## log_lik[17]   -7.04    0.00  0.13  -7.35  -7.09  -7.02  -6.96  -6.85
## log_lik[18]   -6.01    0.00  0.14  -6.30  -6.10  -6.00  -5.91  -5.74
## log_lik[19]   -5.92    0.00  0.15  -6.24  -6.02  -5.91  -5.82  -5.65
## log_lik[20]   -5.93    0.00  0.15  -6.25  -6.03  -5.92  -5.82  -5.65
## log_lik[21]   -5.95    0.00  0.12  -6.21  -6.04  -5.95  -5.87  -5.72
## log_lik[22]   -6.91    0.00  0.08  -7.07  -6.96  -6.91  -6.85  -6.76
## log_lik[23]   -6.15    0.00  0.12  -6.42  -6.23  -6.14  -6.07  -5.94
## log_lik[24]   -6.60    0.00  0.18  -6.98  -6.72  -6.59  -6.47  -6.27
## log_lik[25]   -6.61    0.00  0.10  -6.81  -6.67  -6.60  -6.54  -6.44
## log_lik[26]   -6.39    0.00  0.21  -6.84  -6.53  -6.38  -6.24  -6.01
## log_lik[27]   -5.84    0.00  0.16  -6.19  -5.94  -5.82  -5.72  -5.56
## log_lik[28]   -6.70    0.00  0.09  -6.89  -6.76  -6.69  -6.64  -6.54
## log_lik[29]   -5.92    0.00  0.15  -6.26  -6.01  -5.90  -5.81  -5.65
## log_lik[30]   -5.87    0.00  0.17  -6.25  -5.98  -5.86  -5.75  -5.57
## log_lik[31]   -6.44    0.00  0.12  -6.71  -6.51  -6.43  -6.35  -6.24
## log_lik[32]   -6.51    0.00  0.10  -6.74  -6.57  -6.50  -6.44  -6.34
## log_lik[33]   -5.45    0.00  0.15  -5.76  -5.54  -5.44  -5.34  -5.18
## log_lik[34]   -5.89    0.00  0.13  -6.15  -5.97  -5.88  -5.80  -5.65
## log_lik[35]   -5.96    0.00  0.12  -6.21  -6.04  -5.95  -5.87  -5.73
## log_lik[36]   -5.71    0.00  0.11  -5.95  -5.79  -5.71  -5.63  -5.51
## log_lik[37]   -5.85    0.00  0.11  -6.07  -5.92  -5.85  -5.78  -5.66
## log_lik[38]   -7.35    0.00  0.11  -7.59  -7.41  -7.34  -7.28  -7.17
## log_lik[39]   -5.71    0.00  0.11  -5.95  -5.79  -5.71  -5.63  -5.51
## log_lik[40]   -6.16    0.00  0.09  -6.36  -6.23  -6.16  -6.10  -5.99
## log_lik[41]   -9.15    0.00  0.37 -10.02  -9.37  -9.10  -8.88  -8.58
## log_lik[42]  -12.20    0.01  1.11 -14.74 -12.85 -12.09 -11.40 -10.38
## log_lik[43]   -8.80    0.00  0.26  -9.43  -8.93  -8.76  -8.62  -8.43
## log_lik[44]   -8.21    0.00  0.15  -8.56  -8.28  -8.19  -8.11  -7.99
## log_lik[45]   -7.37    0.00  0.13  -7.64  -7.45  -7.36  -7.28  -7.15
## log_lik[46]   -9.89    0.01  0.62 -11.45 -10.22  -9.77  -9.44  -9.06
## log_lik[47]   -7.31    0.00  0.12  -7.59  -7.38  -7.30  -7.23  -7.12
## log_lik[48]   -6.76    0.00  0.09  -6.94  -6.81  -6.75  -6.70  -6.59
## log_lik[49]   -7.13    0.00  0.25  -7.66  -7.30  -7.12  -6.96  -6.69
## log_lik[50]   -7.76    0.00  0.24  -8.41  -7.86  -7.70  -7.60  -7.46
## log_lik[51]   -6.15    0.00  0.17  -6.53  -6.25  -6.13  -6.02  -5.85
## log_lik[52]   -6.41    0.00  0.14  -6.72  -6.49  -6.39  -6.31  -6.18
## log_lik[53]   -7.87    0.00  0.10  -8.08  -7.93  -7.87  -7.81  -7.70
## log_lik[54]  -12.11    0.01  0.87 -14.02 -12.66 -12.03 -11.50 -10.63
## log_lik[55]   -7.23    0.00  0.15  -7.62  -7.30  -7.20  -7.12  -7.01
## log_lik[56]  -10.09    0.01  0.91 -12.21 -10.62  -9.95  -9.43  -8.74
## log_lik[57]   -7.28    0.00  0.20  -7.72  -7.40  -7.27  -7.15  -6.95
## log_lik[58]   -5.30    0.00  0.23  -5.80  -5.45  -5.29  -5.14  -4.89
```

```
## log_lik[59]  -10.18    0.00   0.45  -11.30  -10.39  -10.08   -9.85   -9.62
## log_lik[60]   -8.72    0.01   0.56  -10.08   -9.02   -8.62   -8.31   -7.92
## log_lik[61]   -7.72    0.00   0.18   -8.12   -7.82   -7.70   -7.59   -7.41
## log_lik[62]   -9.25    0.00   0.32  -10.12   -9.35   -9.15   -9.05   -8.91
## log_lik[63]   -6.96    0.00   0.12   -7.21   -7.04   -6.96   -6.88   -6.75
## log_lik[64]   -6.95    0.00   0.12   -7.21   -7.03   -6.94   -6.86   -6.73
## log_lik[65]   -7.75    0.00   0.12   -8.03   -7.81   -7.73   -7.66   -7.55
## log_lik[66]   -8.28    0.00   0.21   -8.77   -8.39   -8.25   -8.13   -7.97
## log_lik[67]   -7.31    0.00   0.08   -7.48   -7.36   -7.30   -7.25   -7.15
## log_lik[68]   -6.82    0.00   0.10   -7.04   -6.89   -6.82   -6.75   -6.63
## log_lik[69]   -7.61    0.00   0.11   -7.85   -7.67   -7.60   -7.54   -7.43
## log_lik[70]   -7.45    0.00   0.10   -7.64   -7.51   -7.44   -7.38   -7.27
## log_lik[71]   -6.16    0.00   0.14   -6.45   -6.25   -6.15   -6.06   -5.90
## log_lik[72]   -6.94    0.00   0.09   -7.13   -7.00   -6.94   -6.88   -6.78
## log_lik[73]   -6.94    0.00   0.09   -7.13   -7.00   -6.94   -6.88   -6.78
## log_lik[74]   -7.15    0.00   0.11   -7.37   -7.21   -7.14   -7.07   -6.95
## log_lik[75]   -7.66    0.00   0.09   -7.84   -7.72   -7.66   -7.60   -7.50
## log_lik[76]   -8.01    0.00   0.16   -8.39   -8.10   -7.99   -7.90   -7.77
## log_lik[77]   -8.31    0.00   0.08   -8.48   -8.36   -8.30   -8.25   -8.15
## log_lik[78]   -6.94    0.00   0.20   -7.37   -7.07   -6.92   -6.79   -6.57
## log_lik[79]   -7.49    0.00   0.13   -7.76   -7.57   -7.48   -7.40   -7.26
## log_lik[80]   -7.83    0.00   0.10   -8.04   -7.89   -7.82   -7.76   -7.64
## log_lik[81]   -8.44    0.00   0.10   -8.66   -8.49   -8.43   -8.37   -8.27
## log_lik[82]   -7.39    0.00   0.09   -7.57   -7.45   -7.39   -7.33   -7.23
## log_lik[83]   -7.70    0.00   0.15   -8.05   -7.77   -7.67   -7.59   -7.48
## log_lik[84]   -7.46    0.00   0.19   -7.90   -7.57   -7.43   -7.33   -7.19
## log_lik[85]  -10.15    0.00   0.35  -10.99  -10.35  -10.09   -9.90   -9.64
## log_lik[86]   -8.82    0.00   0.19   -9.31   -8.90   -8.78   -8.70   -8.58
## log_lik[87]   -8.62    0.00   0.09   -8.80   -8.68   -8.62   -8.56   -8.46
## log_lik[88]   -8.79    0.00   0.09   -8.99   -8.85   -8.78   -8.73   -8.62
## log_lik[89]   -7.82    0.00   0.11   -8.07   -7.90   -7.82   -7.74   -7.62
## log_lik[90]  -12.35    0.01   0.87  -14.36  -12.86  -12.24  -11.73  -10.97
## log_lik[91]   -7.61    0.00   0.14   -7.91   -7.70   -7.60   -7.51   -7.37
## log_lik[92]   -6.34    0.00   0.34   -7.06   -6.56   -6.32   -6.10   -5.73
## log_lik[93]   -6.46    0.00   0.26   -7.02   -6.63   -6.44   -6.27   -5.99
## mu[1]        340.65    0.74  71.72  223.11  289.13  332.89  382.51  506.01
## mu[2]        340.65    0.74  71.72  223.11  289.13  332.89  382.51  506.01
## mu[3]        340.65    0.74  71.72  223.11  289.13  332.89  382.51  506.01
## mu[4]        340.65    0.74  71.72  223.11  289.13  332.89  382.51  506.01
## mu[5]        340.65    0.74  71.72  223.11  289.13  332.89  382.51  506.01
## mu[6]        340.65    0.74  71.72  223.11  289.13  332.89  382.51  506.01
## mu[7]        340.65    0.74  71.72  223.11  289.13  332.89  382.51  506.01
## mu[8]        343.46    0.74  72.40  224.87  291.62  335.37  385.80  510.11
## mu[9]        374.45    0.98  89.39  233.31  311.35  362.47  425.91  584.51
## mu[10]       304.49    1.67 126.52  137.24  216.12  278.40  362.99  618.56
## mu[11]       378.61    0.99  90.67  235.67  314.50  366.50  430.45  591.92
## mu[12]       304.49    1.67 126.52  137.24  216.12  278.40  362.99  618.56
## mu[13]       336.29    1.13  89.70  199.39  272.25  323.19  385.64  550.79
## mu[14]       374.45    0.98  89.39  233.31  311.35  362.47  425.91  584.51
## mu[15]       377.56    0.99  90.34  235.27  313.81  365.39  429.21  590.36
## mu[16]       333.52    1.12  88.80  198.27  270.06  320.46  382.53  546.25
## mu[17]       304.49    1.67 126.52  137.24  216.12  278.40  362.99  618.56
## mu[18]       378.61    0.99  90.67  235.67  314.50  366.50  430.45  591.92
## mu[19]       377.56    0.99  90.34  235.27  313.81  365.39  429.21  590.36
```

```
## mu[20]        335.36    1.13  89.39  199.01  271.59  322.11  384.67  549.25
## mu[21]        346.30    0.75  73.12  226.69  293.92  338.31  388.93  514.30
## mu[22]        346.30    0.75  73.12  226.69  293.92  338.31  388.93  514.30
## mu[23]        264.76    0.84  76.46  152.81  210.90  253.77  304.03  445.86
## mu[24]        776.78    2.79 225.28  443.43  616.13  744.71  899.14 1306.04
## mu[25]        335.25    0.95  85.07  204.19  275.56  321.87  383.71  534.27
## mu[26]        776.78    2.79 225.28  443.43  616.13  744.71  899.14 1306.04
## mu[27]        253.99    0.90  84.21  135.69  195.48  238.87  295.80  458.08
## mu[28]        253.99    0.90  84.21  135.69  195.48  238.87  295.80  458.08
## mu[29]        253.99    0.90  84.21  135.69  195.48  238.87  295.80  458.08
## mu[30]        300.85    1.05  97.17  161.89  231.66  284.59  350.11  538.18
## mu[31]        310.80    1.14 104.46  161.54  236.45  293.13  364.07  564.28
## mu[32]        291.58    0.98  91.45  161.05  226.56  275.65  337.79  516.70
## mu[33]        298.10    0.52  51.50  213.95  261.60  292.72  328.09  416.95
## mu[34]        348.21    0.76  73.63  227.90  295.29  340.18  391.31  517.05
## mu[35]        348.21    0.76  73.63  227.90  295.29  340.18  391.31  517.05
## mu[36]        291.17    0.56  54.33  204.01  252.59  284.71  322.62  416.27
## mu[37]        291.17    0.56  54.33  204.01  252.59  284.71  322.62  416.27
## mu[38]        335.25    0.95  85.07  204.19  275.56  321.87  383.71  534.27
## mu[39]        291.17    0.56  54.33  204.01  252.59  284.71  322.62  416.27
## mu[40]        291.17    0.56  54.33  204.01  252.59  284.71  322.62  416.27
## mu[41]        395.02    0.95  87.66  254.61  332.72  383.94  446.44  599.55
## mu[42]        343.27    0.91  81.60  215.67  285.48  330.84  389.33  534.01
## mu[43]        616.46    1.87 162.07  368.47  504.44  592.92  703.85  995.35
## mu[44]        616.46    1.87 162.07  368.47  504.44  592.92  703.85  995.35
## mu[45]        914.39    3.42 274.56  508.90  719.97  871.59 1063.54 1559.82
## mu[46]        789.46    3.65 298.14  385.34  583.51  734.00  929.60 1522.83
## mu[47]        685.48    3.22 260.35  338.83  504.86  634.78  809.22 1332.21
## mu[48]        335.36    1.13  89.39  199.01  271.59  322.11  384.67  549.25
## mu[49]       1628.12    7.28 605.94  783.58 1193.32 1513.91 1931.11 3103.36
## mu[50]        337.71    1.38 121.22  166.90  253.14  315.69  395.95  638.50
## mu[51]        338.64    1.38 121.58  167.22  253.84  316.56  397.09  640.22
## mu[52]        339.58    1.39 121.96  167.67  254.76  317.28  398.17  642.44
## mu[53]       1151.42    3.42 309.11  690.42  933.65 1102.76 1308.80 1898.84
## mu[54]        293.26    0.50  50.55  210.50  257.56  288.31  322.68  408.36
## mu[55]        255.34    0.91  82.19  137.82  197.49  242.12  296.40  454.36
## mu[56]        255.14    0.78  81.76  139.28  197.03  241.59  296.38  453.49
## mu[57]       1168.36    5.38 457.95  555.58  854.87 1080.18 1375.44 2317.37
## mu[58]        340.65    0.74  71.72  223.11  289.13  332.89  382.51  506.01
## mu[59]       1899.42    8.25 706.14  932.46 1409.70 1764.37 2243.08 3654.19
## mu[60]        225.44    0.70  73.30  121.71  174.08  213.05  261.41  407.99
## mu[61]       1682.55    7.13 628.70  824.44 1243.88 1562.76 1971.75 3269.30
## mu[62]       1568.90    9.00 757.58  629.64 1058.63 1398.17 1875.03 3536.55
## mu[63]        960.14    1.93 192.49  650.54  825.43  937.57 1070.03 1401.44
## mu[64]        984.02    1.97 197.73  664.86  844.52  960.35 1095.26 1439.19
## mu[65]        430.84    1.12 103.73  267.83  357.30  416.82  489.33  670.51
## mu[66]        428.47    1.11 102.99  266.48  355.40  414.53  486.79  666.88
## mu[67]        427.29    1.11 102.63  266.07  354.40  413.50  485.35  664.25
## mu[68]        654.46    1.32 130.39  442.72  563.54  639.60  729.98  954.57
## mu[69]        429.65    1.11 103.36  267.13  356.29  415.60  487.95  668.48
## mu[70]        981.33    1.97 197.09  663.73  842.08  958.06 1092.63 1435.06
## mu[71]        430.84    1.12 103.73  267.83  357.30  416.82  489.33  670.51
## mu[72]        430.84    1.12 103.73  267.83  357.30  416.82  489.33  670.51
## mu[73]        430.84    1.12 103.73  267.83  357.30  416.82  489.33  670.51
```

```
## mu[74]       989.43    1.99 199.05  668.17  848.65  965.12 1100.89 1446.09
## mu[75]       978.65    1.96 196.47  662.40  840.07  955.67 1089.80 1431.02
## mu[76]       429.65    1.11 103.36  267.13  356.29  415.60  487.95  668.48
## mu[77]      1497.34    4.01 370.80  927.83 1234.30 1446.55 1697.94 2382.40
## mu[78]      1244.96    4.23 373.52  695.90  982.19 1183.53 1437.04 2134.84
## mu[79]      1167.13    3.46 313.05  699.44  945.54 1118.95 1328.10 1927.05
## mu[80]      1167.13    3.46 313.05  699.44  945.54 1118.95 1328.10 1927.05
## mu[81]      1253.93    4.71 397.70  673.77  972.59 1193.02 1458.63 2206.22
## mu[82]       446.23    1.24 114.99  268.80  365.04  430.04  509.45  714.09
## mu[83]      1048.68    6.03 521.46  400.02  693.49  935.87 1267.80 2347.50
## mu[84]      1048.68    6.03 521.46  400.02  693.49  935.87 1267.80 2347.50
## mu[85]      1541.48    4.36 403.24  927.07 1254.68 1487.83 1756.32 2507.71
## mu[86]      1187.27    4.91 421.94  598.84  892.59 1113.26 1387.05 2248.16
## mu[87]      1541.48    4.36 403.24  927.07 1254.68 1487.83 1756.32 2507.71
## mu[88]      2379.88    9.09 792.94 1242.39 1825.10 2250.24 2770.24 4323.09
## mu[89]      1378.57    3.97 373.42  817.77 1118.93 1325.38 1572.71 2252.68
## mu[90]      1554.13    4.38 406.64  931.91 1265.67 1500.12 1770.72 2532.93
## mu[91]      1264.13    4.73 400.69  679.53  980.85 1202.88 1471.09 2223.10
## mu[92]      1264.13    4.73 400.69  679.53  980.85 1202.88 1471.09 2223.10
## mu[93]       959.60    3.97 336.60  487.87  721.07  900.09 1126.04 1766.49
## phi            0.74    0.00   0.10    0.57    0.67    0.73    0.80    0.94
## lp__        -670.45    0.03   1.90 -675.00 -671.53 -670.16 -669.05 -667.70
##             n_eff Rhat
## intercept    6999    1
## beta1        7937    1
## beta2        7020    1
## beta3        7027    1
## beta4        6704    1
## beta5        6978    1
## log_phi      7087    1
## log_lik[1]   8740    1
## log_lik[2]   8740    1
## log_lik[3]   9408    1
## log_lik[4]   9528    1
## log_lik[5]   9214    1
## log_lik[6]   8339    1
## log_lik[7]   8468    1
## log_lik[8]   6504    1
## log_lik[9]   8723    1
## log_lik[10]  6167    1
## log_lik[11]  8255    1
## log_lik[12]  5935    1
## log_lik[13]  6655    1
## log_lik[14]  7535    1
## log_lik[15]  8653    1
## log_lik[16]  6298    1
## log_lik[17]  5770    1
## log_lik[18]  8764    1
## log_lik[19]  8653    1
## log_lik[20]  6550    1
## log_lik[21]  9684    1
## log_lik[22]  6491    1
## log_lik[23]  7852    1
## log_lik[24]  6973    1
```

```
## log_lik[25]  6994    1
## log_lik[26]  7154    1
## log_lik[27]  9136    1
## log_lik[28]  5947    1
## log_lik[29]  8886    1
## log_lik[30]  8947    1
## log_lik[31]  7504    1
## log_lik[32]  7044    1
## log_lik[33]  9063    1
## log_lik[34]  9681    1
## log_lik[35]  9679    1
## log_lik[36]  9842    1
## log_lik[37]  9792    1
## log_lik[38]  6908    1
## log_lik[39]  9842    1
## log_lik[40]  8372    1
## log_lik[41]  9021    1
## log_lik[42]  8163    1
## log_lik[43]  7662    1
## log_lik[44]  7116    1
## log_lik[45]  6560    1
## log_lik[46]  7441    1
## log_lik[47]  6438    1
## log_lik[48]  6414    1
## log_lik[49]  7633    1
## log_lik[50]  6782    1
## log_lik[51]  8074    1
## log_lik[52]  7596    1
## log_lik[53]  6901    1
## log_lik[54]  9424    1
## log_lik[55]  7187    1
## log_lik[56] 11398    1
## log_lik[57]  7707    1
## log_lik[58]  8339    1
## log_lik[59]  8176    1
## log_lik[60] 11070    1
## log_lik[61]  8014    1
## log_lik[62]  7650    1
## log_lik[63] 10407    1
## log_lik[64] 10528    1
## log_lik[65]  7728    1
## log_lik[66]  8813    1
## log_lik[67]  6420    1
## log_lik[68]  9625    1
## log_lik[69]  7278    1
## log_lik[70]  8378    1
## log_lik[71]  9021    1
## log_lik[72]  6901    1
## log_lik[73]  6901    1
## log_lik[74] 10050    1
## log_lik[75]  7373    1
## log_lik[76]  8431    1
## log_lik[77]  6598    1
## log_lik[78]  8441    1
```

```
## log_lik[79]  7882    1
## log_lik[80]  7043    1
## log_lik[81]  6958    1
## log_lik[82]  6351    1
## log_lik[83]  6473    1
## log_lik[84]  7327    1
## log_lik[85]  9432    1
## log_lik[86]  7602    1
## log_lik[87]  6785    1
## log_lik[88]  6266    1
## log_lik[89]  7727    1
## log_lik[90]  9182    1
## log_lik[91]  7061    1
## log_lik[92]  7802    1
## log_lik[93]  7946    1
## mu[1]        9502    1
## mu[2]        9502    1
## mu[3]        9502    1
## mu[4]        9502    1
## mu[5]        9502    1
## mu[6]        9502    1
## mu[7]        9502    1
## mu[8]        9498    1
## mu[9]        8400    1
## mu[10]       5761    1
## mu[11]       8398    1
## mu[12]       5761    1
## mu[13]       6280    1
## mu[14]       8400    1
## mu[15]       8399    1
## mu[16]       6286    1
## mu[17]       5761    1
## mu[18]       8398    1
## mu[19]       8399    1
## mu[20]       6284    1
## mu[21]       9489    1
## mu[22]       9489    1
## mu[23]       8341    1
## mu[24]       6541    1
## mu[25]       8089    1
## mu[26]       6541    1
## mu[27]       8764    1
## mu[28]       8764    1
## mu[29]       8764    1
## mu[30]       8539    1
## mu[31]       8399    1
## mu[32]       8674    1
## mu[33]       9972    1
## mu[34]       9481    1
## mu[35]       9481    1
## mu[36]       9543    1
## mu[37]       9543    1
## mu[38]       8089    1
## mu[39]       9543    1
```

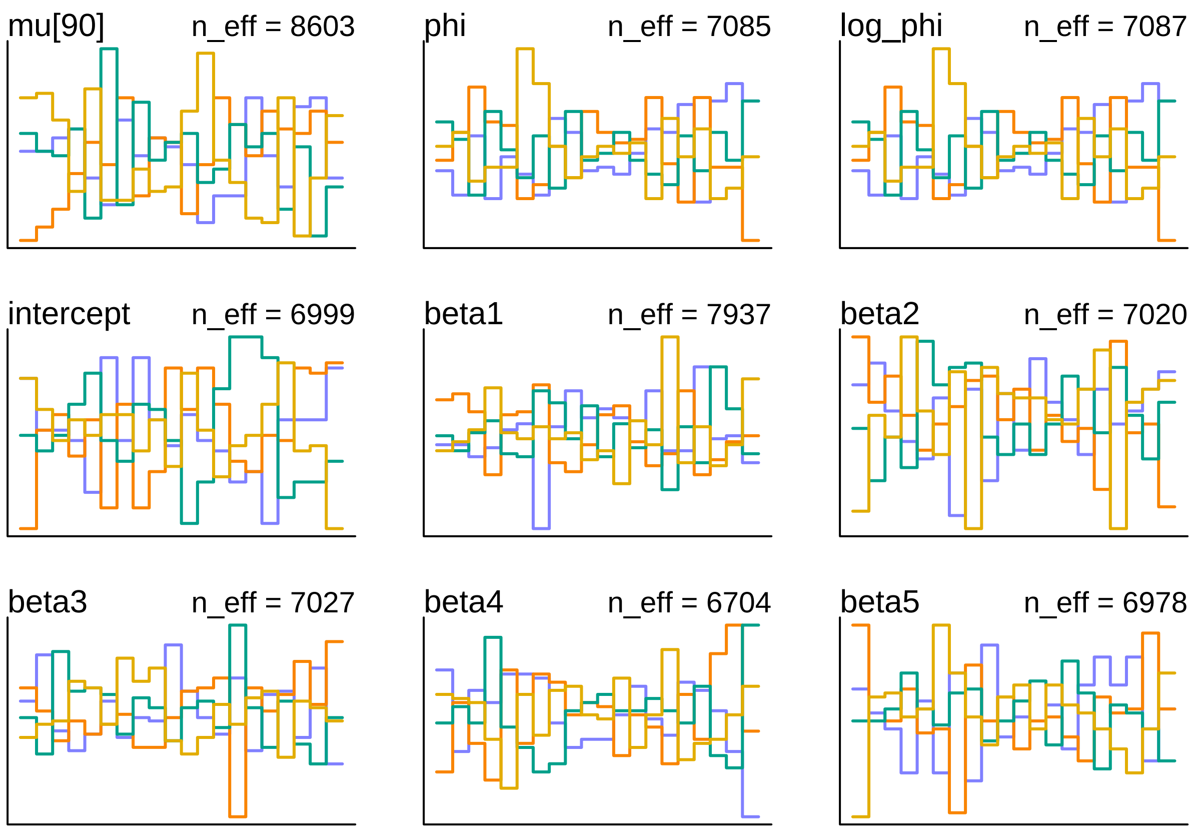
```
## mu[40]        9543     1
## mu[41]        8591     1
## mu[42]        8025     1
## mu[43]        7517     1
## mu[44]        7517     1
## mu[45]        6451     1
## mu[46]        6671     1
## mu[47]        6524     1
## mu[48]        6284     1
## mu[49]        6925     1
## mu[50]        7744     1
## mu[51]        7744     1
## mu[52]        7744     1
## mu[53]        8166     1
## mu[54]       10036     1
## mu[55]        8180     1
## mu[56]       10949     1
## mu[57]        7239     1
## mu[58]        9502     1
## mu[59]        7334     1
## mu[60]       10865     1
## mu[61]        7779     1
## mu[62]        7089     1
## mu[63]        9965     1
## mu[64]       10032     1
## mu[65]        8600     1
## mu[66]        8597     1
## mu[67]        8595     1
## mu[68]        9831     1
## mu[69]        8599     1
## mu[70]       10026     1
## mu[71]        8600     1
## mu[72]        8600     1
## mu[73]        8600     1
## mu[74]       10041     1
## mu[75]       10020     1
## mu[76]        8599     1
## mu[77]        8559     1
## mu[78]        7809     1
## mu[79]        8166     1
## mu[80]        8166     1
## mu[81]        7117     1
## mu[82]        8554     1
## mu[83]        7481     1
## mu[84]        7481     1
## mu[85]        8548     1
## mu[86]        7382     1
## mu[87]        8548     1
## mu[88]        7615     1
## mu[89]        8842     1
## mu[90]        8603     1
## mu[91]        7166     1
## mu[92]        7166     1
## mu[93]        7183     1
```

```
## phi          7085    1
## lp__         4048    1
##
## Samples were drawn using NUTS(diag_e) at Sat Jun 10 5:51:30 AM 2023.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
precis(m_negbinom_f)
```

```
##                  mean          sd          5.5%        94.5%    n_eff
## intercept  0.556222436 3.029628637 -4.3107516000  5.416517400 6998.666
## beta1     -0.028532178 0.035044606 -0.0836030225  0.028061207 7937.029
## beta2      0.002718877 0.001543939  0.0002491828  0.005195026 7019.920
## beta3      0.415681008 0.094672028  0.2677967450  0.567255035 7027.359
## beta4     -0.101503957 0.215463160 -0.4455532750  0.243966520 6704.052
## beta5     -0.127300338 0.185511093 -0.4215274250  0.169995950 6978.091
## log_phi   -0.315775778 0.129137236 -0.5248961050 -0.108636260 7086.769
##               Rhat4
## intercept 1.0003442
## beta1     1.0003609
## beta2     1.0003140
## beta3     0.9999171
## beta4     1.0001161
## beta5     0.9999814
## log_phi   1.0002706
```

```
# Trank plots

selected_vars <- c("mu[90]", "phi", "log_phi", "intercept", "beta1", "beta2", "beta3", "beta4", "beta5")

trankplot(m_negbinom_f, pars = selected_vars )
```



Environment

```r
#CORES = 4 # set to the number of available CPU cores
#remotes::install_github("stan-dev/cmdstanr")
#cmdstanr::install_cmdstan(cores = CORES)
# you can now run rethinking with cmdstan instead of rstan
```

```r
cmdstanr::cmdstan_version()
```

```
## [1] "2.32.2"
```

```r
print(sessionInfo(), locale=FALSE)
```

```
## R version 4.3.0 (2023-04-21 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 11 x64 (build 22621)
##
## Matrix products: default
##
##
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods
```

```
## [8] base
##
## other attached packages:
##  [1] digest_0.6.31      bnlearn_4.8.3      ggdag_0.2.10
##  [4] MASS_7.3-60        vioplot_0.4.0      zoo_1.8-12
##  [7] sm_2.2-5.7.1       GGally_2.1.2       dagitty_0.3-1
## [10] loo_2.6.0          devtools_2.4.5     usethis_2.2.0
## [13] mvtnorm_1.2-2      coda_0.19-4        here_1.0.1
## [16] foreign_0.8-84     bayesplot_1.10.0   lubridate_1.9.2
## [19] forcats_1.0.0      stringr_1.5.0      dplyr_1.1.2
## [22] purrr_1.0.1        readr_2.1.4        tidyr_1.3.0
## [25] tibble_3.2.1       tidyverse_2.0.0    rstanarm_2.21.4
## [28] Rcpp_1.0.10        posterior_1.4.1    rethinking_2.31
## [31] cmdstanr_0.5.3     rstan_2.21.8       ggplot2_3.4.2
## [34] StanHeaders_2.26.26 RWeka_0.4-46
##
## loaded via a namespace (and not attached):
##   [1] RColorBrewer_1.1-3   tensorA_0.36.2       rstudioapi_0.14
##   [4] jsonlite_1.8.5       shape_1.4.6          magrittr_2.0.3
##   [7] farver_2.1.1         nloptr_2.0.3         rmarkdown_2.21
##  [10] fs_1.6.2             RWekajars_3.9.3-2    vctrs_0.6.2
##  [13] memoise_2.0.1        minqa_1.2.5          base64enc_0.1-3
##  [16] htmltools_0.5.5      curl_5.0.0           distributional_0.3.2
##  [19] htmlwidgets_1.6.2    plyr_1.8.8           cachem_1.0.8
##  [22] igraph_1.4.2         mime_0.12            lifecycle_1.0.3
##  [25] pkgconfig_2.0.3      colourpicker_1.2.0   Matrix_1.5-4
##  [28] R6_2.5.1             fastmap_1.1.1        shiny_1.7.4
##  [31] reshape_0.8.9        colorspace_2.1-0     ps_1.7.5
##  [34] rprojroot_2.0.3      pkgload_1.3.2        crosstalk_1.2.0
##  [37] labeling_0.4.2       fansi_1.0.4          timechange_0.2.0
##  [40] abind_1.4-5          compiler_4.3.0       remotes_2.4.2
##  [43] withr_2.5.0          backports_1.4.1      inline_0.3.19
##  [46] shinystan_2.6.0      highr_0.10           pkgbuild_1.4.0
##  [49] sessioninfo_1.2.2    gtools_3.9.4         tools_4.3.0
##  [52] httpuv_1.6.11        threejs_0.3.3        glue_1.6.2
##  [55] callr_3.7.3          nlme_3.1-162         promises_1.2.0.1
##  [58] grid_4.3.0           checkmate_2.2.0      reshape2_1.4.4
##  [61] generics_0.1.3       gtable_0.3.3         tzdb_0.4.0
##  [64] data.table_1.14.8    hms_1.1.3            tidygraph_1.2.3
##  [67] utf8_1.2.3           pillar_1.9.0         markdown_1.7
##  [70] later_1.3.1          rJava_1.0-6          splines_4.3.0
##  [73] lattice_0.21-8       survival_3.5-5       tidyselect_1.2.0
##  [76] miniUI_0.1.1.1       knitr_1.42           gridExtra_2.3
##  [79] V8_4.3.0             stats4_4.3.0         xfun_0.39
##  [82] matrixStats_1.0.0    DT_0.28              stringi_1.7.12
##  [85] yaml_2.3.7           boot_1.3-28.1        evaluate_0.21
##  [88] codetools_0.2-19     cli_3.6.1            RcppParallel_5.1.7
##  [91] shinythemes_1.2.0    xtable_1.8-4         munsell_0.5.0
##  [94] processx_3.8.1       rstantools_2.3.1     ellipsis_0.3.2
##  [97] prettyunits_1.1.1    dygraphs_1.1.1.6     profvis_0.3.8
## [100] urlchecker_1.0.1     lme4_1.1-33          scales_1.2.1
## [103] xts_0.13.1           crayon_1.5.2         rlang_1.1.1
## [106] shinyjs_2.1.0
```