Bayesian Modeling Using Stan

Jim Albert

2020-07-16

Contents

Chapter 1

Introduction to the brms Package

In **Probability and Bayesian Modeling**, the JAGS software is illustrated to fit various Bayesian models by Markov Chain Monte Carlo (MCMC) methods. JAGS consists of a mix of conjugate, Gibbs sampling, and Metropolis algorithms. In recent years, Hamiltonian sampling and the associated Stan software are becoming popular in fitting Bayesian models by MCMC.

The purpose of this supplement is to illustrate Bayesian fitting of common models using the `brms` package which is a popular interface for the Stan software.

1.1 Installing the `brms` package

Basic information about installing the `brms` package is available at https://github.com/paul-buerkner/brms

Since the package is an interface to the Stan software, a C++ compiler is required.

1.2 One Bayesian fitting function `brm()`

One attractive feature of the `brms` package is that one function `brm()` can be used to fit all of the models described in **Probability and Bayesian Modeling**.

The basic function syntax of the `brm()` function is:

```
brm(model_description,
    data = my_data,
    family = the_family,
```

```
   prior = the_prior)
```

where

- `model_description` is the description of the regression model including any random effects similar to the notation used in the `glm()` and `glmer` functions
- `my_data` is the data frame containing the data
- `family` is the sampling family (normal, binomial, Poisson, etc)
- `prior` is the specification of the prior on the regression terms and the error standard deviation

The output of the `brm()` function is an object of class brmsfit that contains the posterior samples and other information about the model.

———————————————————

Chapter 2

Binomial Modeling

2.1 Packages for example

```r
library(ProbBayes)
library(brms)
library(dplyr)
library(ggplot2)
```

2.2 Example

Suppose a sample of $n = 20$ college students are asked if they plan on wearing masks while attending class. Let p denote the proportion of all students who plan on wearing masks.

2.3 Prior on proportion

Suppose you believe that $p = 0.40$ and you are 90 percent sure that $p < 0.60$.

Use `beta.select()` from the `ProbBayes` package to find the shape parameters of the matching beta curve prior.

```r
beta.select(list(x = 0.4, p = 0.5),
            list(x = 0.6, p = 0.9))
```

```
## [1] 4.31 6.30
```

A beta(4.31, 6.30) prior represents one's beliefs about the proportion p.

2.4 Prior on the logit parameter

Since we will writing a model in terms of the logit function

$$\theta = \log\left(\frac{p}{1-p}\right)$$

We want to find a corresponding normal prior on θ.

A simple way of doing this is by simulation …

1. Simulate 1000 draws from the beta prior on p.
2. Compute θ on these simulated draws of p.
3. Find the sample mean and standard deviation of these draws – those will be estimates of the mean and standard deviation of the normal prior on θ.

```r
set.seed(123)
p_sim <- rbeta(1000, 4.31, 6.30)
theta_sim <- log(p_sim / (1 - p_sim))
c(mean(theta_sim), sd(theta_sim))
```

```
## [1] -0.4000904  0.6540093
```

The corresponding prior on the logit parameter θ is assumed to be normal with mean -0.400 and standard deviation 0.654.

2.5 Fitting the model

The model is $y_1, ..., y_{20}$ are a random sample from a Bernoulli distribution with probability p where p has the logistic representation.

$$\log\left(\frac{p}{1-p}\right) = \theta$$

where $\theta \sim N(-0.400, 0.654)$.

We put the twenty binary responses in a data frame.

```r
bdata <- data.frame(y = c(1, 0, 1, 0, 0, 0, 1,
0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0))
```

We use the `brm()` function from the `brms` package to fit the model.

```r
fit <- brm(data = bdata,
           family = bernoulli,
           y  ~ 0 + Intercept,
           prior = c(prior(normal(-0.400, 0.654),
                           coef = Intercept)),
```

```
            iter = 1000,
            refresh = 0)
```

Compiling Stan program...

Start sampling

The `plot()` function will display a density plot and a trace plot of the intercept θ.

```
plot(fit)
```
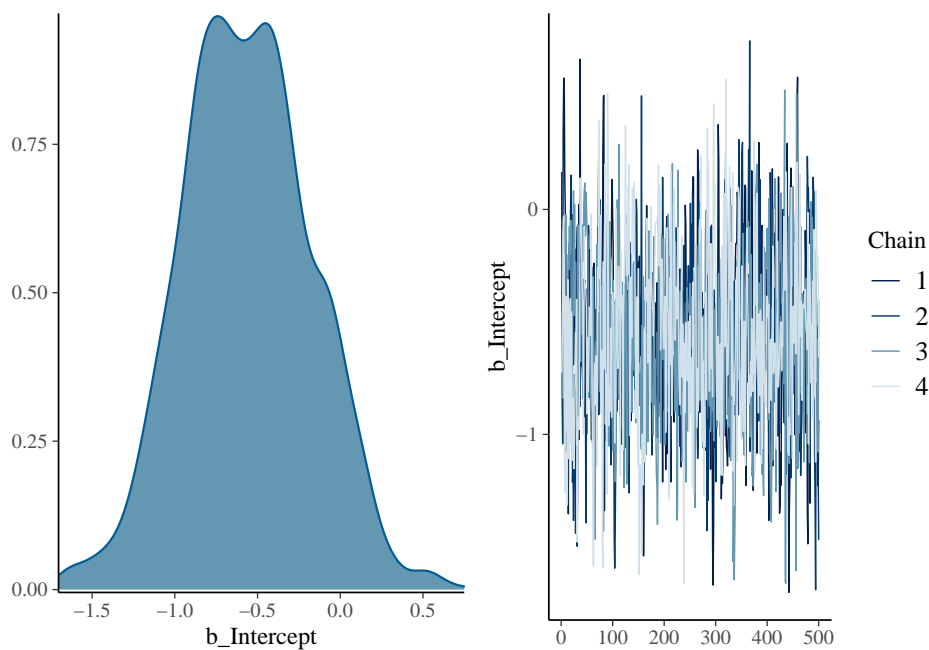

The `summary()` function provides summary statistics for θ.

```
summary(fit)
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: y ~ 0 + Intercept
##    Data: bdata (Number of observations: 20)
## Samples: 4 chains, each with iter = 1000; warmup = 500; thin = 1;
##          total post-warmup samples = 2000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept    -0.56      0.39    -1.29     0.17 1.01      684     1111
##
```

```
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

The `posterior_samples()` function will display the simulated draws of θ.

```
post <- posterior_samples(fit)
head(post)
```

```
##    b_Intercept        lp__
## 1  -0.3644459 -13.59560
## 2  -0.4101895 -13.54451
## 3  -0.5691616 -13.48236
## 4   0.3479299 -16.35609
## 5   0.5788761 -18.04030
## 6   0.5842954 -18.08432
```

2.6   Inferences about the proportion

To obtain a sample of draws from the posterior distribution on p, one can use
the inverse logit transformation on the simulated draws of θ.

$$p = \frac{\exp(\theta)}{1 + \exp(\theta)}$$

```
post %>%
  mutate(p = exp(b_Intercept) / (1 + exp(b_Intercept))) -> post
```

The posterior density for p is found by constructing a density plot of the simu-
lated draws of p.

```
ggplot(post, aes(p)) +
  geom_density()
```
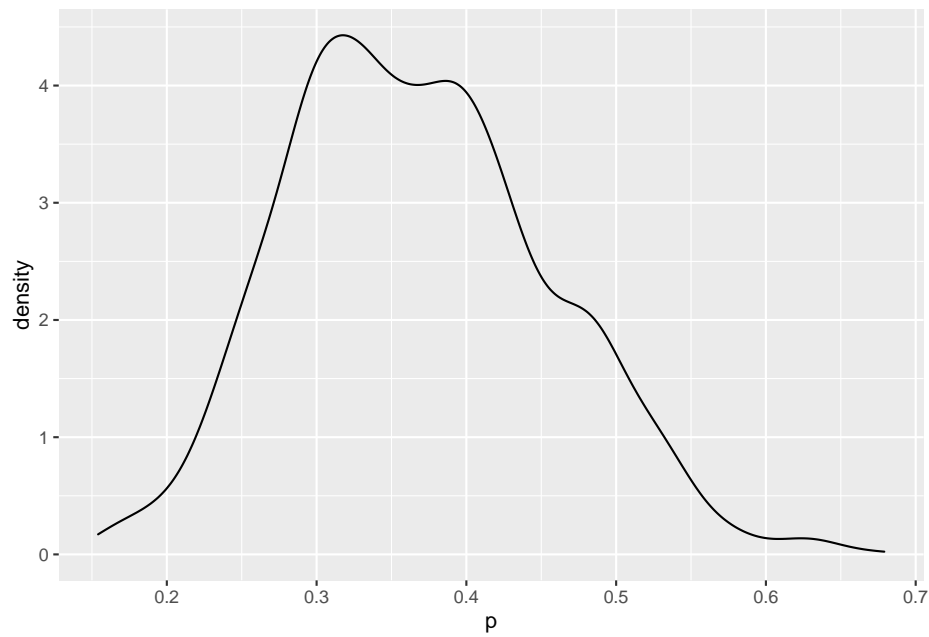
A 90% posterior interval estimate is found by selecting particular quantiles from the simulated values of p.

```
quantile(post$p, c(.05, .95))
```

```
##        5%        95%
## 0.2378037 0.5192776
```

Chapter 3

Normal Modeling

3.1 Packages for example

```r
library(ProbBayes)
library(brms)
```

3.2 Normal sampling model

Assume that $y_1, ..., y_n$ are a sample from a normal distribution with mean μ and standard deviation σ.

For a prior, we assume that μ and σ are independent where μ is assigned a normal prior and σ is assigned a uniform prior on an interval.

3.3 Data and prior

We consider the variable `time` from the dataset `federer_time_to_serve` that contains the time to serve for 20 serves of Roger Federer.

We place a weakly informative prior on the parameters. We assume the mean time-to-serve μ is N(15, 5) and assume the standard deviation σ is uniform on the interval (0, 20).

3.4 Bayesian fitting

We use the `brm()` function with the family = gaussian option. Note how the prior is specified by the prior argument.

```r
fit <- brm(data = federer_time_to_serve,
           family = gaussian,
           time ~ 1,
    prior = c(prior(normal(15, 5), class = Intercept),
              prior(uniform(0, 20), class = sigma)),
    iter = 1000, refresh = 0, chains = 4)
```

```
## Warning: It appears as if you have specified an upper bounded prior on a parameter
## If this is really what you want, please specify argument 'ub' of 'set_prior' appropr
## Warning occurred for prior
## sigma ~ uniform(0, 20)

## Compiling Stan program...

## Start sampling
```

One obtains density plots and trace plots for μ and σ by the `plot()` function.

```r
plot(fit)
```
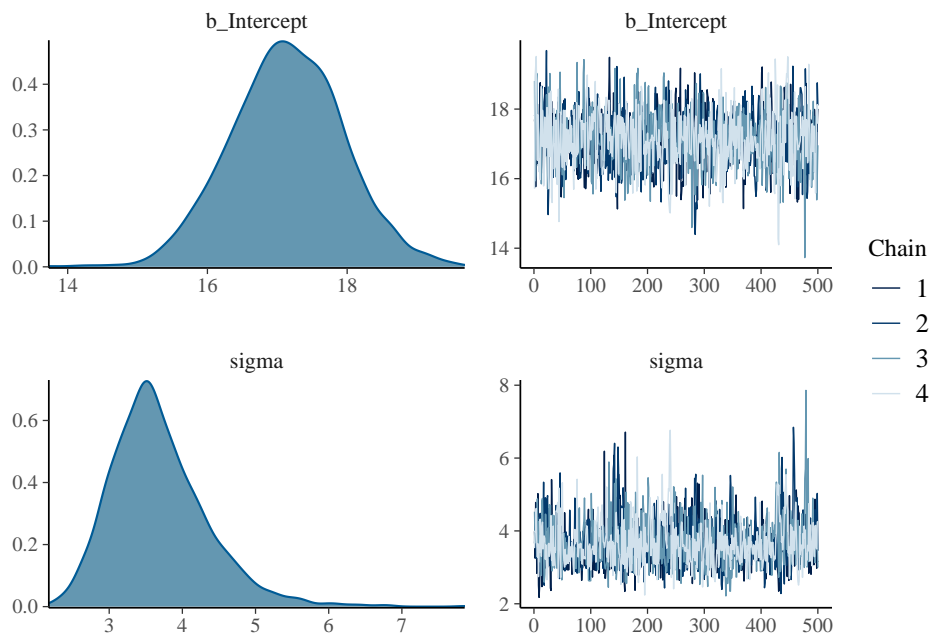

One obtains posterior summaries for each parameter by the `summary()` function.

```r
summary(fit)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: time ~ 1
##    Data: federer_time_to_serve (Number of observations: 20)
```

```
## Samples: 4 chains, each with iter = 1000; warmup = 500; thin = 1;
##          total post-warmup samples = 2000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept    17.14      0.80    15.58    18.70 1.00     1394     1053
##
## Family Specific Parameters:
##        Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma      3.68      0.66     2.64     5.21 1.00     1137      878
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

One can obtain a matrix of simulated draws by the `posterior_samples()` function.

```
post <- posterior_samples(fit)
head(post)
```

```
##   b_Intercept    sigma      lp__
## 1    17.69829 3.989865 -57.47744
## 2    17.47939 3.252316 -57.01475
## 3    15.76746 4.781054 -59.40308
## 4    17.62919 4.469536 -58.14521
## 5    16.42508 2.837153 -58.30991
## 6    18.14501 3.504918 -57.70406
```

Chapter 4

Poisson Modeling

4.1 Packages for example

```r
library(ProbBayes)
library(brms)
```

4.2 Poisson log-linear model

Here we observe counts $y_1, ..., y_n$ distributed according to a Poisson distribution with mean λ.

Write a model in terms of the logarithm of the mean:

$$\theta = \log \lambda$$

Complete the model by assigning a $N(\mu, \sigma)$ prior to the log mean parameter θ.

4.3 Learning about website counts

In the `ProbBayes` package, the variable `Count` in the dataset `web_visits` contains counts of daily visits to a blog website. We are interested in learning about the mean count of visits λ.

We place a N(0, 10) prior on $\theta = \log \lambda$ reflecting weak prior information about the location of this paramter.

4.4   Bayesian Fitting

In this run of the `brm()` function, we assume Poisson sampling and a normal prior with mean 0 and standard deviation 10 placed on the log mean $\theta = \log \lambda$.

```r
fit <- brm(Count ~ 0 + Intercept,
           data = web_visits,
           family = poisson,
            refresh = 0,
          prior = prior(normal(0, 10),
                        class = b,
                        coef = "Intercept"))
```

```
## Compiling Stan program...

## Start sampling
```

We confirm the prior with the `prior_summary()` function.

```r
prior_summary(fit)
```

```
##              prior class       coef group resp dpar nlpar bound
## 1                      b
## 2 normal(0, 10)        b Intercept
```

The `summary()` function provides summaries of the posterior of θ.

```r
summary(fit)
```

```
##  Family: poisson
##    Links: mu = log
## Formula: Count ~ 0 + Intercept
##     Data: web_visits (Number of observations: 28)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     4.59      0.02     4.55     4.62 1.00     1574     2209
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

The `posterior_samples()` function outputs the posterior simulations of θ.

```r
post <- posterior_samples(fit)
head(post)
```

```
##   b_Intercept      lp__
## 1    4.556476 -136.7185
```

```
## 2    4.562960 -136.2752
## 3    4.560979 -136.3987
## 4    4.598670 -135.8759
## 5    4.594855 -135.7515
## 6    4.590376 -135.6570
```

4.5 Posterior predictive model checks

Actual this is a poor model for these data. One can see that by several posterior predictive checks.

The `pp_check()` shows density plots of 10 replicated datasets from the posterior predictive distribution. Note that these replicated datasets look different (smaller variation) than the observed data.

```
pp_check(fit)
```

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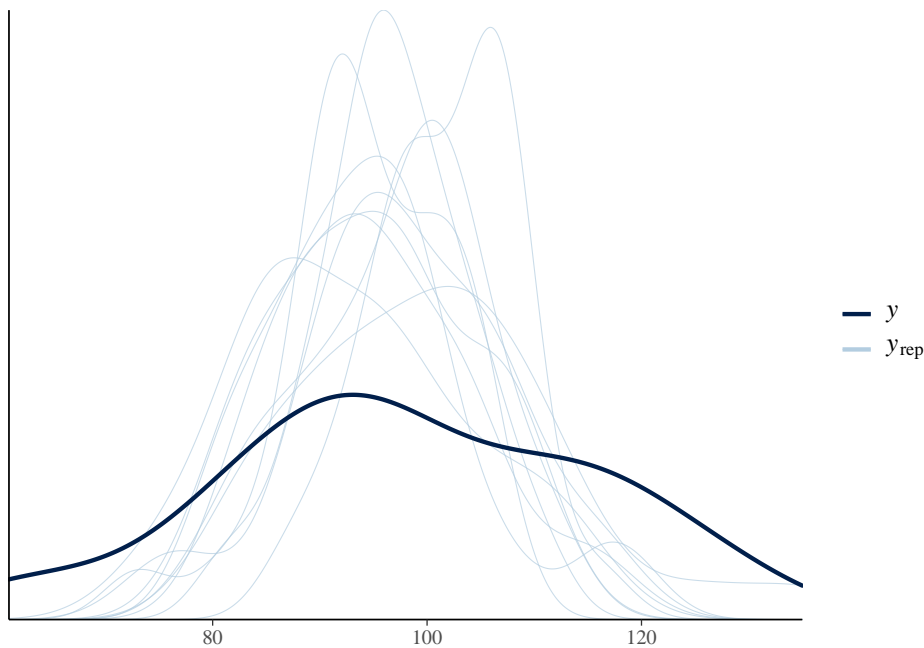

The `pp_check()` function will implement a posterior predictive check using various checking functions. Here we are using $(\bar{y}, s)$ as a bivariate checking function. The scatterplot represents values of $(\bar{y}, s)$ from the simulated predictive distributions and the observed values of $(\bar{y}, s)$ is displayed. The takeaway is that the observed data has more variation than predicted from the Poisson model.

```
pp_check(fit, type = "stat_2d")
```

```
## Using all posterior samples for ppc type 'stat_2d' by default.
```
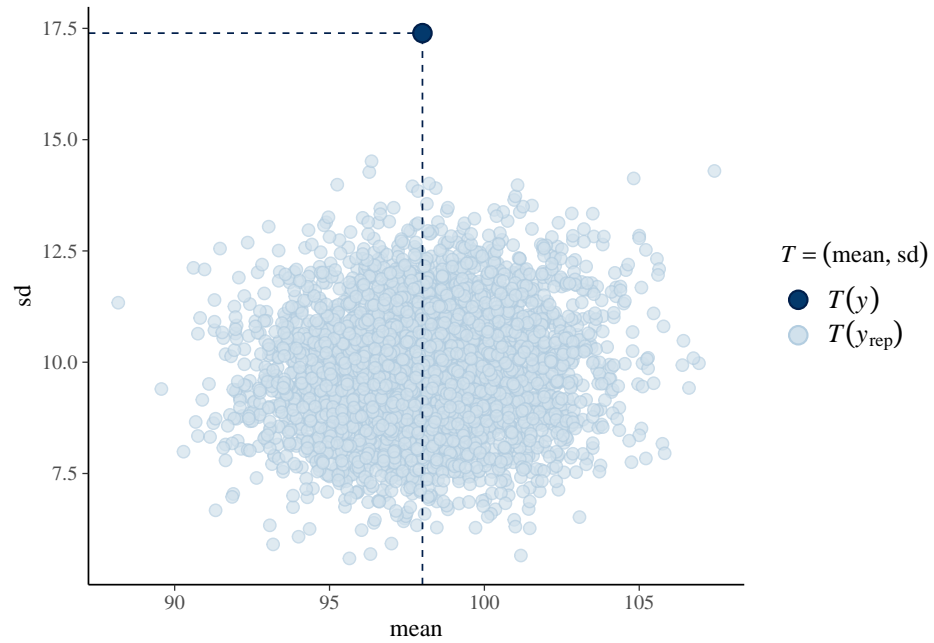
Chapter 5

Comparing Proportions

5.1 Packages for example

```r
library(ProbBayes)
library(brms)
library(dplyr)
library(ggplot2)
```

5.2 Facebook use example

In Chapter 9, we consider the following comparison of proportions example. A sample of students were asked their gender and the average number of times they visited Facebook in a day.

Of n_M males sampled, y_M had a high number of Facebook visits, and of n_F females sampled, y_F had a high number of visits.

Suppose the data is organized as a data frame as follows:

Gender	Sample_size	Visits
male	n_M	y_M
female	n_F	y_F

5.3 Sampling model

Suppose we have two independent samples where y_M is binomial(n_M, p_M) and y_F is binomial(n_F, p_F).

Write the proportions using a logistic model:

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 I(Gender = Male)$$

Note for females, the logit of p_F is given by

$$\log\left(\frac{p}{1-p}\right) = \beta_0$$

and for males the logit for p_M is given by

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1$$

5.4 The data

Here's the observed data:

```
fb_data <- data.frame(Gender = c("male", "female"),
                      Sample_Size = c(93, 151),
                      Visits = c(39, 75))
```

5.5 Priors

In this model, β_0 is the logit of the proportion of women who are high Facebook users and β_1 represents the difference in the logits of the proportions for men and women.

Assume that you don't know much about the location of β_0, but you believe men and women are similar in their use of Facebook. So you assign a N(0, 31.6) prior to β_0 with a high standard deviation, reflecting little knowledge. To reflect the belief that β_1 is close to 0, you use a N(0, 0.71) prior.

The `get_prior()` function lists all parameters to define priors on for this particular model, assigning the result to `prior`. Then the two components of `prior` are assigned that reflect the statements above.

```
(my_prior <- get_prior(family = binomial,
          Visits | trials(Sample_Size) ~ Gender,
          data = fb_data))
```

```
##                    prior     class        coef group resp dpar nlpar bound
## 1                              b
## 2                              b Gendermale
## 3 student_t(3, 0, 2.5) Intercept
```

```
my_prior$prior[3] <- "normal(0, 31.6)"
my_prior$prior[2] <- "normal(0, 0.71)"
```

5.6 Posterior sampling

Here is the run of `brm()` where I use the prior specification in `my_prior`.

```
fit <- brm(family = binomial,
           Visits | trials(Sample_Size) ~ Gender,
           data = fb_data,
           prior = my_prior,
           iter = 1000,
           refresh = 0)
```

```
## Compiling Stan program...
```

```
## Start sampling
```

One obtains the matrix of simulated values of the parameters by the `posterior_samples()` function.

```
post <- posterior_samples(fit)
head(post)
```

```
##     b_Intercept b_Gendermale       lp__
## 1   0.149825204   -0.3929578 -10.89798
## 2  -0.077477237   -0.1550768 -10.36612
## 3   0.014914205   -0.2591160 -10.32287
## 4  -0.244104208   -0.2079800 -11.39275
## 5  -0.006092644   -0.2035093 -10.36111
## 6   0.012292753   -0.4538551 -10.53464
```

The `plot()` function provides trace plots and density plots of each parameter.

```
plot(fit)
```
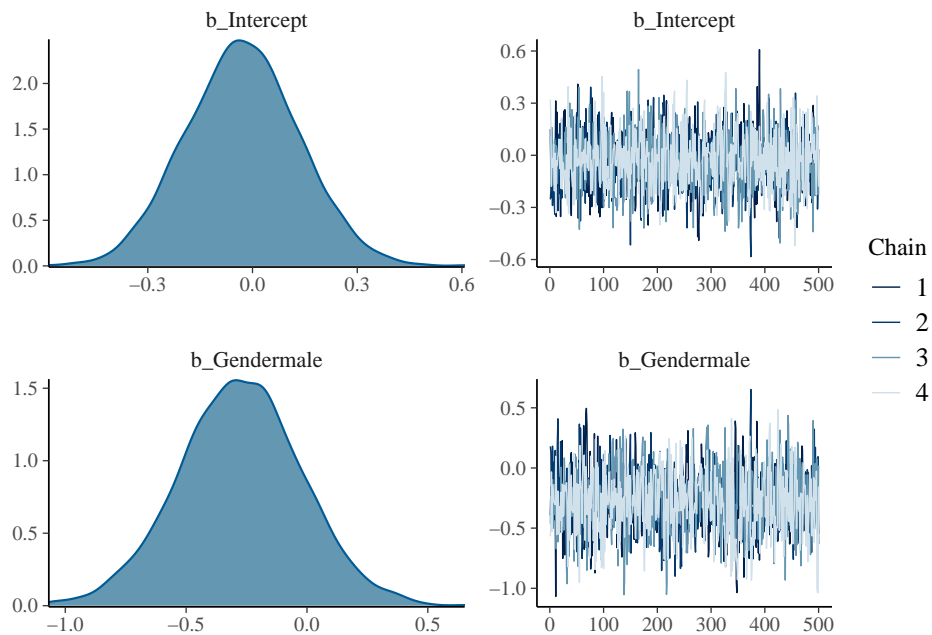
Posterior summaries are provided by the `print()` function.

```
print(fit)
```

```
##  Family: binomial
##   Links: mu = logit
## Formula: Visits | trials(Sample_Size) ~ Gender
##     Data: fb_data (Number of observations: 2)
## Samples: 4 chains, each with iter = 1000; warmup = 500; thin = 1;
##          total post-warmup samples = 2000
##
## Population-Level Effects:
##            Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     -0.03      0.16    -0.34     0.28 1.00     2198     1576
## Gendermale    -0.28      0.25    -0.78     0.23 1.01     1525     1121
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Chapter 6

Comparing Rates

6.1 Packages for example

```
library(ProbBayes)
library(brms)
library(dplyr)
```

6.2 Comparing two Poisson Rates

Suppose we observe two independent samples: $x_1, ..., x_m$ are a random sample from a Poisson distribution with mean λ_x, and $w_1, ..., w_n$ are a random sample from a Poisson distribution with mean λ_y. We are interested in learning about the ratio of Poisson means

$$\theta = \frac{\lambda_x}{\lambda_y}$$

6.3 Write as a log-linear model

Suppose we collect the observations

$$y = c(x_1, ..., x_m, w_1, ..., w_n)$$

and let `group2` be an indicator variable for the second group.

$$group2 = c(0, 0, ..., 0, 1, 1, ..., 1)$$

Then we can represent the model as

$$y_1, ..., y_{m+n}$$

independent from Poisson distributions with means $\lambda_1, ..., \lambda_{m_n}$ where the means follow the log-linear model

$$\log \lambda_j = \beta_0 + \beta_1 group2$$

In this model, $\beta_0 = \log \lambda_x$, and $\beta_0 + \beta_1 = \log \lambda_y$. So $\beta_1 = \log(\lambda_y) - \log(\lambda_x)$ represents the increase in the means on the log scale.

6.4   The data

We collect web count visits for a number of days stored in the data frame `web_visits` in the `ProbBayes` package. The key variables are `Day`, the day of the week, and `Count`, the website visit count. We define a new variable `Type` that is either "weekend" or "weekday".

We are interested in comparing the mean visit counts for weekdays and weekend days.

```r
web_visits %>%
  mutate(Type = ifelse(Day %in%
      c("Fri", "Sat", "Sun"), "weekend", "weekday")) -> web_visits
```

6.5   Priors

Here we are assume weakly informative priors on the regression parameters β_0 and β_1.
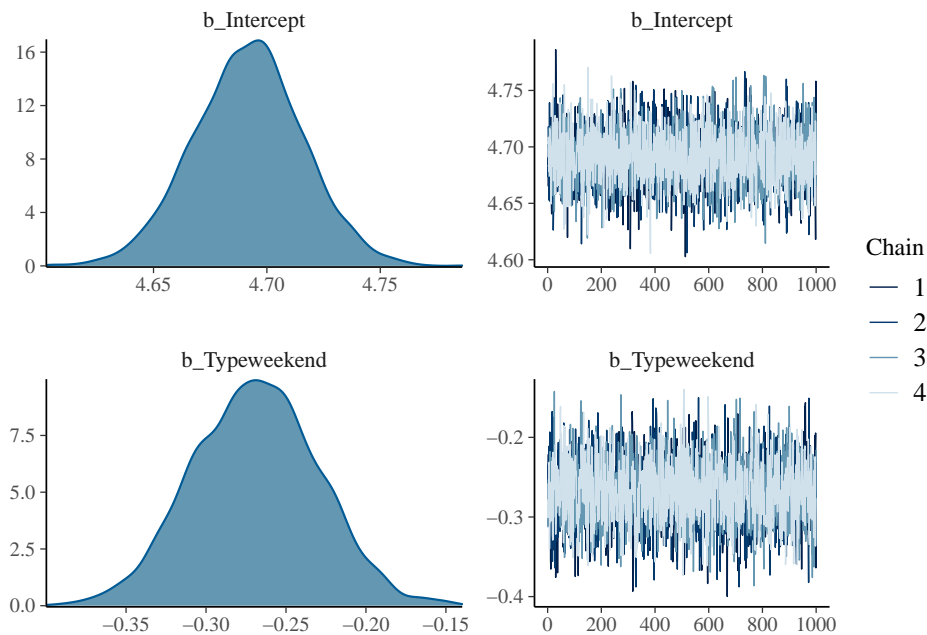
6.6   Bayesian fitting

```r
fit <- brm(Count ~ Type,
           family = poisson,
           data = web_visits,
           refresh = 0)
```

```
## Compiling Stan program...
```

```
## Start sampling
```

```r
plot(fit)
```

```
summary(fit)
```

```
##  Family: poisson
##    Links: mu = log
## Formula: Count ~ Type
##     Data: web_visits (Number of observations: 28)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##             Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept       4.69      0.02     4.64     4.74 1.00     3888     2663
## Typeweekend    -0.27      0.04    -0.35    -0.19 1.00     3292     2805
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
sim_draws <- posterior_samples(fit)
head(sim_draws)
```

```
##   b_Intercept b_Typeweekend        lp__
## 1    4.683972    -0.3003928 -111.6071
## 2    4.690432    -0.2403990 -111.1721
## 3    4.665176    -0.2213930 -111.6114
## 4    4.683088    -0.2696015 -110.9016
```

```
## 5     4.678049     -0.2929583 -111.6562
## 6     4.699903     -0.2566505 -111.0721
```

Chapter 7

Multilevel Modeling of Proportions

7.1 Packages for example

```r
library(ProbBayes)
library(tidyverse)
library(brms)
```

7.2 Hospital Study

Table 10.2 gives the number of cases and number of deaths from heart attacks for 13 hospitals in New York City. This data is contained in the data frame `DeathHeartAttackManhattan` in the `ProbBayes` package.

7.3 A Multilevel Model

We consider a different formulation of the hierarchical model described in Section 10.3.

Sampling

We first assume that y_j, the number of deaths for the jth hospital, is binomial with sample size n_j and probability p_j. Let $\theta_j = \log(p_j/(1 - p_j))$ denote the logit for the jth hospital.

Write $\theta_j = \beta + \gamma_j$.

Prior

1. We assume the intercept β has a student t distribution with mean 0, scale parameter 2.5 and 3 degrees of freedom.

2. We assume $\gamma_1, ..., \gamma_N$ have a normal distribution with mean 0 and standard deviation σ.

3. The standard deviation σ is assumed to have a t density with mean 0 and standard deviation 3.5.

7.4   Fitting the Bayesian model

We fit the multilevel model using the '`brm()` function. Note the use of the "family = binomial" argument to indicate the sampling distribution. The "(1 | Hospital)" component indicates that the γ_j have a random distribution.

```
fit <- brm(data = DeathHeartAttackManhattan,
           family = binomial,
           Deaths | trials(Cases) ~ 1 + (1 | Hospital),
           refresh = 0)
```

```
## Compiling Stan program...
```

```
## Start sampling
```

We didn't specify priors, but there are default priors behind the scenes. The `prior_summary()` function displays the priors.

```
prior_summary(fit)
```

```
##                   prior     class      coef     group resp dpar nlpar bound
## 1 student_t(3, 0, 2.5) Intercept
## 2 student_t(3, 0, 2.5)          sd
## 3                              sd            Hospital
## 4                              sd Intercept Hospital
```

7.5   Posterior summaries of β and σ

The `summary()` function shows posterior summaries of β (the intercept) and the standard deviation σ.

```
summary(fit)
```

```
##  Family: binomial
##   Links: mu = logit
## Formula: Deaths | trials(Cases) ~ 1 + (1 | Hospital)
##    Data: DeathHeartAttackManhattan (Number of observations: 13)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
```

```
## Group-Level Effects:
## ~Hospital (Number of levels: 13)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)     0.19      0.15     0.01     0.56 1.00      913     1675
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept    -2.60      0.11    -2.82    -2.37 1.00     2555     1376
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

7.6 Posterior summaries of hospital effects

The `posterior_samples()` function produces a large matrix of simulated draws where the column corresponds to the parameter and the row corresponds to the iteration number.

By use of the `pivot_longer()` function, I reformat the simulation matrix where there is a new variable `Hospital` indicating the name of the hospital and `Effect` is the simulated value of γ_j. Also I create a new variable that is the number of the hospital from 1 to 13.

```
posterior_samples(fit) %>%
  pivot_longer(starts_with("r_Hospital"),
               names_to = "Hospital",
               values_to = "Effect") -> post
post$Hospital_No <- as.character(as.numeric(factor(post$Hospital)))
```

Below is a graph of the posterior distribution of the parameters $\{\beta + \gamma_j\}$ for all 13 hospitals.

These are graphed on the logit scale. By taking the inverse logit function, one could find the posterior distributions of the death rates $p_1, ..., p_N$.

```
ggplot(post, aes(Hospital_No, Effect + b_Intercept)) +
  geom_boxplot() +
  coord_flip()
```
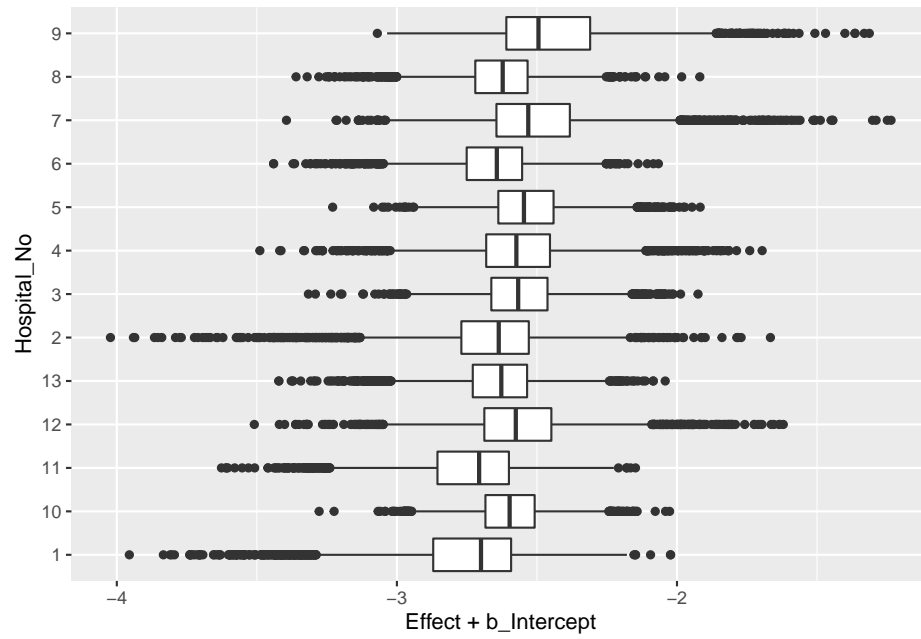
Chapter 8

Multilevel Modeling of Means

8.1 Packages for example

```r
library(ProbBayes)
library(tidyverse)
library(brms)
```

8.2 Movie Ratings Study

Table 10.1 gives summaries of the ratings for eight different animation movies. The table includes the number of ratings, the mean and the standard deviation of the ratings. The data is contained in the data frame `animation_ratings` in the `ProbBayes` package.

8.3 The Multilevel Model

Sampling

Let y_{ij} denote the rating of the ith individual for the jth movie.

We assume that $y_{ij} \sim N(\mu_j, \sigma)$.

Prior

The parameters $\mu_1, ..., \mu_8$ represent the mean ratings for the eight movies. Write

$$\mu_j = \beta + \gamma_j$$

1. The intercept parameter β has a student t distribution with mean 4, scale parameter 2.5, and 3 degrees of freedom.

2. We assume the effect parameters $\gamma_1, ..., \gamma_8$ have a normal distribution with mean 0 and standard deviation τ.

3. There are two standard deviations, the sampling standard deviation σ and the between-means standard deviation τ. Each of these standard deviations are given weakly informative student t distributions with mean 0, scale 2.5 and 3 degrees of freedom.

8.4　Bayesian Fitting

The model is fit by use of the `brm()` function. By default, this function assumes a Gaussian (normal) sampling distribution. The "(1 | movieID)" argument indicates that the $\mu_1, ..., \mu_8$ have a random distribution.

```r
fit <- brm(rating ~ (1 | movieId),
           data = animation_ratings,
           refresh = 0)
```

```
## Compiling Stan program...
```

```
## Start sampling
```

```
## Warning: There were 6 divergent transitions after warmup. Increasing adapt_delta ab
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

One can check the default priors by use of the `prior_summary()` function.

```r
prior_summary(fit)
```

```
##                    prior     class      coef    group resp dpar nlpar bound
## 1 student_t(3, 4, 2.5) Intercept
## 2 student_t(3, 0, 2.5)          sd
## 3                              sd            movieId
## 4                              sd Intercept movieId
## 5 student_t(3, 0, 2.5)       sigma
```

The posterior matrix of simulated draws is available by use of the `posterior_samples()` function. Below I construct density estimates of the two standard deviation parameters σ (blue) and τ (red).

```r
ggplot(posterior_samples(fit),
       aes(sigma)) +
  geom_density(color = "blue") +
  geom_density(aes(sd_movieId__Intercept),
               color = "red")
```
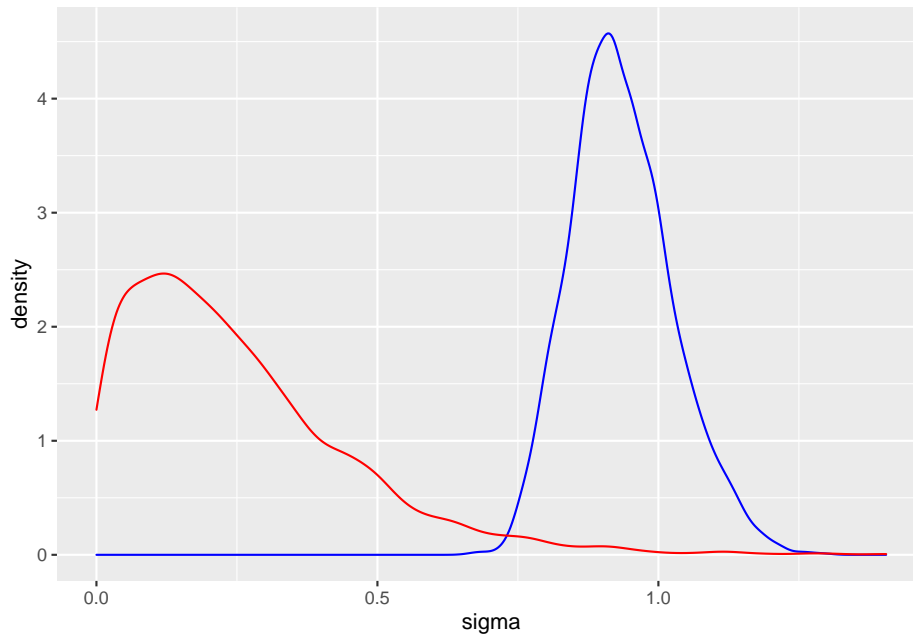
To show the posterior distributions of the means, I reshape the matrix of simulated draws by use of the `pivot_longer()` function.

```
posterior_samples(fit) %>%
  pivot_longer('r_movieId[76093,Intercept]':'r_movieId[81847,Intercept]',
               names_to = "Movie",
               values_to = "Effect") -> post
```

Remember that we represented the movie ratings mean as $\mu_j = \beta + \gamma_j$. Below are parallel boxplots of the posterior distributions of $\mu_1, ..., \mu_8$.

```
ggplot(post, aes(Movie, Effect + b_Intercept)) +
  geom_boxplot() +
  coord_flip()
```
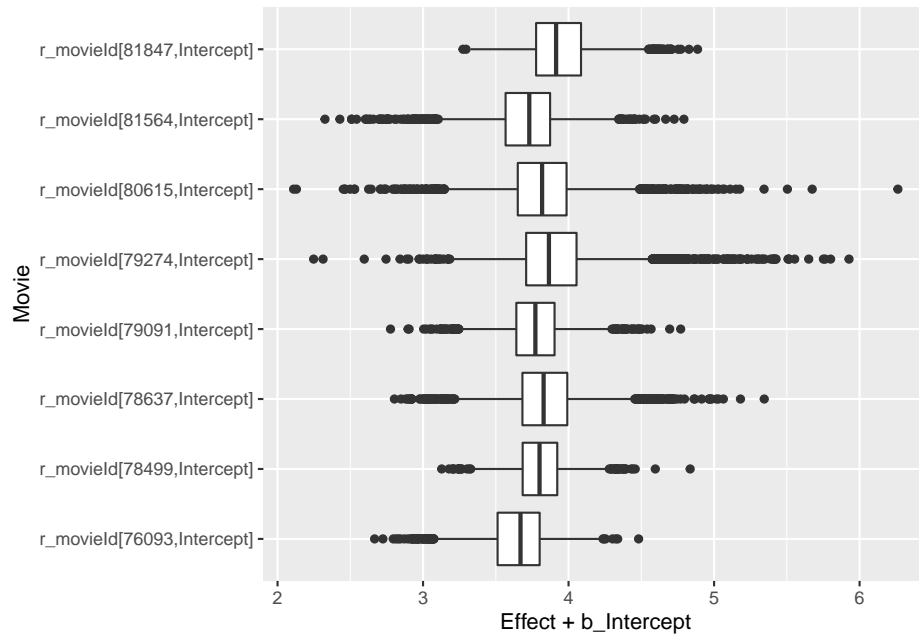
Chapter 9

Multiple Regression and Logistic Models

9.1 Load Packages

```
library(ProbBayes)
library(brms)
library(dplyr)
library(ggplot2)
```

9.2 Multiple regression example

Exercise 1 in Chapter 12 describes a dataset that gives the winning time in seconds for the men's and women's 100 m butterfly race for the Olympics for the years 1964 through 2016. This data is available as the data frame `olympic_butterfly` in the `ProbBayes` package.

```
head(olympic_butterfly)
```

```
##   Year Gender  Time
## 1 2016  Women 55.48
## 2 2012  Women 55.98
## 3 2008  Women 56.73
## 4 2004  Women 57.72
## 5 2000  Women 56.61
## 6 1996  Women 59.13
```

Create a new variable `year_64` which is equal to the number of years after 1964. Also define a `gender` variable that is 1 (0) if the race is for women (men).

```r
olympic_butterfly %>%
  mutate(year_64 = Year - 1964,
         gender = ifelse(Gender == "Women",
                         1, 0)) %>%
  filter(is.na(Time) == FALSE) ->
  olympic_butterfly2
```

9.3   The model

Let y_j denote the winning time in seconds for the jth race. We assume that y_j is normal(μ_j) where the means satisfy the regression model

$$\mu_j = \beta_0 + \beta_1 x_{1j} + \beta_2 x_{2j},$$

where x_{1j} and x_{2j} are respectively the years after 1964 and the indicator for gender.

We will assume a weakly informative prior, where each of β_0, β_1, β_2 and σ are given flat distributions with high variances.

9.4   Fitting the model

The `brm()` function resembles the syntax of the popular `lm()` function, but this is implementing a Stan fit.
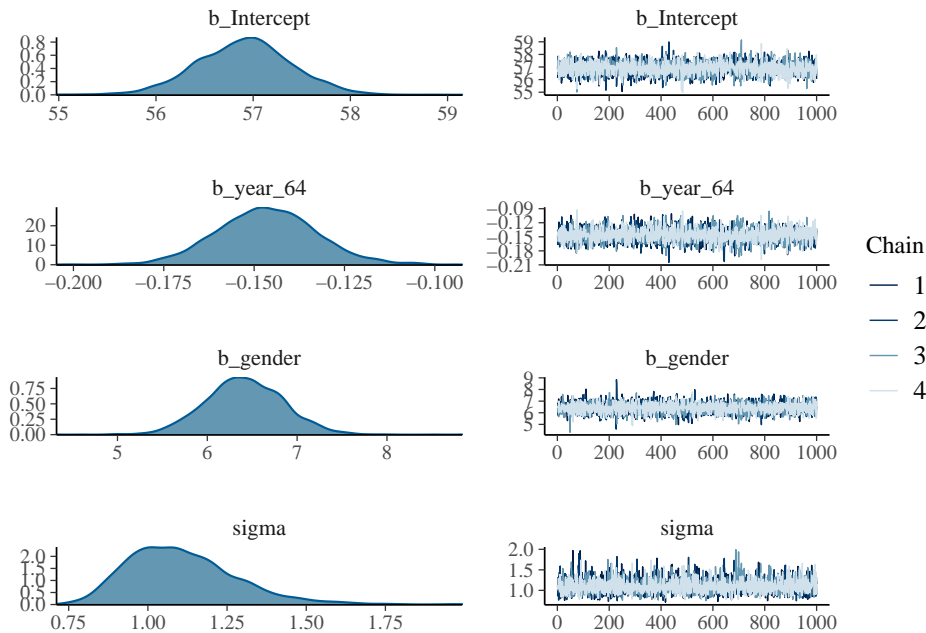
```r
fit <- brm(Time ~ year_64 + gender,
           data = olympic_butterfly2,
           refresh = 0)
```

```
## Compiling Stan program...
```

```
## Start sampling
```

The `plot()` function will plot trace and density graphs for each parameter.

```r
plot(fit)
```

The `summary()` function provides posterior summaries for each parameter.

```
summary(fit)
```

```
##  Family: gaussian
##    Links: mu = identity; sigma = identity
## Formula: Time ~ year_64 + gender
##     Data: olympic_butterfly2 (Number of observations: 27)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept    56.91      0.49    55.95    57.85 1.00     3758     2788
## year_64      -0.15      0.01    -0.17    -0.12 1.00     3977     2958
## gender        6.42      0.43     5.58     7.25 1.00     4018     2651
##
## Family Specific Parameters:
##        Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma      1.11      0.17     0.84     1.50 1.00     3393     2634
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

The `posterior_samples()` function outputs a matrix of simulated draws of $(\beta_0, \beta_1, \beta_2, \sigma)$.

```r
post <- posterior_samples(fit)
```

To illustrate an inference problem, suppose one is interested in estimating the mean winning time of the men's race in 1972 which is the function

$$h(\beta) = \beta_0 + 8\beta_1$$

Below we compute the function $h(\beta)$ on the simulated draws and draw a posterior density estimate.

```r
post %>%
  mutate(mean_time_1972 =
            b_Intercept + 8 * b_year_64) ->
  post
```

```r
ggplot(post, aes(mean_time_1972)) +
  geom_density()
```
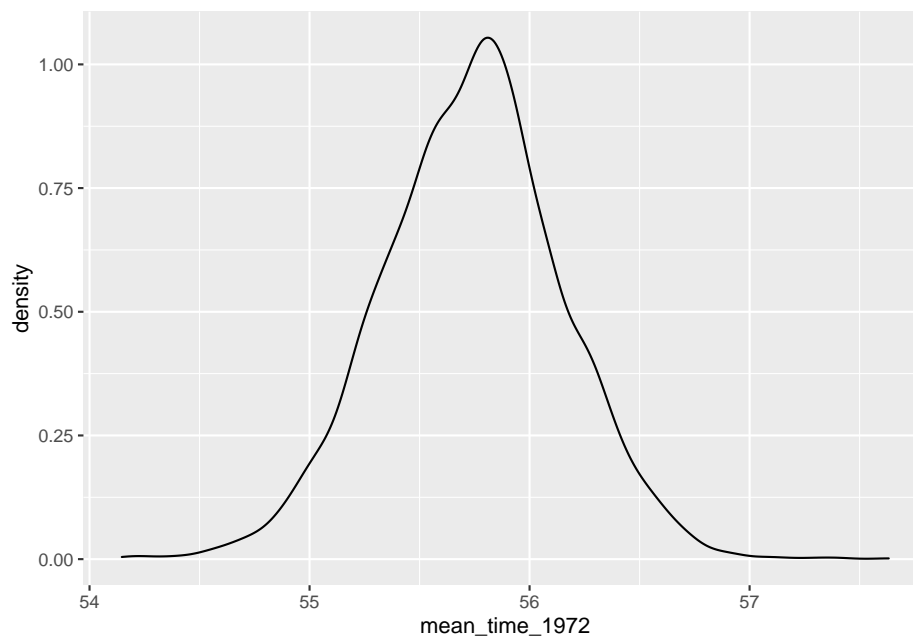

9.5   Logistic example

Exercise 8 of Chapter 12 describes a study where data was collected on some graduate student admission cases. The data is available as the data frame `GradSchoolAdmission` in the `ProbBayes` package. The variables include y, a binary variable indicating admission and x_1 and x_2, the GRE score and GPA for the student.

```
head(GradSchoolAdmission)
```

```
##   Admission GRE  GPA
## 1         0 380 3.61
## 2         1 660 3.67
## 3         1 800 4.00
## 4         1 640 3.19
## 5         0 520 2.93
## 6         1 760 3.00
```

9.6   The model

Let $p_i = P(y_i = 1)$ denote the probability of admission for the ith student. We consider the logistic model

$$\log\left(\frac{p_i}{1 - p_i}\right) = \beta_0 + \beta_1 x_{1j} + \beta_2 x_{2j},$$

We assume flat, weakly informative priors for $\beta_0, \beta_1, \beta_2$.

9.7   Fitting the model

We use the `brm()` function for the Bayesian fitting of this model by Stan-MCMC where the syntax is similar to the `glm()` function for a traditional fit.
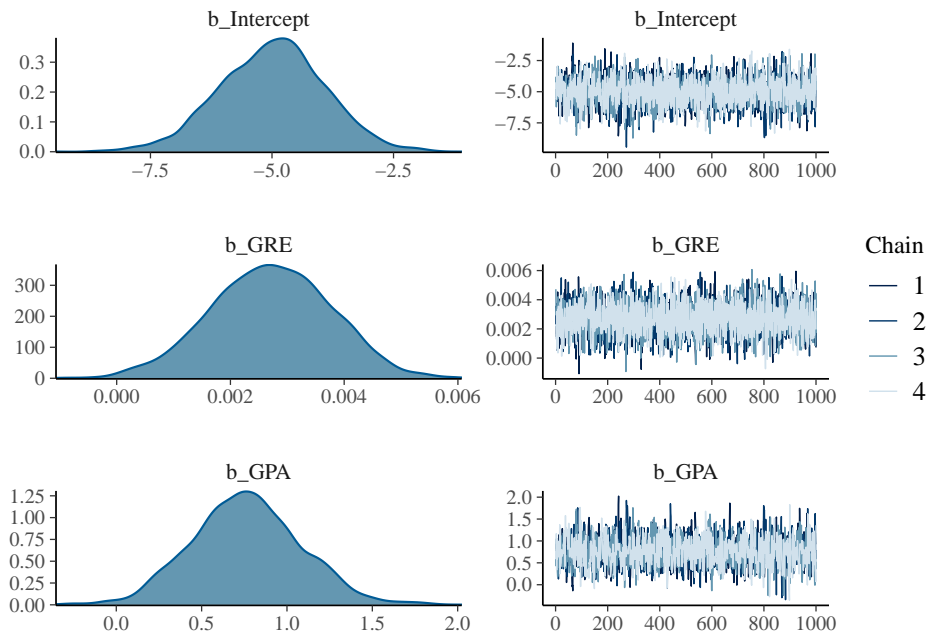
```
fit <- brm(Admission ~ GRE + GPA,
           data = GradSchoolAdmission,
           family = bernoulli(),
            refresh = 0)
```

```
## Compiling Stan program...
```

```
## Start sampling
```

We display trace and density estimate graphs for each regression parameter.

```
plot(fit)
```


We summarize the marginal posterior distributions for each parameter.

```
print(fit)
```

```
##  Family: bernoulli
##    Links: mu = logit
## Formula: Admission ~ GRE + GPA
##     Data: GradSchoolAdmission (Number of observations: 400)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##            Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     -5.00      1.08    -7.21    -2.92 1.00     2205     1937
## GRE            0.00      0.00     0.00     0.00 1.00     4208     3302
## GPA            0.76      0.32     0.17     1.40 1.00     1843     1661
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

We use the `posterior_samples()` function to obtain the matrix of simulated draws from the joint posterior.

```
post <- posterior_samples(fit)
```

Suppose we are interested in estimating the probability of admission for two students with the following covariate values

- GRE = 550, GPA = 3.5
- GRE = 500, GPA = 4.0

Below we first define a function that computes the inverse logit of a value. By using this inverse logit function, we compute the probability of admission for each of these two students.

```
invlogit <- function(x){
  exp(x) / (1 + exp(x))
}
post %>%
  mutate(p_550_3.5 = invlogit(b_Intercept +
                b_GRE * 550 + b_GPA * 3.5),
         p_550_4.0 = invlogit(b_Intercept +
                b_GRE * 550 + b_GPA * 4.0)) ->
  post
```

We display density estimates for the probability of admission for these two students where the blue curve corresponds to the student where GRE = 550 and GPA = 3.5 and the red curve corresponds to the student where GRE = 550 and GPA = 4.0.

```
ggplot(post, aes(p_550_3.5)) +
  geom_density(color = "blue") +
  geom_density(aes(p_550_4.0),
               color = "red")
```
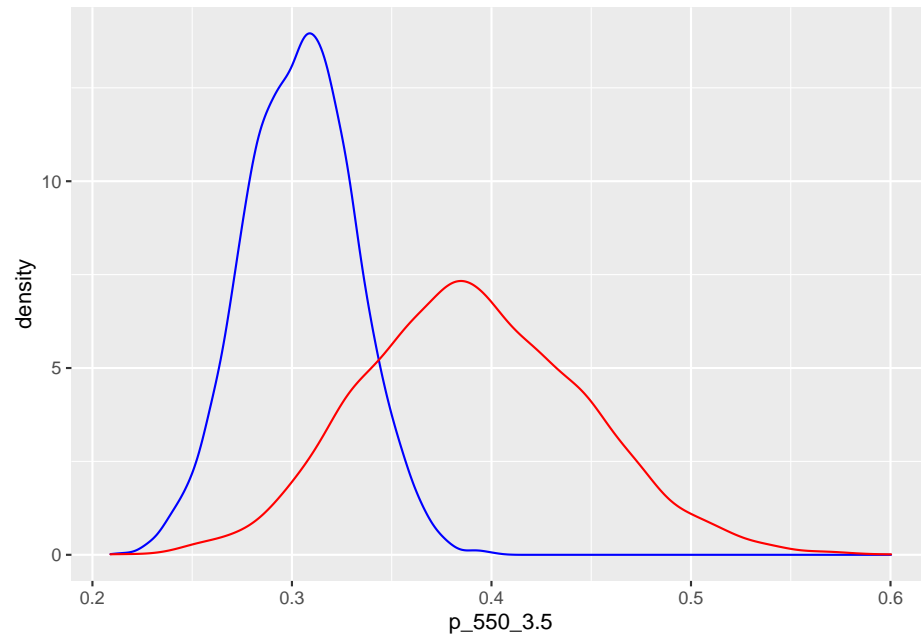
Chapter 10

Federalist Paper Study

10.1 Packages for this example

```
library(tidyverse)
library(brms)
library(bayesplot)
library(ProbBayes)
```

10.2 Federalist paper data

The data frame `federalist_word_study` contains frequency use of words for Federalist Papers written by either Alexander Hamilton or James Madison.

We'll focus on the frequencies of the word "can" in groups of 1000 words written by Hamilton

```
federalist_word_study %>%
  filter(word == "can",
         Authorship == "Hamilton") -> d
head(d)
```

```
##                   Name Total word  N        Rate Authorship Disputed
## 65     Federalist No. 1  1622  can  3 0.0018495684   Hamilton       no
## 1526 Federalist No. 11  2511  can  5 0.0019912386   Hamilton       no
## 2437 Federalist No. 12  2171  can  2 0.0009212345   Hamilton       no
## 3125 Federalist No. 13   970  can  4 0.0041237113   Hamilton       no
## 4256 Federalist No. 15  3095  can 14 0.0045234249   Hamilton       no
## 5530 Federalist No. 16  2047  can  1 0.0004885198   Hamilton       no
```

10.3   The Poisson sampling model

If y_i represents the count of "can" in the i group of words, we assume

$$y_i \sim Poisson(n_i \lambda/1000), i = 1, ..., N$$

where λ is the true rate of the word among 1000 words.

On log scale, the Poisson mean can be written

$$\log E(y_i) = \log \lambda + \log(n_i/1000)$$

which can be fit as a generalized linear model with Poisson sampling, log link, intercept model with an offset of $\log(n_i/1000)$.

We complete this model by assigning the prior

$$\log \lambda \sim N(0, 2)$$

10.4   Fitting the model

We use the 'brm() function with "family = poisson", specifying the offset "N", and specifying the prior by use of the "prior" argument.

```
fit <- brm(data = d, family = poisson,
    N ~ offset(log(Total / 1000)) + 1,
    prior = c(prior(normal(0, 2),
                    class = Intercept)),
            refresh = 0
    )
```

```
## Compiling Stan program...
```

```
## Start sampling
```

We display summaries of the posterior for λ.

```
summary(fit)
```

```
##  Family: poisson
##   Links: mu = log
## Formula: N ~ offset(log(Total/1000)) + 1
##    Data: d (Number of observations: 49)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     0.97      0.06     0.86     1.09 1.00     1602     1974
```

```
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

We save `post` as a matrix of simulated draws.

```
post <- posterior_samples(fit)
```

The function `mcmc_areas()` displays a density estimate of the simulated draws and shows the location of a 50% probability interval.

```
mcmc_areas(post, pars = "b_Intercept")
```
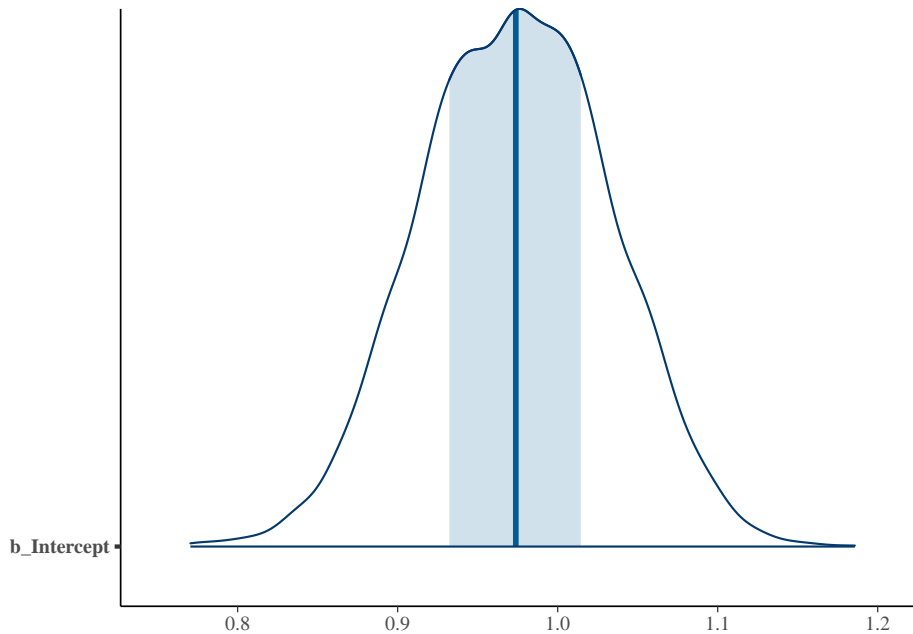

10.5 Model checking

To check if the Poisson sampling model is appropriate we illustrate several posterior predictive checks.

Here we display density estimates for 10 replicated samples from the posterior predictive distribution of y and overlay the observed values as a dark line.

```
pp_check(fit)
```

```
## Using 10 posterior samples for ppc type 'dens_overlay' by default.
```
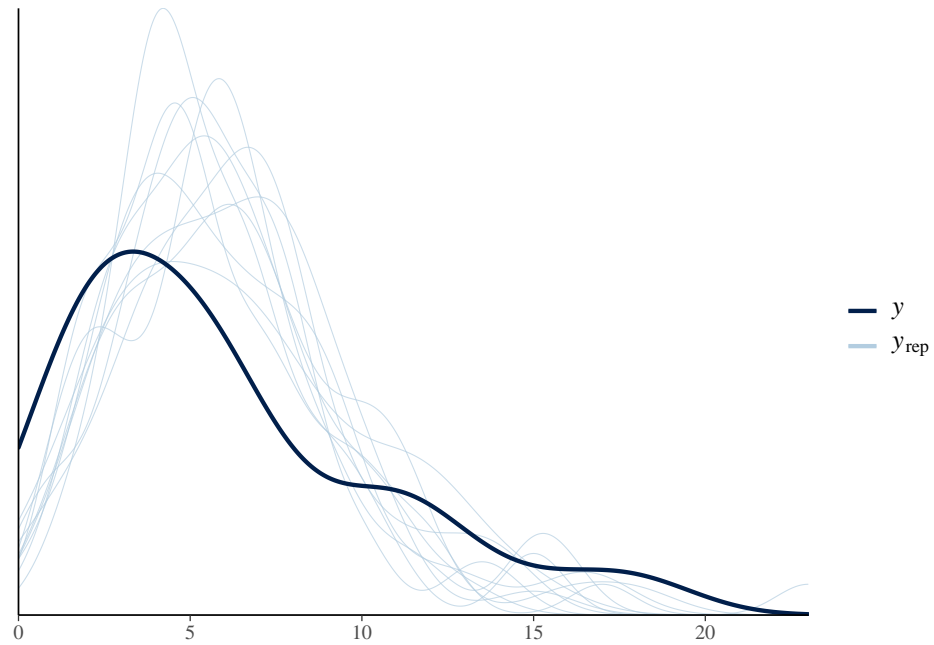
Here we use $(\bar{y}, s_y)$ as a checking function. The scatterplot represents values of $(\bar{y}, s_y)$ from the posterior predictive distribution of replicated data, and the observed value of $(\bar{y}, s_y)$ is shown as a dot.

```
pp_check(fit, type = "stat_2d")
```

```
## Using all posterior samples for ppc type 'stat_2d' by default.
```

The takeaway is that the observed data shows more variability than predicted from the Poisson sampling model.
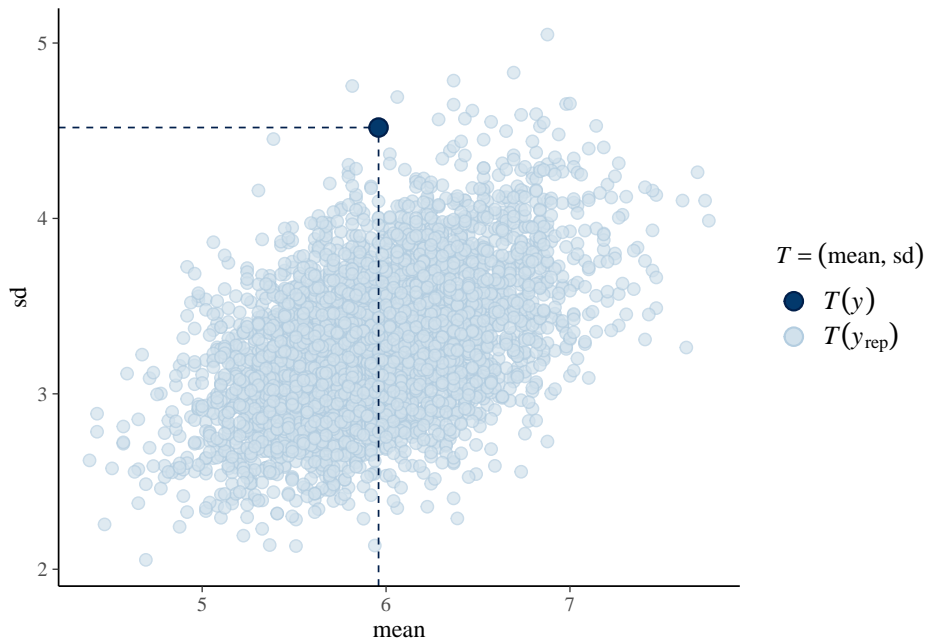
10.6 Negative binomial sampling

One way to handle the extra variability is to assume that the y_i have a negative binomial distribution. (See the text for details.)

Here we outline the code for fitting this model.

We fit the model with the `brm()` function with the "family = negbinomial" option.

```
fit_nb <- brm(data = d, family = negbinomial,
          N ~ offset(log(Total / 1000)) + 1,
          refresh = 0)
```

Compiling Stan program...

Start sampling

Here I can checking on the default priors used by brm:

```
prior_summary(fit_nb)
```

```
##                      prior     class coef group resp dpar nlpar bound
## 1 student_t(3, 1.6, 2.5) Intercept
## 2      gamma(0.01, 0.01)     shape
```

Here are the posterior summaries.

```
summary(fit_nb)
```

```
##  Family: negbinomial
##   Links: mu = log; shape = identity
## Formula: N ~ offset(log(Total/1000)) + 1
##    Data: d (Number of observations: 49)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     1.00      0.10     0.81     1.19 1.00     3457     2694
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## shape     3.28      1.10     1.75     5.97 1.00     3450     2535
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

I save the posterior samples in the data frame `post`.

```
post <- posterior_samples(fit)
```

I try the same posterior predictive checks as before. The message is that the negative binomial sampling model is a better fit to these data.

```
pp_check(fit_nb)
```

```
## Using 10 posterior samples for ppc type 'dens_overlay' by default.
```
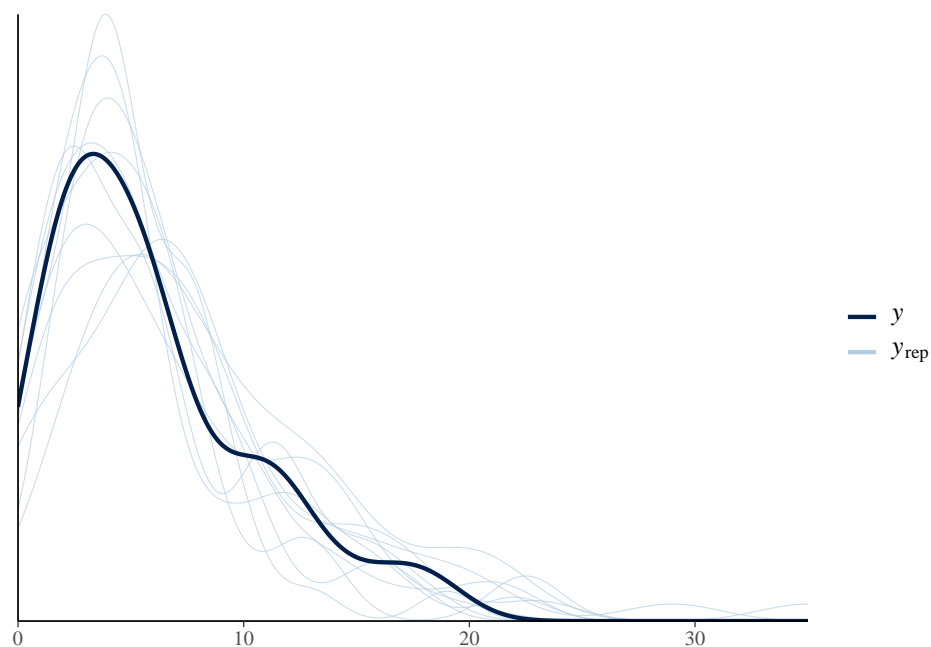
```r
pp_check(fit_nb, type = "stat_2d")
```

```
## Using all posterior samples for ppc type 'stat_2d' by default.
```
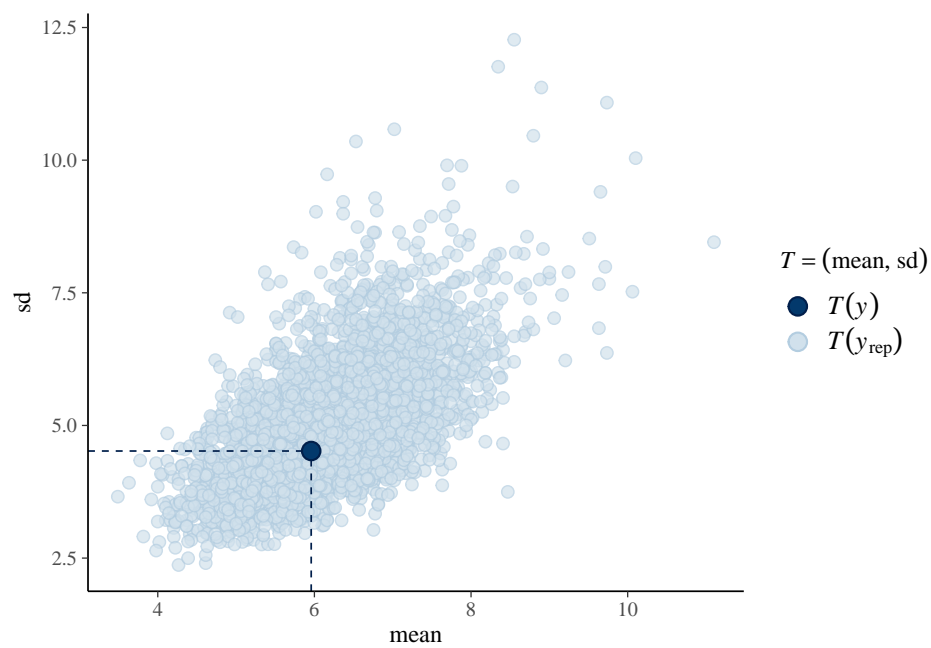
10.7   Comparing use of a word

Next we compare Madison and Hamilton use of the word "can". The data frame
`d2` contains only the word data for the essays that were known to be written by
Hamilton or Madison.

```r
federalist_word_study %>%
  filter(word == "can",
         Authorship %in% c("Hamilton", "Madison")) -> d2
```

Here I fit a regression model for the mean use of "can", where the one predictor
is the categorical variable "Authorship".

```r
fit_nb <- brm(data = d2, family = negbinomial,
           N ~ offset(log(Total / 1000)) +
         Authorship ,
          refresh = 0)
```

```
## Compiling Stan program...
```

```
## Start sampling
```

By summarizing the fit, we can see if the two authors differ in their use of the
word "can" in their writings.

```r
summary(fit_nb)
```

```
##  Family: negbinomial
##   Links: mu = log; shape = identity
## Formula: N ~ offset(log(Total/1000)) + Authorship
##    Data: d2 (Number of observations: 74)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept             1.00      0.09     0.82     1.19 1.00     3486     2799
## AuthorshipMadison    -0.08      0.16    -0.40     0.25 1.00     3265     2711
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## shape     3.84      1.12     2.23     6.48 1.00     3497     2824
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Chapter 11

Multilevel Regression

11.1 Packages for example

```r
library(tidyverse)
library(brms)
```

11.2 Some baseball data

The function `get_onbase_data()` function collects on-base data for all players born in the year 1977 who have had at least 1000 career plate appearances.

```r
source("get_onbase_data.R")
d78 <- get_onbase_data(1977, 1000)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## `summarise()` regrouping output by 'playerID' (override with `.groups` argument)
```

```r
unique(d78$nameLast)
```

```
##  [1] "Beltran"    "Bergeron"   "Bigbie"     "Bloomquist" "Byrd"
##  [6] "Caruso"     "Chavez"     "Davis"      "Ellis"      "Everett"
## [11] "Fukudome"   "Furcal"     "Gerut"      "Gibbons"    "Gonzalez"
## [16] "Hafner"     "Hinske"     "Hudson"     "Inge"       "Jimenez"
## [21] "Jones"      "Monroe"     "Munson"     "Nieves"     "Overbay"
## [26] "Pierre"     "Punto"      "Quinlan"    "Redman"     "Roberts"
## [31] "Ross"       "Rowand"     "Sanchez"    "Thames"     "Tyner"
## [36] "Wigginton"  "Wilkerson"  "Wilson"
```

11.3 Quadratic aging model

Let y_{ij} denote the number of on-base events in n_{ij} opportunities (plate appearances) of the ith batter in the jth season. Assume that y_{ij} is binomial with sample size n_{ij} and probability of success p_{ij}.

Assume that the on-base probabilities for the ith player satisfy the logistic model

$$\log\left(\frac{p_{ij}}{1-p_{ij}}\right) = \beta_{i0} + \beta_{i1}D_{ij} + \beta_{i2}D_{ij}^2$$

where $D_{ij} = x_{ij} - 30$, x_{ij} is the age of the ith player in the jth season.

11.4 Multilevel Prior

The ith player's trajectory is described by the regression vector $\beta_i = (\beta_{i0}, \beta_{i1}, \beta_{i2})$. We place a two-stage prior on the trajectories $\beta_1, ..., \beta_N$:

1. $\beta_1, ..., \beta_N$ are a sample from a multivariate normal density with mean β and variance-covariance matrix Σ.

2. The second-stage parameters β and Σ are independent with weakly informative priors.

11.5 Bayesian fitting

The fitting of this model is done using the `brm()` function.

```
fit <- brm(OB | trials(PA) ~ AgeD + I(AgeD ^ 2) +
             (AgeD + I(AgeD ^  2) | Player),
           data = filter(d78, PA > 0),
           family = binomial("logit"),
           refresh = 0)
```

```
## Compiling Stan program...
```

```
## Start sampling
```

I find posterior means of the fitted trajectories for all players.

```
Player_Fits <- coef(fit)$Player[, "Estimate", ] %>%
  as.data.frame() %>%
    mutate(Player = 1:max(d78$Player))
```

```
d78 <- inner_join(d78, Player_Fits, by = "Player")
```

```
d78 %>%
  mutate(PH = plogis(Intercept + AgeD.y * AgeD.x +
                       IAgeDE2 * AgeD.x ^ 2)) -> d78
```

```
ggplot(d78, aes(Age, PH, group = Player)) +
  geom_line(color = "red", alpha = 0.7) +
  ggtitle("Multilevel Fits")
```


Multilevel Fits
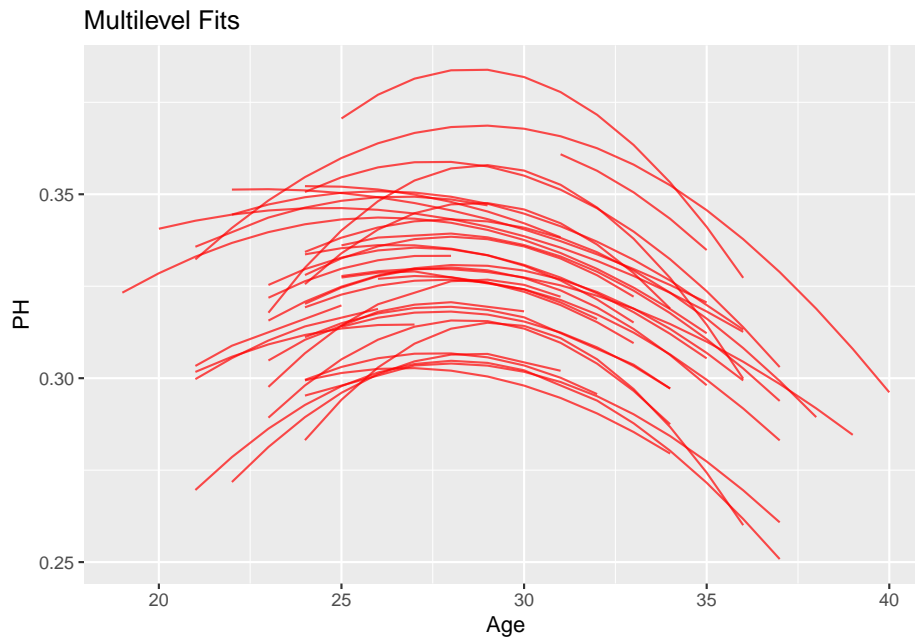
For a given player, define the peak age

$$Age_j = 30 - \frac{\beta_{j1}}{2\beta_{j2}}.$$

the age at which the player achieves peak performance.

The following graph shows the posterior distributions of the peak ages for all players.

```
d78 %>% group_by(Player) %>%
  summarize(b0 = first(Intercept),
            b1 = first(AgeD.y),
            b2 = first(IAgeDE2)) %>%
  mutate(MLM_Peak_Age = 30 - b1 / 2 / b2)  %>%
  ggplot(aes(MLM_Peak_Age)) +
  geom_histogram(bins = 12,
                 color = "white", fill = "tan") +
  xlim(20, 35)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```