Bayesian Statistics

Zack Treisman

WESTERN
COLORADO UNIVERSITY

MATHEMATICS & COMPUTER
SCIENCE DEPARTMENT

Spring 2021

Philosophy

Bayesian statistics is based on a fairly simple procedure, not dissimilar to what is done in the non-Bayesian scenario.

- Propose a form for a model. (For example, define a deterministic function and a stochastic error distribution.)
- Set probability distributions of parameters of the model based on *prior* information. (This is the controversial part.)
- Update the parameter distributions based on data. (This is the part that can be computationally challenging.)

The initial part of a Bayesian analysis is exactly the same as a frequentist analysis: Explore the data graphically and numerically, and come up with appropriate forms for models.

Bayes' Rule

Supposing a model with parameters θ, and given observed data x, compute the probability distribution for θ conditioned on the observations x using Bayes' Rule.

$$P(\theta|x) = \frac{P(x|\theta)P(\theta)}{P(x)}$$

If we consider the data to be fixed, then $P(x)$ is the same for all possible models, so if we only want to compare models with different values of θ, we can ignore the denominator.

$$P(\theta|x) \propto P(x|\theta)P(\theta)$$

Or, more colloquially:

$$\text{Posterior} \propto \text{Likelihood} \times \text{Prior}$$

Distributions of parameters

A key difference between Bayesian analysis and frequentist analysis is that the parameters θ of the model are not fixed but described by probability distributions.

- Maximum likelihood estimation (and thus `lm`, `glm`, etc.) selects the *mode*[1] of the likelihood.
- A Bayesian point estimate of a parameter is more likely to be the *mean* of the posterior distribution.

[1] *Mode* is another word for *local maximum*.

The problem with priors

If a prior distribution has too much information, then it will require a lot of data to alter the posterior.

For example, suppose that we are quite certain that $\theta = \theta_0$, and so we choose a prior

$$\mathit{Prior}(\theta) \sim N(\theta_0, \delta)$$

with δ some very small number. The graph of this distribution is basically a spike at θ_0 and approximately zero everywhere else.

Then for any θ that is very different from θ_0, the posterior is still going to be approximately zero, and the data won't be able to change our minds.

Flat or uninformative priors

The solution to this issue is to work with a suitably uninformative prior.

- A uniform distribution can make a good prior.
    - Uniform on what scale?

For a flat prior, the posterior distribution is proportional to the likelihood.

Conjugate priors

Binary classification is particularly simple with a Bayesian method.

Recall the binomial distribution for x successes in N trials with probability p of success.

$$P(x|p) = \binom{N}{x} p^x (1-p)^{N-x}$$

Fixing x and N as coming from observed data and thinking of p as the variable makes $\binom{N}{x}$ a constant.

Previous observation of s successes and r failures would lead one to set the prior $P(p)$ to follow a Beta distribution, which takes a similar form:

$$P(p) \propto p^s (1-p)^r$$

Thus, by Bayes' rule:

$$P(p|x) \propto p^{x+s} (1-p)^{N-x+r}$$

the posterior is another Beta distribution.

Tadpole predation

In the subset of the tadpole data we looked at last week, there were 30 out of 40 tadpoles that survived the experiment.

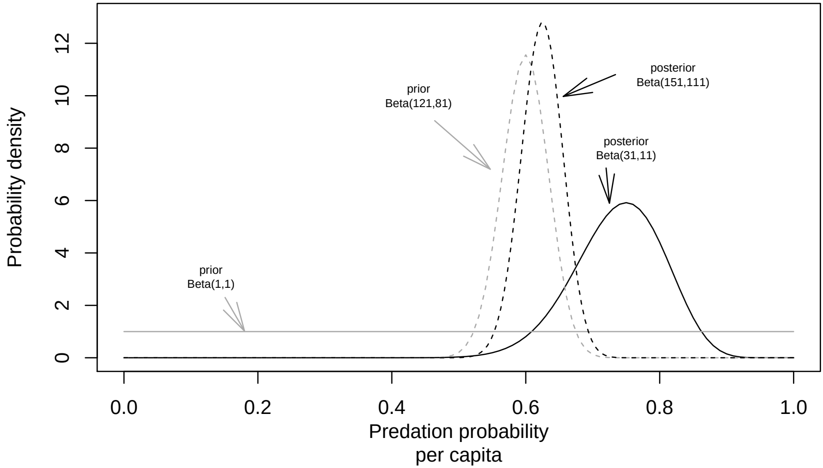


Figure 6.3 from Bolker (2008).

Some things to note

- With a flat prior, the posterior has mode equal to the maximum likelihood estimate. The mean is shifted slightly towards the mean of the prior.
  - Mode of Beta(31,11) is $(31 - 1)/(31 + 11 - 2) = 0.75$.
  - Mean of Beta(31,11) is $31/(31 + 11) = 0.738$.
- Having a conjugate prior like Binomial/ Beta can make the math easy but is not typical, and there are problems that can arise with some conjugate priors.
- One big experiment or many small experiments? Doesn't matter. The posterior can be updated one observation at a time if we want.

Bayesian tools in R

Many options. Bayesian Regression Models with Stan[2] (`brms`) seems good. See Bürkner (2018).

Start by fitting an error only (null) model for the tadpole data. Be warned, Bayesian computations can take some time.

```
fit_tad0 <- brm(surv | trials(density) ~ 1,
                family=binomial(), data=ReedfrogPred)
```

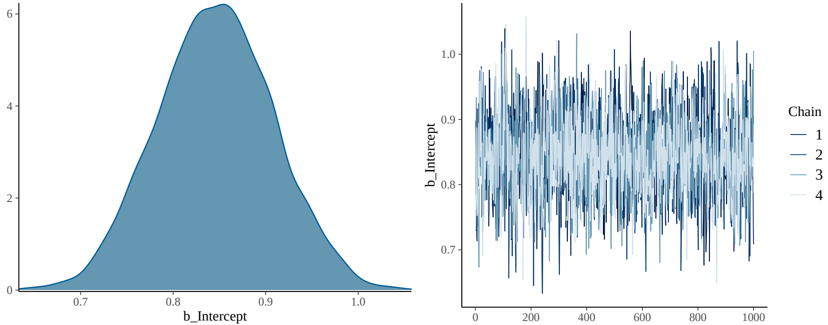
```
summary(fit_tad0)
```

```
##  Family: binomial
##   Links: mu = logit
## Formula: surv | trials(density) ~ 1
##    Data: ReedfrogPred (Number of observations: 48)
## 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     0.85      0.06     0.73     0.97 1.00     1662     1682
##
## 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).
```

[2] Stan is a program outside of R that does the computational heavy lifting

Plot

```
plot(fit_tad0)
```



This plot shows a density estimate for the model parameter, and a diagnostic graph of the convergence of the algorithm used to arrive at that estimate. The graph on the right indicates a problem if it looks like anything but random noise.

Accessing model parameters

- The coefficients of the `brm` object are accessed with `$fixed`.

```r
summary(fit_tad0)$fixed
```

```
##            Estimate  Est.Error  l-95% CI  u-95% CI     Rhat Bulk_ESS Tail_ES
## Intercept 0.8458727 0.06239771 0.7260189 0.9686483 1.002481     1662     168
```

- The intercept can be converted to a probability with the inverse link function.

```r
plogis(summary(fit_tad0)$fixed[1])
```

```
## [1] 0.6997006
```

- As expected, this is the overall probability of survival in the data.

```r
sum(ReedfrogPred$surv)/sum(ReedfrogPred$density)
```

```
## [1] 0.6991071
```

Set an informative prior

We can impose an informative prior, say of a 75% survival rate.

- Convert 75% to the model scale using the link function.

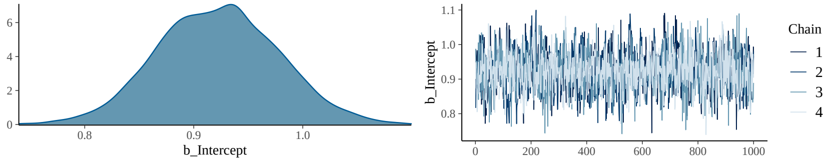
```
qlogis(0.75)
```

```
## [1] 1.098612
```

- Because `brm` creates code that is compiled outside of R, this number has to be included explicitly in the prior.

```
fit_tad0_prior <- brm(surv | trials(density) ~ 1,
                      family=binomial(), data=ReedfrogPred,
                      prior = set_prior("normal(1.098612,0.1)",
                                        class = "Intercept"))
```

```
plot(fit_tad0_prior)
```



A model with predictors

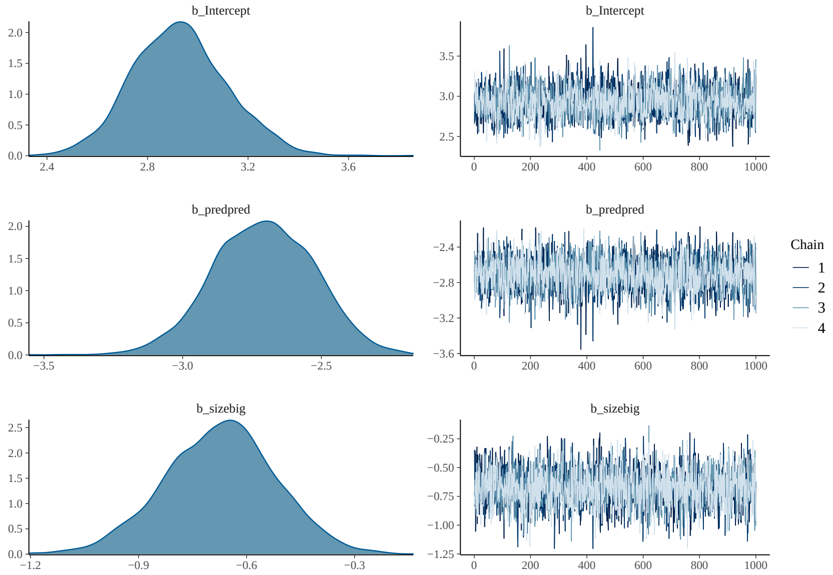
```
fit_tad <- brm(surv | trials(density) ~ pred + size,
               family=binomial(), data=ReedfrogPred)
```

```
summary(fit_tad)
```

```
##  Family: binomial
##   Links: mu = logit
## Formula: surv | trials(density) ~ pred + size
##    Data: ReedfrogPred (Number of observations: 48)
## 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     2.93      0.19     2.57     3.31 1.00     2062     2110
## predpred     -2.71      0.18    -3.07    -2.36 1.00     2272     2654
## sizebig      -0.67      0.15    -0.98    -0.38 1.00     2979     2657
##
## 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).
```

Plot brm output

```
plot(fit_tad)
```



The brm with an uninformative prior is close to a glm

```
summary(fit_tad)$fixed[,1:4]
```

```
##              Estimate Est.Error   l-95% CI   u-95% CI
## Intercept   2.9279091 0.1882503  2.5743609  3.3119765
## predpred   -2.7081062 0.1825929 -3.0693010 -2.3593588
## sizebig    -0.6733214 0.1547661 -0.9814715 -0.3789356
```

```
plogis(summary(fit_tad)$fixed["Intercept", "Estimate"])
```

```
## [1] 0.949209
```

```
exp(summary(fit_tad)$fixed[,"Estimate"])
```

```
##   Intercept    predpred     sizebig
## 18.68851334  0.06666293  0.51001180
```

```
glm_fit_tad <- glm(cbind(surv, density-surv) ~ pred + size,
                   family = binomial(), data = ReedfrogPred)
coef(summary(glm_fit_tad))
```

```
##               Estimate Std. Error    z value     Pr(>|z|)
## (Intercept)  2.9231663  0.1901078  15.376358 2.358447e-53
## predpred    -2.7039259  0.1854929 -14.576976 3.935674e-48
## sizebig     -0.6738492  0.1532298  -4.397639 1.094349e-05
```

Set an informative prior

Priors can be set on many parameters of the model.

```
fit_tad_prior <- brm(surv | trials(density) ~ pred + size,
                     family=binomial(), data=ReedfrogPred,
                     prior = set_prior("normal(-1,0.1)",
                                       class = "b",
                                       coef = "sizebig"))
```

```
summary(fit_tad_prior)
```

```
##  Family: binomial
##   Links: mu = logit
## Formula: surv | trials(density) ~ pred + size
##    Data: ReedfrogPred (Number of observations: 48)
## 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     3.08      0.18     2.75     3.43 1.00     1762     2103
## predpred     -2.75      0.19    -3.13    -2.39 1.00     2238     2401
## sizebig      -0.90      0.08    -1.07    -0.73 1.00     2908     2826
##
## 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).
```

A nonlinear model

The `brm` command will fit nonlinear models, such as the Myxomatosis example in Section 6.3 of Bolker (2008).

- You have to specify a prior, there's no default. Specify priors for nonlinear parameters with `nlpar`.
- The formula goes inside the function `bf` (for `brms` formula).
- Parameters of the nonlinear function are defined by formulas.
- Set `nl=TRUE` (for nonlinear).

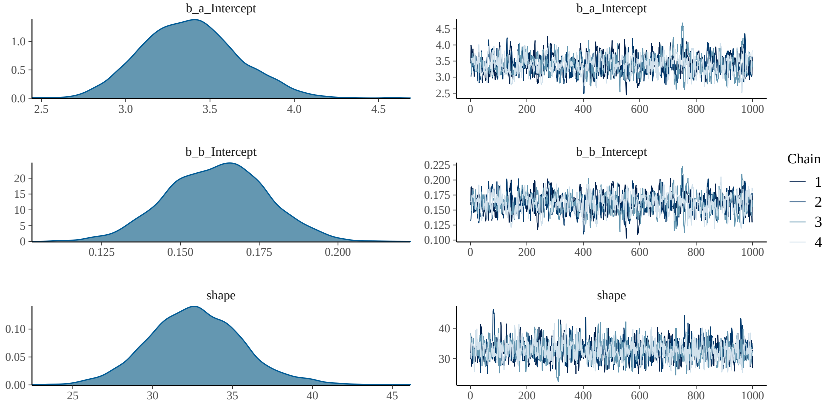
```
MyxoTiter_sum$fgrade <- factor(MyxoTiter_sum$grade)
prior1 <- prior(normal(1, 2), nlpar = "a") +
  prior(normal(0.2, 0.4), nlpar = "b") +
  prior(gamma(50, 0.1), class="shape")
fit_myx <- brm(bf(titer ~ a*day*exp(-b*day), a~fgrade, b~fgrade, nl=TRUE),
               data = MyxoTiter_sum,
               prior = prior1,
               family=Gamma(link = "identity"))
```

The function we are using, $\mu_{titer} = a_g t e^{-b_g t}$ is a reasonable model for a quantity that grows from zero to a peak, and then decays back to zero.

Nonlinear model diagnostic plots

See `?plot.brmsfit` or `?bayesplot` for options. This model seems to converge okay.

```r
plot(fit_myx, pars=c("b_a_Intercept", "b_b_Intercept", "shape"))
```



Model parameters

```
summary(fit_myx)
```

```
##  Family: gamma
##   Links: mu = identity; shape = identity
## Formula: titer ~ a * day * exp(-b * day)
##          a ~ fgrade
##          b ~ fgrade
##    Data: MyxoTiter_sum (Number of observations: 149)
## 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
## a_Intercept     3.38      0.29     2.86     3.96 1.01      795     1094
## a_fgrade3      -0.89      0.33    -1.56    -0.27 1.00      932     1459
## a_fgrade4      -1.38      0.30    -2.00    -0.83 1.00      824     1126
## a_fgrade5      -0.97      0.31    -1.61    -0.40 1.01      835     1141
## b_Intercept     0.16      0.02     0.13     0.19 1.01      792      964
## b_fgrade3      -0.06      0.02    -0.10    -0.03 1.00      821     1060
## b_fgrade4      -0.08      0.02    -0.11    -0.05 1.00      793      992
## b_fgrade5      -0.02      0.02    -0.05     0.02 1.01      800     1073
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## shape    32.71      2.96    27.19    39.03 1.00     1509     1718
##
## 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).
```

Grade 1 (the intercept) looks more virulent than the others.

Interpreting the deterministic model

Our deterministic model takes the form

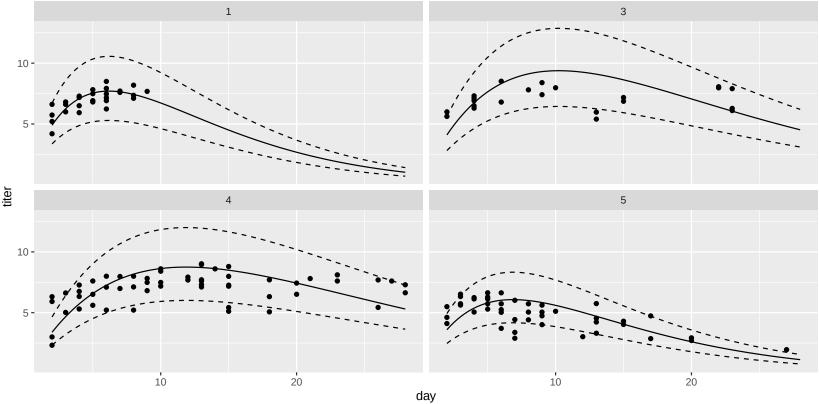
$$\mu_{titer} = a_g t e^{-b_g t}$$

The parameters relate to the onset rapidity and the recovery rate for each grade of the virus.

- a_g is the initial slope
- $1/b_g$ is the t-coordinate of the maximum.

```
x <- summary(fit_myx)$fixed[,"Estimate"]
myx_det <- function(day, grade){
  a <- x[1] + x[2]*(grade==3) + x[3]*(grade==4) + x[4]*(grade==5)
  b <- x[5] + x[6]*(grade==3) + x[7]*(grade==4) + x[8]*(grade==5)
  a*day*exp(-b*day)
}
```

Comparing the model to the data



The code to create this plot is ugly, so it's not included on the slide. It looks like the model does okay, but there might be some dynamics to the virus in the second week of infection that the model misses.

Note that all the rabbits infected with grade 1 die before day 10.

Leave-one-out cross validation

- For each observation, build a model excluding that observation.
- Compute the log-likelihood of the excluded observation.
- Combine all these log-likelihoods to assess the model.

```
loo(fit_myx, moment_match = TRUE)
```

```
## Warning: Some Pareto k diagnostic values are slightly high. See help('pareto-k-diagnostic') for details

##
## Computed from 4000 by 149 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -278.8 16.6
## p_loo        16.1  3.2
## looic       557.7 33.3
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     145   97.3%   408
##  (0.5, 0.7]   (ok)         4    2.7%   272
##    (0.7, 1]   (bad)        0    0.0%   <NA>
##    (1, Inf)   (very bad)   0    0.0%   <NA>
##
## All Pareto k estimates are ok (k < 0.7).
## See help('pareto-k-diagnostic') for details.
```

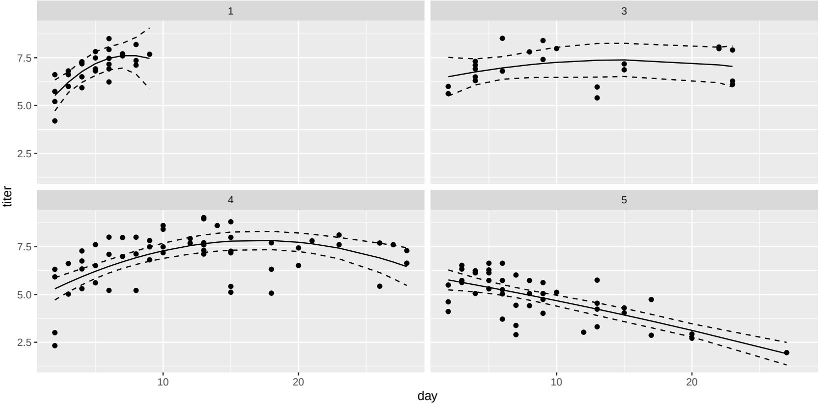
How does a glm do?

Try a quadratic in day for the deterministic part of the model.

```
myx_glm <- glm(titer~poly(day,2)*fgrade,
               family = Gamma(link = "identity"), data=MyxoTiter_sum)
summary(myx_glm)
```

```
##
## Call:
## glm(formula = titer ~ poly(day, 2) * fgrade, family = Gamma(link = "identity"),
##     data = MyxoTiter_sum)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.72518  -0.07252   0.02296   0.10848   0.32272
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             4.2571     3.1725   1.342    0.182
## poly(day, 2)1         -62.8351    67.8145  -0.927    0.356
## poly(day, 2)2         -40.6835    29.9023  -1.361    0.176
## fgrade3                 2.7455     3.1820   0.863    0.390
## fgrade4                 2.4561     3.1760   0.773    0.441
## fgrade5                 0.4325     3.1745   0.136    0.892
## poly(day, 2)1:fgrade3  64.4313    67.8661   0.949    0.344
## poly(day, 2)2:fgrade3  37.6151    30.1050   1.249    0.214
## poly(day, 2)1:fgrade4  70.0160    67.8333   1.032    0.304
## poly(day, 2)2:fgrade4  34.0673    29.9390   1.138    0.257
## poly(day, 2)1:fgrade5  50.5914    67.8263   0.746    0.457
## poly(day, 2)2:fgrade5  40.0406    29.9247   1.338    0.183
##
## (Dispersion parameter for Gamma family taken to be 0.02619576)
##
##     Null deviance: 10.6424  on 148  degrees of freedom
## Residual deviance:  4.1168  on 137  degrees of freedom
## AIC: 454.09
```

The GLM fits well but is less interpretable



If we just want to interpolate, this seems like a good model.

AIC

brms doesn't support AIC, but we can compute it if we want to. AIC_maybe of the Bayesian model looks reasonable - it's similar to the looic and not too far from the AIC for the glm, which is perhaps overfit. I did have to build it by hand, the numeric(0) means there isn't a brmsfit method for AIC.

```
AIC(myx_glm)
```

```
## [1] 454.0936
```

```
AIC(fit_myx)
```

```
## numeric(0)
```

```
llMyx <- log_lik(fit_myx)
# ?brms::logLik.brmsfit
nllMyx <- -sum(colMeans(llMyx))
(AIC_maybe <- 2*nllMyx-2*dim(fit_myx$data)[2])
```

```
## [1] 532.6945
```

References

Bolker, Benjamin M. 2008. *Ecological Models and Data in R*. Princeton University Press.

Bürkner, Paul-Christian. 2018. "Advanced Bayesian Multilevel Modeling with the R Package brms." *The R Journal* 10 (1): 395–411. https://doi.org/10.32614/RJ-2018-017.