Standard Errors, Priors, and Bridge Sampling: A Discussion of Liu et al.

Duco Veen[1] **and Irene Klugkist**[1]

[1] Methodology and Statistics, Social and Behavioral Sciences, Utrecht University

Liu, Hu, Cao, Wang and Chen (2019) present five methods to estimate the marginal likelihood of Item Response Theory (IRT) models and compare the performance on English examination data. On these data they compute the marginal likelihood for the 1Pl and 2Pl models using five different methods that were extensively described in the methods section of their paper. In this discussion, we will first shortly reflect on the standard errors of the estimates in relation to the number of iterations used. Then the choice of prior distributions and their potential impact is discussed. And finally, we will give results of an additional estimation approach.

Standard errors

The authors compare the performance of the different estimation methods in Tables 2 (estimates and standard errors) and 3 (sampling and computation time). As the authors point out, the Prior-Based Monte Carlo (PMC) method is theoretically correct and simple but in practice may require such huge MC samples that its use is unfeasible. This is also the case for this example: with 500000 iterations the MC standard errors are still very large, which is not surprising since the estimated model is high dimensional; for the 2PL model 570 parameters are sampled from the joint prior distribution of $(\boldsymbol{\theta}, \boldsymbol{b}, \boldsymbol{a}, \mu_b, \psi)$. So, although the time usage, as presented in Table 3, is relatively low and therefore one could consider increasing the number of iterations, this will very likely not solve this problem. One rarely samples from the high-likelihood regions as pointed out by Lartillot and Philippe (2006) and Liu et al. (2019). To illustrate how rarely samples will come from the high-likelihood region, consider for each parameter one value above and below the posterior median. That results in $2^{570} \approx 3.8 \times 10^{171}$ possible combination. To get a high-likelihood sample for all parameters jointly, one would need to sample somewhere in between all of these orthants in the parameter space. For an interesting discussion on Monte Carlo sampling in high dimensions see Carpenter (2019). In addition we like to point out that in the context of testing data and IRT models, data sets are often considerably larger than the example used here. In line with what the authors conclude, in our opinion the PMC method is not a realistic option for most IRT data.

In contrast, the standard errors of the Chib method are extremely small (e.g., 0.22 for a log ML estimate of -12184 for the 1Pl model). In fact, the standard errors lead us to believe that this method requires much fewer iterations than were applied in the analysis. In that sense we do not entirely agree with the conclusion the authors seem to suggest, namely that the Chib method is less attractive (e.g. compared with the faster Chen method) because of its long computation time. To make a fair comparison on time usage, perhaps the number of iterations should be chosen such that the standard errors of the estimated log marginal likelihoods are comparable.

Prior distributions

It is well-known that prior distributions can have a huge impact on marginal likelihoods (Kass and Raftery, 1995; Liu and Aitkin, 2008). Therefore, the specification and motivation for the priors is extremely important. For the possible impact of the priors specified for the discrimination parameters a_j in the 2PL model, the authors perform a sensitivity analysis. The resulting log ML values are not equal (ranging from -11878 to -11959) but do lead to the same conclusion in the sense that the 2PL model is favoured over the 1Pl model. Our main concern in this example is the bounded uniform prior used in the original analyses. This choice is not explicitly motivated and the constraints induced by this prior do not fit with the data. This is illustrated in our Figure 1, where we plot the estimated a_j's using the log normal prior $LN(0.5, 1)$. As can be seen from this figure, several posterior estimates fall outside the constraints of the bounded uniform that Liu et al. (2019) used for their Figure 1.

Bridge sampling

We believe that educational scientists or psychologist who are aiming to evaluate IRT models will prefer ready-to-use standard software over programming their own code. Therefor we explored the use of bridge sampling (Meng and Wong, 1996) for estimating the marginal likelihoods of the 1Pl and 2Pl models for the Examination data. Bridge sampling can be done using the R packages `rstan` (Stan Development Team, 2018) and `bridgesampling` (Gronau and Singmann, 2018). All code is provided in the supplementary file at `https://osf.io/kse9w/`. A tutorial on bridge sampling is provided by Gronau et al. (2017). Here, we will just report the resulting log marginal likelihood, using the same priors as Liu et al. (2019), but with the log normal $LN(0.5, 1)$ for the a_j's in the 2PL model instead of the bounded uniform, for the reasons described above. The marginal likelihoods are respectively -12183.88 (1pl) and -11882.68 (2pl), which are highly similar to those using the Chib, Chen or SSM method. Bridge sampling could thus be a viable, ready-to-use, alternative to calculate marginal likelihoods for IRT models.

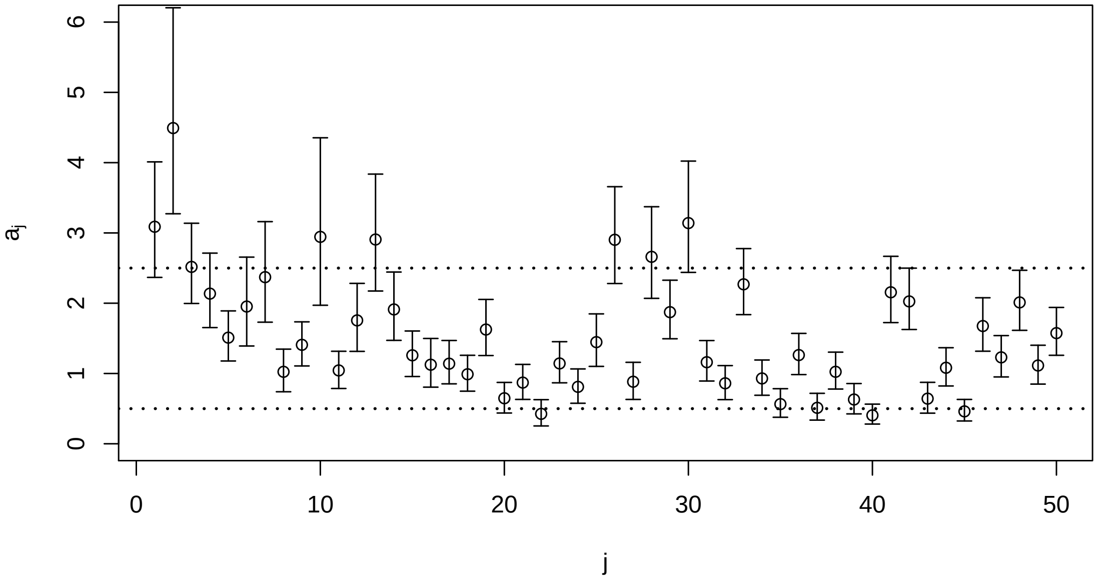


Figure 1: Posterior medians and 95% credible intervals of a_j's under 2PL model using $LN(0.5, 1)$ prior on a_j's. Dashed lines indicate bounds of $U(0.5, 2.5)$ prior.

Acknowledgement We like to thank the editor for inviting us to contribute a discussion paper, and Liu et al. for sharing the English examination data with us.

References

Carpenter, B. (2019). (Markov chain) Monte Carlo doesn't "explore the posterior". Available from: `https://statmodeling.stat.columbia.edu/2019/03/25/mcmc-does-not-explore-posterior/`.

Gronau, Q. F., Sarafoglou, A., Matzke, D., Ly, A., Boehm, U., Marsman, M., Leslie, D. S., Forster, J. J., Wagenmakers, E.-J., and Steingroever, H. (2017). A tutorial on bridge sampling. *Journal of mathematical psychology*, 81:80–97.

Gronau, Q. F. and Singmann, H. (2018). *bridgesampling: Bridge Sampling for Marginal*

Likelihoods and Bayes Factors. Available from: `https://CRAN.R-project.org/package=bridgesampling`.

Kass, R. E. and Raftery, A. E. (1995). Bayes factors. *Journal of the american statistical association*, 90(430):773–795.

Lartillot, N. and Philippe, H. (2006). Computing Bayes factors using thermodynamic integration. *Systematic biology*, 55(2):195–207.

Liu, C. C. and Aitkin, M. (2008). Bayes factors: Prior sensitivity and model generalizability. *Journal of Mathematical Psychology*, 52(6):362–375.

Liu, Y., Ha, G., Wang, X., and Chen, M.-H. (2019). A Comparison of Monte Carlo Methods for Computing Marginal Likelihoods of Item Response Theory Models.

Meng, X.-L. and Wong, W. H. (1996). Simulating ratios of normalizing constants via a simple identity: a theoretical exploration. *Statistica Sinica*, pages 831–860.

Stan Development Team (2018). *RStan: the R interface to Stan*. Available from: `http://mc-stan.org/`.

Supplementairy files for 'Standard Errors, Priors, and Bridge Sampling: A Discussion of Liu et al.'

Dcuo Veen and Irene Klugkist

21 mei 2019

Contents

This document illustrates how marginal likelihoods can be obtained using a combination of stan, via the r packages rstan (Stan Development Team 2018) and bridgesampling (Gronau and Singmann 2018). This document is a supplementary file to the discussion by Veen and Klugkist of the paper by Liu, Cao, Wang and Chen (2019)

Data

We load in the data:

```r
data <- read.csv("irtdata.csv")
head(data)
```

```
##   Class.ID Student.ID Item1 Item2 Item3 Item4 Item5 Item6 Item7 Item8
## 1        1          1     1     1     1     1     1     0     1     1
## 2        1          2     1     1     1     1     1     1     1     0
## 3        1          3     1     1     1     1     0     1     1     1
## 4        1          4     1     1     1     1     1     1     1     1
## 5        1          5     1     1     0     1     0     1     1     1
## 6        1          6     1     1     1     1     1     1     1     1
##   Item9 Item10 Item11 Item12 Item13 Item14 Item15 Item16 Item17 Item18
## 1     1      1      1      1      1      1      1      1      0      0
## 2     1      1      0      1      1      1      1      1      0      0
## 3     1      1      1      1      1      1      1      1      1      1
## 4     0      1      0      1      1      1      0      1      1      0
## 5     0      1      1      1      1      1      1      0      1      0
## 6     0      1      1      1      1      1      1      1      1      0
##   Item19 Item20 Item21 Item22 Item23 Item24 Item25 Item26 Item27 Item28
## 1      1      0      1      0      1      1      1      1      1      1
## 2      1      1      1      1      1      1      1      1      1      1
## 3      1      0      1      1      1      1      0      1      0      1
## 4      0      0      0      0      1      0      1      1      1      1
## 5      0      0      0      0      1      1      1      1      0      1
## 6      1      0      1      0      1      1      0      1      1      1
##   Item29 Item30 Item31 Item32 Item33 Item34 Item35 Item36 Item37 Item38
## 1      1      1      1      1      1      1      0      0      0      1
## 2      1      1      1      1      0      1      1      1      0      0
## 3      1      1      1      0      1      0      1      1      0      0
## 4      1      1      1      0      1      1      0      1      0      1
## 5      1      1      0      1      1      1      0      1      0      0
## 6      0      1      1      0      1      1      0      1      1      0
##   Item39 Item40 Item41 Item42 Item43 Item44 Item45 Item46 Item47 Item48
## 1      0      1      1      1      1      1      0      0      1      1
## 2      1      1      1      1      0      0      0      1      0      1
## 3      0      1      1      1      0      0      1      1      1      1
## 4      0      1      1      1      0      1      1      0      1      1
## 5      0      1      1      1      1      0      0      1      0      1
## 6      0      0      1      1      0      0      0      1      1      1
##   Item49 Item50
## 1      1      0
## 2      1      1
## 3      0      1
## 4      1      1
## 5      1      1
## 6      1      0
```

We received the data from Liu et al. (2019), the data should be the same as for their paper. There is one notable difference however. Liu et al. (2019) noted on page 14 that 452 students attended the exam for their data, however the data we received contained 468 cases.

```
nrow(data)
```

```
## [1] 468
```

The other summaries they provide are however exactly in line with the data they provided us, namely an overal item accuracy of 0.607;

```
sum(data[, c(3:52)]) / (dim(data[, c(3:52)])[1] * dim(data[, c(3:52)])[2])
```

```
## [1] 0.6073077
```

the accuracy of item 10 of 0.929;

```
mean(data[, 12])
```

```
## [1] 0.9294872
```

and the accuracy of item 45 of 0.177;

```
mean(data[, 47])
```

```
## [1] 0.1773504
```

We therefore continue under the assumption that this in indeed the same data as Liu et al. (2019) used in their paper.

Model Specification

We use one of the priors for a_j as defined in the sensitivity analysis of Liu et al. (2019) for the 2PL, namely the log normal prior $LN(0.5, 1)$ as we did not want to put additional constraints on the parameter bounds of a_j on top of the lower bound of 0. For the 1PL all a_j are one and no prior needs to be specified.

We specified the 1PL model in stan using the following model code:

```
data {
  int<lower=1> I;             // number of students
  int<lower=1> J;             // number of questions
  int<lower=0,upper=1> y[I, J];  // correctness for observations
}

parameters {
  real mu_beta;               // mean question difficulty
  vector[I] theta;            // ability
  vector[J] beta;               // difficulty for k
  real<lower=0> psi_inv;    // scale of difficulties
}

model{
 target += normal_lpdf(theta | 0, 1);
 target += normal_lpdf(beta | mu_beta, sqrt(psi_inv));
 target += normal_lpdf(mu_beta | 0, sqrt(4*psi_inv));
 target += inv_gamma_lpdf(psi_inv | 1, 1);

 for(i in 1:I){
   for(j in 1:J){
     target += bernoulli_logit_lpmf(y[i, j] | 1 * (theta[i] - beta[j]));
   }
 }
}
```

We specified the 2PL model in stan using the following model code:

```
data {
  int<lower=1> I;               // number of students
  int<lower=1> J;               // number of questions
  int<lower=0,upper=1> y[I, J];   // correctness for observations
}

parameters {
  real mu_beta;                 // mean question difficulty
  vector[I] theta;              // ability
  vector[J] beta;               // difficulty for k
  vector<lower=0>[J] alpha;    // discrimination of k
  real<lower=0> psi_inv;    // scale of difficulties
}

model{
 target += normal_lpdf(theta | 0, 1);
 target += normal_lpdf(beta | mu_beta, sqrt(psi_inv));
 target += lognormal_lpdf(alpha | 0.5, 1);
 target += normal_lpdf(mu_beta | 0, sqrt(4*psi_inv));
 target += inv_gamma_lpdf(psi_inv | 1, 1);

 for(i in 1:I){
   for(j in 1:J){
     target += bernoulli_logit_lpmf(y[i, j] | alpha[j] .* (theta[i] - beta[j]));
   }
 }
}
```

Note that we use the += notation in stan instead of the ~ notation, otherwise the bridgesampling will not work correctly (Gronau 2018).

Model Estimation

```
fit_1pl <- rstan::sampling(chen1pl, data = list(y = data[, 3:52],
                                                I = 468,
                                                J = 50),
                           seed = 499987576, iter = 10000)
```

```
fit_2pl <- rstan::sampling(chen2pl, data = list(y = data[, 3:52],
                                                I = 468,
                                                J = 50),
                           seed = 499987576, iter = 10000)
```

Obtaining the marginal likelihood

We use bridgesampling to obtain the marginal likelihood for the 1PL and 2PL:

```
bridge_1pl <- bridge_sampler(fit_1pl, method = "normal", maxiter = 2000,
                              silent = TRUE)
```

```
bridge_2pl <- bridge_sampler(fit_2pl, method = "normal", maxiter = 2000,
                              silent = TRUE)
```

The obtained values are:

```
bridge_1pl
```

```
## Bridge sampling estimate of the log marginal likelihood: -12183.88
## Estimate obtained in 17 iteration(s) via method "normal".
```

```
bridge_2pl
```

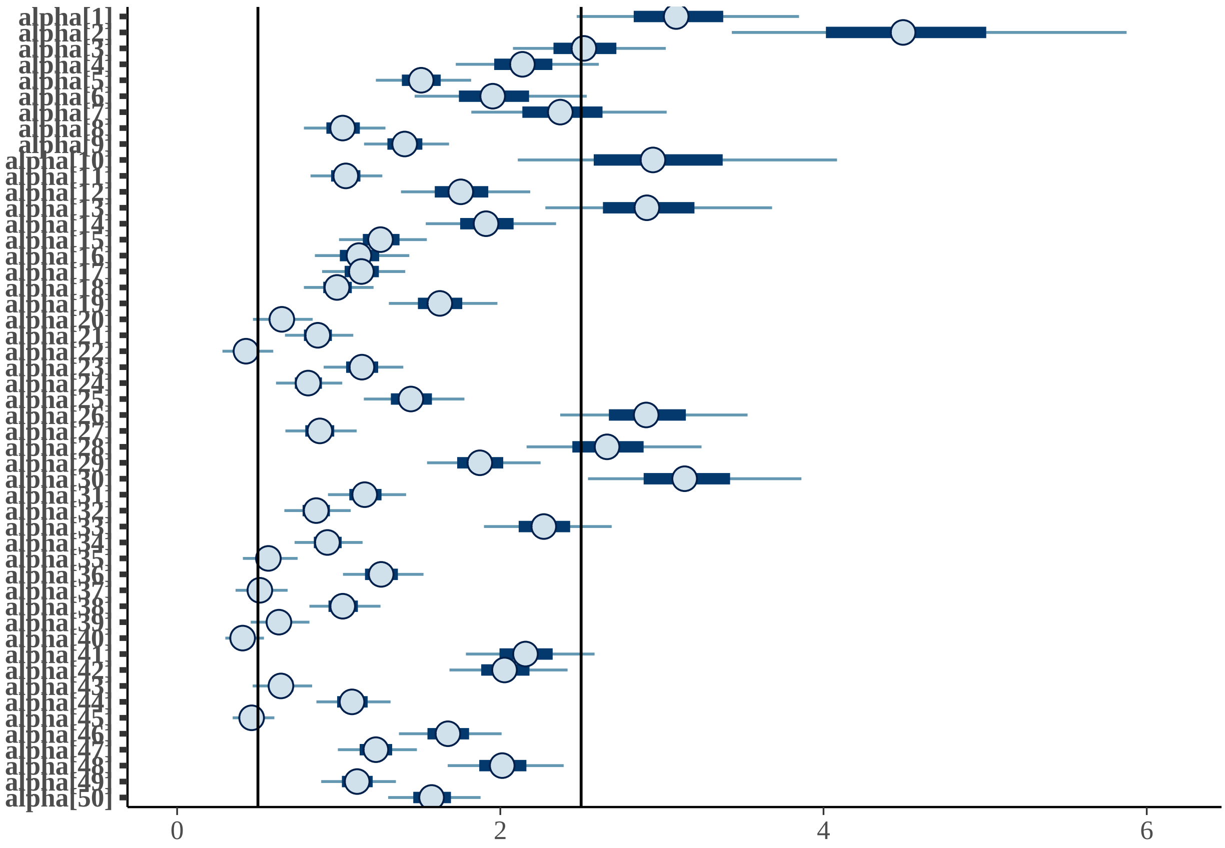
```
## Bridge sampling estimate of the log marginal likelihood: -11882.68
## Estimate obtained in 10 iteration(s) via method "normal".
```

This values for these marginal likelihoods are very close to the values reported in the paper of Liu et al. (2019) for the `Chib`, `Chen`, and `SSM` Monte Carlo Methods.

Discrimination parameter (a) estimates

We plot our obtained estimates for a_j together with two lines that represent the hard constraint that would be specified if the prior for a_j would be a Uniform prior between 0.5 and 2.5.

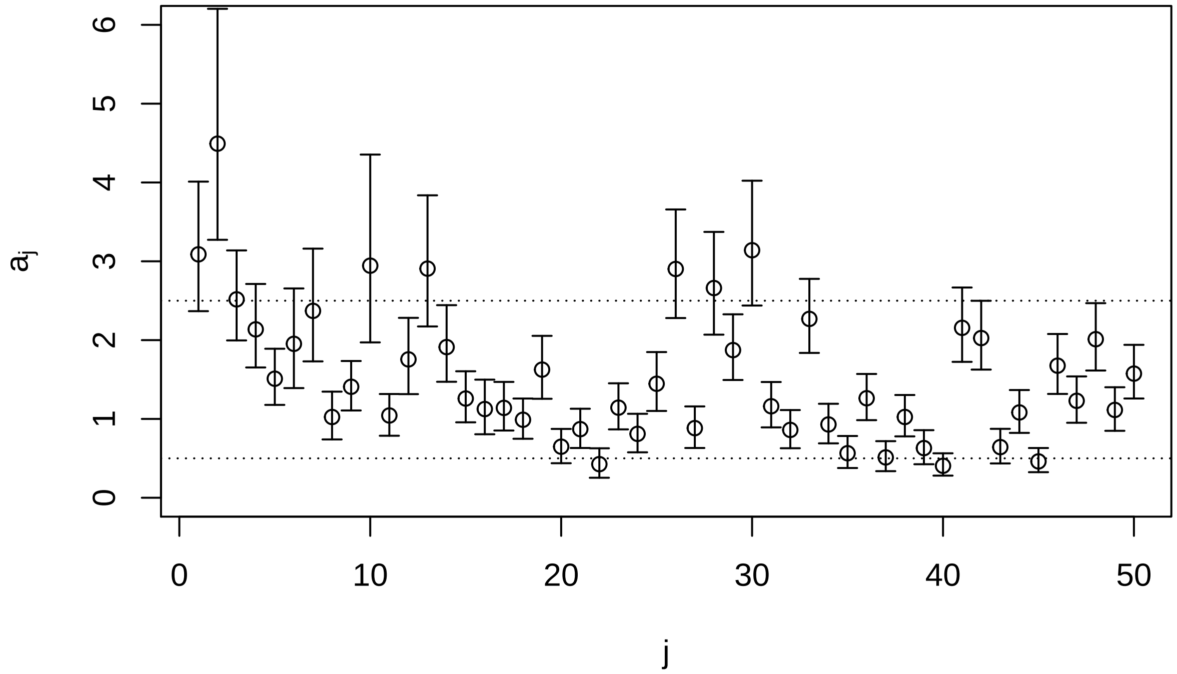
```
bayesplot::mcmc_intervals(extract(fit_2pl, pars = "alpha", permuted = FALSE)) +
  vline_at(0.5) + vline_at(2.5)
```



Or to show it in the same Figure style as Liu et al. (2019).

```
# pdf("Figure1_discussion.pdf", width = 10, height = 6)
plot(x = 1:50, y = summary(fit_2pl, pars = "alpha")$summary[, 6],
     xlab = "j", ylab = expression(a[j]), ylim = c(0,6))

for(i in 1:50){
  arrows(i, summary(fit_2pl, pars = "alpha")$summary[i, 4],
         i, summary(fit_2pl, pars = "alpha")$summary[i, 8],
         length=0.05, angle=90, code=3 )
}
abline(h = 0.5, lty = 3)
abline(h = 2.5, lty = 3)
```



```r
# dev.off()
```

We see that there are multiple estimates of a_j that would fall outside these constraints.

Session Information

```r
sessionInfo()
```

```
## R version 3.5.1 (2018-07-02)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 17134)
## 
## Matrix products: default
## 
## locale:
## [1] LC_COLLATE=Dutch_Netherlands.1252  LC_CTYPE=Dutch_Netherlands.1252   
## [3] LC_MONETARY=Dutch_Netherlands.1252 LC_NUMERIC=C                      
## [5] LC_TIME=Dutch_Netherlands.1252    
## 
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base     
## 
## other attached packages:
## [1] bindrcpp_0.2.2       bridgesampling_0.6-0 bayesplot_1.6.0.9000
## [4] rstan_2.18.2         StanHeaders_2.18.0   ggplot2_2.2.1.9000  
## 
```

```
## loaded via a namespace (and not attached):
##  [1] Brobdingnag_1.2-6  tidyselect_0.2.5   xfun_0.4          
##  [4] reshape2_1.4.3     purrr_0.2.5        lattice_0.20-35   
##  [7] colorspace_1.3-2   htmltools_0.3.6    stats4_3.5.1      
## [10] loo_2.0.0          yaml_2.2.0         base64enc_0.1-3   
## [13] rlang_0.3.0.1      pkgbuild_1.0.2     pillar_1.3.1      
## [16] glue_1.3.0         matrixStats_0.54.0 plyr_1.8.4        
## [19] bindr_0.1.1        stringr_1.3.1      munsell_0.5.0     
## [22] gtable_0.2.0       mvtnorm_1.0-8      coda_0.19-2       
## [25] evaluate_0.12      labeling_0.3       inline_0.3.15     
## [28] knitr_1.21         callr_3.0.0        ps_1.2.1          
## [31] parallel_3.5.1     Rcpp_1.0.0         scales_1.0.0      
## [34] backports_1.1.3    gridExtra_2.3      digest_0.6.18     
## [37] stringi_1.2.4      processx_3.2.0     dplyr_0.7.8       
## [40] grid_3.5.1         rprojroot_1.3-2    cli_1.0.1         
## [43] tools_3.5.1        magrittr_1.5       lazyeval_0.2.1    
## [46] tibble_1.4.2       crayon_1.3.4       pkgconfig_2.0.2   
## [49] Matrix_1.2-14      prettyunits_1.0.2  ggridges_0.5.1    
## [52] assertthat_0.2.0   rmarkdown_1.10     R6_2.3.0          
## [55] compiler_3.5.1
```

References

Gronau, Quentin F. 2018. "Hierarchical Normal Example (Stan)." https://cran.r-project.org/web/packages/bridgesampling/vignettes/bridgesampling_example_stan.html.

Gronau, Quentin F., and Henrik Singmann. 2018. *Bridgesampling: Bridge Sampling for Marginal Likelihoods and Bayes Factors*. https://CRAN.R-project.org/package=bridgesampling.

Liu, Yang, Guanyu Ha, Xiaojing Wang, and Ming-Hui Chen. 2019. "A Comparison of Monte Carlo Methods for Computing Marginal Likelihoods of Item Response Theory Models."

Stan Development Team. 2018. *RStan: The R Interface to Stan*. http://mc-stan.org/.