brms: An R Interface for Flexible Bayesian Multilevel Modeling using Stan

Paul Bürkner

2020-11-27

Why use Bayesian Statistics?

Advantages and Disadvantages of Bayesian Statistics

Advantages:

- Natural approach to expressing uncertainty
- **Ability to incorporate prior information**
- Increased modeling flexibility
- Full posterior distribution of parameters
- **Natural propagation of uncertainty**

Disadvantages:

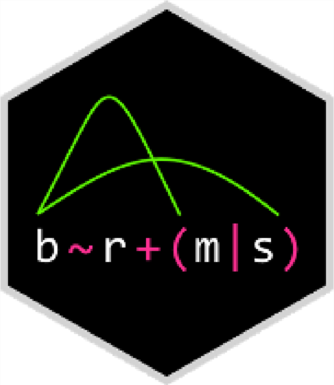
- **Slow Speed of model estimation**

Stan

A probabilistic programming language

brms

A unified framework for Bayesian regression models



Purpose of brms

Make Stan usable for a wider range of audience

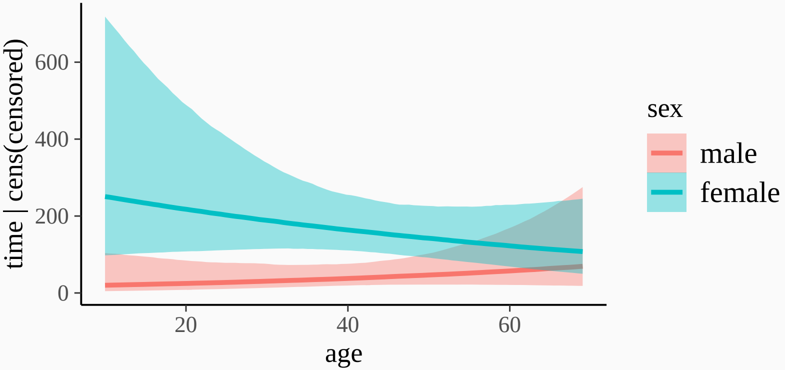
- Use R formula syntax
- Internally write Stan code that is readable yet fast
- Provide an easy interface for defining priors
- Facilitate post-processing

brms is officially supported by the Stan Development Team

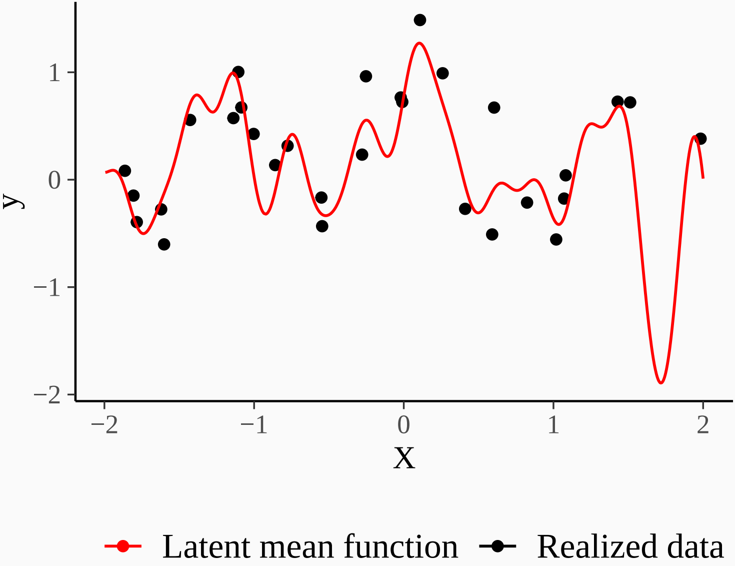
Example: Censored Recurrance Times of Kidney Infections

```
fitk <- brm(time | cens(censored) ~
              age * sex + (1|patient),
            data = kidney, family = weibull())
```

```
conditional_effects(fitk, "age:sex")
```



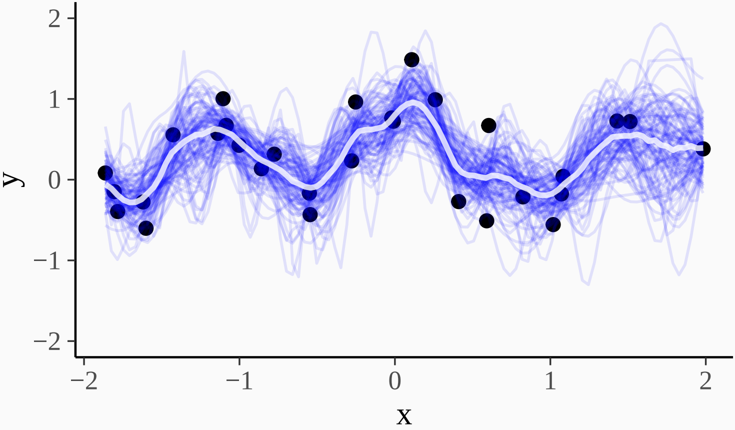
Example: Complex Non-Linear Relationships



Modeling Non-Linear Relationships with Gaussian Processes

```
fitgp <- brm(y ~ gp(x), bdata)
```

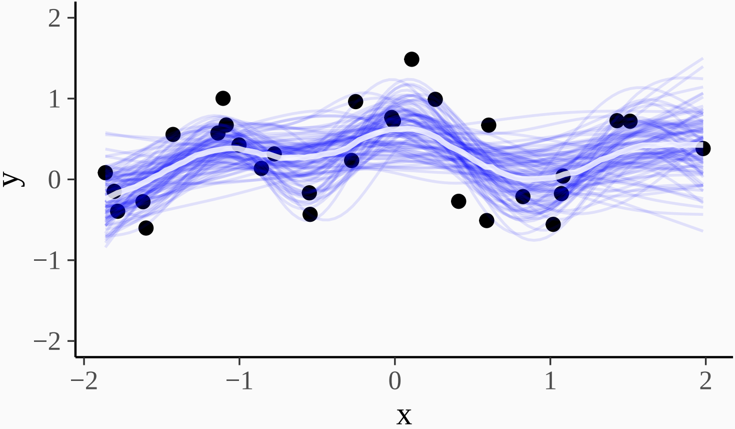
```
conditional_effects(fitgp, nsamples = 100, spaghetti = TRUE)
```



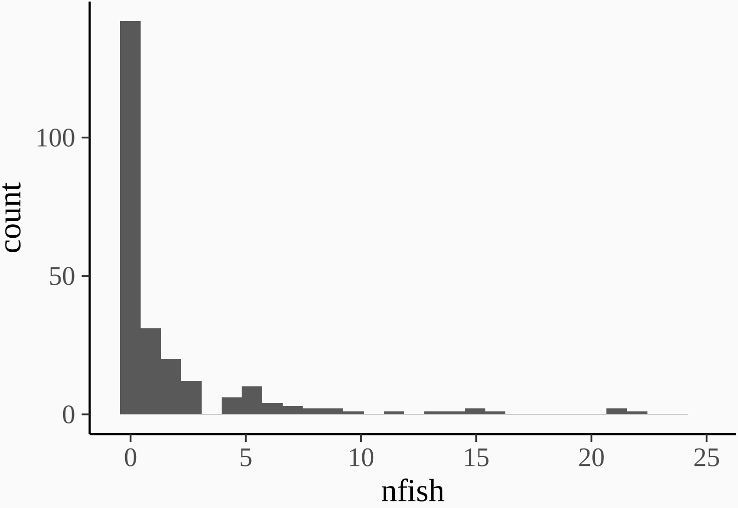
Modeling Non-Linear Relationships with Splines

```r
fits <- brm(y ~ s(x), bdata)
```

```r
conditional_effects(fits, nsamples = 100, spaghetti = TRUE)
```



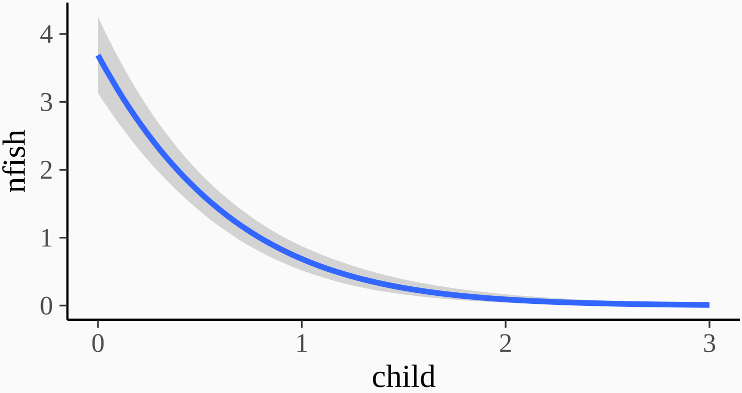
Example: Number of Fish Caught at a Camping Place



Modeling Zero-Inflation

```r
form <- bf(nfish ~ persons + child + camper, zi ~ child)
fit_zinb <- brm(form, zinb, zero_inflated_poisson())
```

```r
conditional_effects(fit_zinb, effects = "child")
```



Learn more about Stan

- Website: http://mc-stan.org/
- Manual: http://mc-stan.org/users/documentation/index.html
- Forums: http://discourse.mc-stan.org/

Selected Publications:

- Carpenter B., Gelman A., Hoffman M. D., Lee D., Goodrich B., Betancourt M., Brubaker M., Guo J., Li P., and Riddell A. (2017). Stan: A probabilistic programming language. *Journal of Statistical Software*. 76(1). 10.18637/jss.v076.i01
- Gelman A., Lee D., and Guo J. (2015). Stan: A probabilistic programming language for Bayesian inference and optimization. *Journal of Education and Behavioral Statistics*. 40(5):530–543.

Learn more about brms

- Help within R: `help("brms")`
- Vignettes: `vignette(package = "brms")`
- List of all methods: `methods(class = "brmsfit")`
- GitHub: https://github.com/paul-buerkner/brms
- Forums: http://discourse.mc-stan.org/
- Contact me: `paul.buerkner@gmail.com`
- Twitter: @paulbuerkner

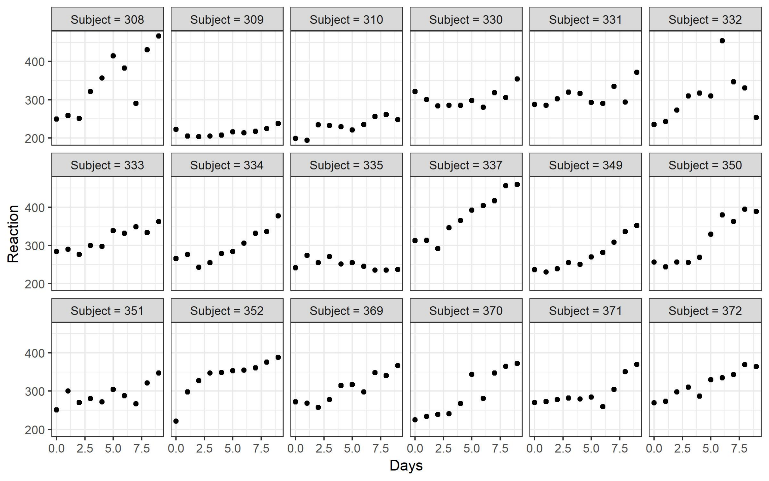
Publications

- Bürkner P. C. (2017). brms: An R Package for Bayesian Multilevel Models using Stan. *Journal of Statistical Software*. 80(1), 1-28. doi:10.18637/jss.v080.i01
- Bürkner P. C. (2018). Advanced Bayesian Multilevel Modeling with the R Package brms. *The R Journal*. 10(1), 395–411. doi:10.32614/RJ-2018-017

Appendix

Why using Multilevel Models?

Example: Effects of Sleep Deprivation on Reaction Times



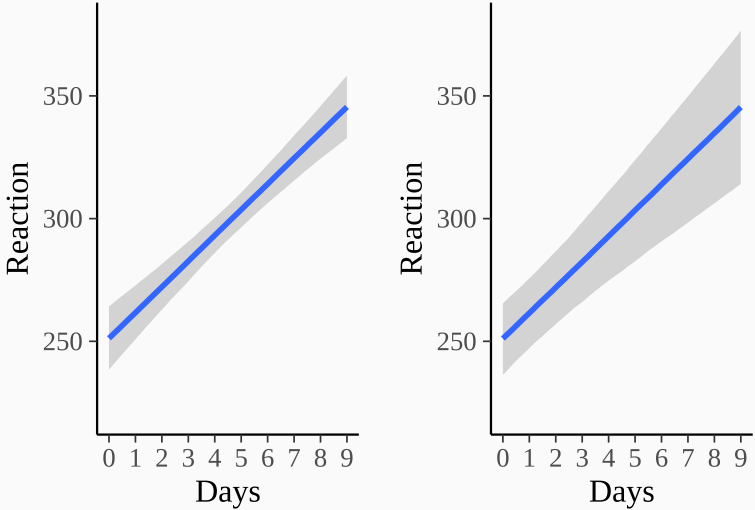
Frequentist Multilevel Models with lme4

```r
fit <- lmer(Reaction ~ 1 + Days + (1 + Days|Subject),
            data = sleepstudy)
```

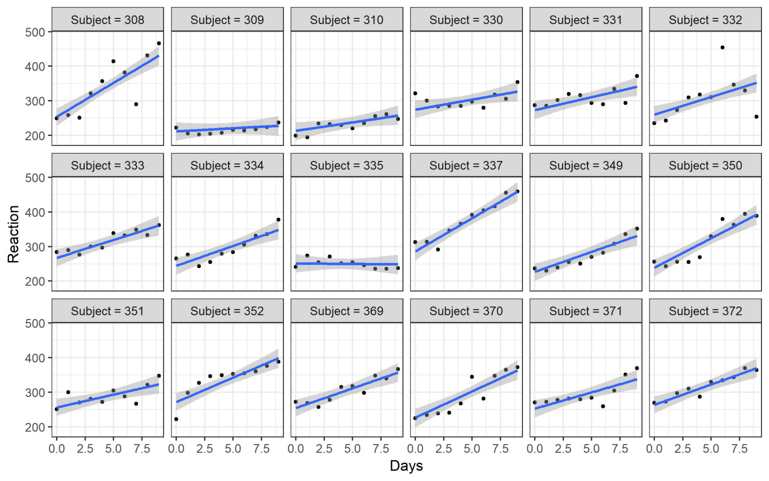
Bayesian Multilevel Models with brms

```r
fit <- brm(Reaction ~ 1 + Days + (1 + Days|Subject),
           data = sleepstudy)
```

Linear Regression vs. Multilevel Regression



Regression Lines for Specific Subjects



Stan Overview

- Probabilistic programming language...
- ... to fit open-ended Bayesian models
- Powerful sampling algorithms: HMC and NUTS
- Automatic differentiation library
- Runs on all major platforms (Windows, OS X, Linux)
- Has interfaces to R, Python, and many other languages

Post-Processing Methods in brms

```r
methods(class = "brmsfit")
```

```
##  [1] add_criterion           add_ic                  as.array                as.data.frame
##  [5] as.matrix               as.mcmc                 autocor                 bayes_factor
##  [9] bayes_R2                bridge_sampler          coef                    conditional_effects
## [13] conditional_smooths     control_params          cv_varsel               expose_functions
## [17] family                  fitted                  fixef                   formula
## [21] get_refmodel            getCall                 hypothesis              kfold
## [25] launch_shinystan        log_lik                 log_posterior           logLik
## [29] loo                     LOO                     loo_compare             loo_linpred
## [33] loo_model_weights       loo_moment_match        loo_predict             loo_predictive_interval
## [37] loo_R2                  loo_subsample           marginal_effects        marginal_smooths
## [41] mcmc_plot               model.frame             model_weights           neff_ratio
## [45] ngrps                   nobs                    nsamples                nuts_params
## [49] pairs                   parnames                plot                    post_prob
## [53] posterior_average       posterior_epred         posterior_interval      posterior_linpred
## [57] posterior_predict       posterior_samples       posterior_smooths       posterior_summary
## [61] pp_average              pp_check                pp_mixture              predict
## [65] predictive_error        predictive_interval     prepare_predictions     print
## [69] prior_samples           prior_summary           ranef                   reloo
## [73] residuals               rhat                    stancode                standata
## [77] stanplot                summary                 update                  VarCorr
## [81] varsel                  vcov                    waic                    WAIC
## see '?methods' for accessing help and source code
```