Bayesian Item Response Modeling in R with brms and Stan

Paul Bürkner

Paper

Bürkner P. C. (accepted). Bayesian Item Response Modeling in R with brms and Stan. *Journal of Statistical Software*.

IRT Models

We need:

- A set of parameters ξ_i for item i
- A set of parameters θ_p for person p
- A model for the responses y_{ip}

$$y_{ip} \sim \text{model}(\xi_i, \theta_p)$$

- Some restrictions on the parameters (ξ, θ)

Bayesian IRT Models

We need:

- A set of parameters ξ_i for item i
- A set of parameters θ_p for person p
- A model for the responses y_{ip}

$$y_{ip} \sim \text{model}(\xi_i, \theta_p)$$

- Some priors on the parameters (ξ, θ):

$$\xi_i \sim \text{prior}(.)$$

$$\theta_p \sim \text{prior}(.)$$

IRT models as Distributional Regression

Consider the data to be in long format and a (pointwise) likelihood with *distributional parameters* ψ_1 to ψ_K:

$$y \sim \text{likelihood}(\psi_1, \psi_2, \ldots, \psi_K)$$

Connect the distributional parameters to the item and person parameters via response functions f_k:

$$\psi_k = f_k(\xi_i, \theta_p)$$

IRT Models of Binary Responses

Binary response y and a single distributional parameter ψ:

$$y \sim \text{Bernoulli}(\psi) = \psi^y (1 - \psi)^{1-y},$$

Rasch Model:

$$\psi = f(\xi_i + \theta_p) = \frac{\exp(\theta_p + \xi_i)}{1 + \exp(\theta_p + \xi_i)}$$

2PL Model:

$$\psi = f(\alpha_i(\theta_p + \xi_i))$$

3PL Model:

$$\psi = \gamma_i + (1 - \gamma_i)\, f(\alpha_i(\theta_p + \xi_i))$$

Priors (Examples)

Non-hierarchical prior for item parameters:

$$\xi_i \sim \text{Normal}(0, 3)$$

Hierarchical prior for single parameter per item:

$$\xi_i \sim \text{Normal}(0, \sigma_\xi)$$

$$\sigma_\xi \sim \text{Normal}_+(0, 1)$$

Priors (Examples)

Hierarchical prior for multiple parameters per item:

$$(\xi_{1i}, \ldots, \xi_{Ki}) \sim \mathrm{MultiNormal}(0, \Sigma_\xi)$$

Decompose the covariance matrix Σ_ξ as:

$$\Sigma_\xi = D(\sigma_{\xi 1}, \ldots, \sigma_{\xi K})\, \Omega_\xi\, D(\sigma_{\xi 1}, \ldots, \sigma_{\xi K})$$

$$\sigma_{\xi k} \sim \mathrm{Normal}_+(0, 1)$$

$$\Omega_\xi \sim \mathrm{LKJ}(1)$$

Stan and brms

Model specification in brms: `family`

General structure:

```
family = brmsfamily(
  family = "<family>", link = "<link>",
  <more_link_arguments>
)
```

Binary Model:

```
family = brmsfamily(family = "bernoulli", link = "logit")
```

Gaussian Model:

```
family = brmsfamily(
  family = "gaussian", link = "identity",
  link_sigma = "log"
)
```

Model Specification in brms: `formula`

Item parameters have independent priors, person parameters have hierarchical priors:

```
formula = y ~ 0 + item + (1 | person)
```

Both item and person parameters have hierarchical priors:

```
formula = y ~ 1 + (1 | item) + (1 | person)
```

Add a covariate:

```
formula = y ~ 1 + x + (1 | item) + (1 | person)
```

Model Specification in brms: `formula`

Linear formulas for multiple distributional parameters:

```
formula = bf(
  y ~ 1 + (1 | item) + (1 | person),
  par2 ~ 1 + (1 | item) + (1 | person),
  par3 ~ 1 + (1 | item) + (1 | person),
)
```

Non-linear formula for a single distributional parameter:

```
formula = bf(
  y ~ fun(x, nlpar1, nlpar2),
  nlpar1 ~ 1 + (1 | item) + (1 | person),
  nlpar2 ~ 1 + (1 | item),
  nl = TRUE
)
```

Model Specification in brms: `formula`

Linear formulas for multiple distributional parameters:

```
formula = bf(
  y ~ 1 + (1 |i| item) + (1 |p| person),
  par2 ~ 1 + (1 |i| item) + (1 |p| person),
  par3 ~ 1 + (1 |i| item) + (1 |p| person),
)
```

Non-linear formula for a single distributional parameter:

```
formula = bf(
  y ~ fun(x, nlpar1, nlpar2),
  nlpar1 ~ 1 + (1 |i| item) + (1 |p| person),
  nlpar2 ~ 1 + (1 |i| item),
  nl = TRUE
)
```

Case Study: The VerbAgg Data Set

```r
data("VerbAgg", package = "lme4")
```

Anger	Gender	item	resp	id	btype	situ	mode	r2
20	M	S1WantCurse	no	1	curse	other	want	N
11	M	S1WantCurse	no	2	curse	other	want	N
17	F	S1WantCurse	perhaps	3	curse	other	want	Y
21	F	S1WantCurse	perhaps	4	curse	other	want	Y
17	F	S1WantCurse	perhaps	5	curse	other	want	Y
21	F	S1WantCurse	yes	6	curse	other	want	Y
39	F	S1WantCurse	yes	7	curse	other	want	Y
21	F	S1WantCurse	no	8	curse	other	want	N
24	F	S1WantCurse	no	9	curse	other	want	N
16	F	S1WantCurse	yes	10	curse	other	want	Y

Fitting a Rasch Model

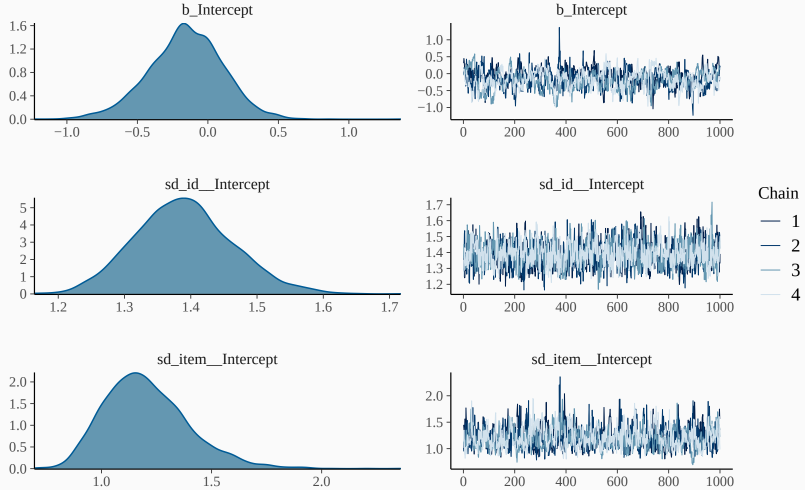
```r
formula_va_1pl <- bf(r2 ~ 1 + (1 | item) + (1 | id))
```

```r
prior_va_1pl <-
  prior("normal(0, 3)", class = "sd", group = "id") +
  prior("normal(0, 3)", class = "sd", group = "item")
```

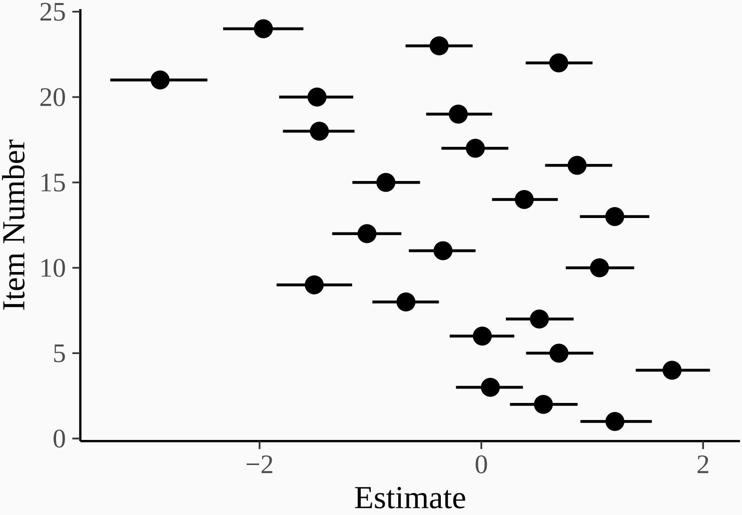
```r
fit_va_1pl <- brm(
  formula = formula_va_1pl,
  data = VerbAgg,
  family = brmsfamily("bernoulli", "logit"),
  prior = prior_va_1pl
)
```

Rasch Model: Investigate the Posterior

```
plot(fit_va_1pl)
```



Rasch Model: Item Parameters



Fitting a 2PL Model

```
formula_va_2pl <- bf(
  r2 ~ exp(logalpha) * eta,
  eta ~ 1 + (1 |i| item) + (1 | id),
  logalpha ~ 1 + (1 |i| item),
  nl = TRUE
)
```

```
prior_va_2pl <-
  prior("normal(0, 5)", class = "b", nlpar = "eta") +
  prior("normal(0, 1)", class = "b", nlpar = "logalpha") +
  prior("constant(1)", class = "sd", group = "id", nlpar = "eta") +
  prior("normal(0, 3)", class = "sd", group = "item", nlpar = "eta") +
  prior("normal(0, 1)", class = "sd", group = "item", nlpar = "logalpha")
```

```
fit_va_2pl <- brm(
  formula = formula_va_2pl,
  data = VerbAgg,
  family = brmsfamily("bernoulli", "logit"),
  prior = prior_va_2pl,
)
```

Model Comparison via Leave-one-out Cross-Validation

The `loo` method implements approximate leave-one-out cross-validation via Pareto-Smoothed importance sampling:

```
loo_compare(loo(fit_va_1pl), loo(fit_va_2pl))
```

```
##             elpd_diff se_diff
## fit_va_2pl  0.0       0.0
## fit_va_1pl -3.0       2.4
```

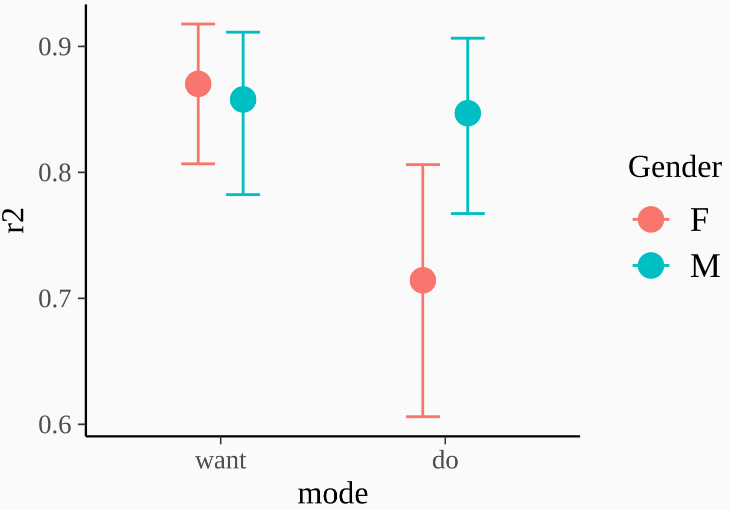
Adding Person and Item Covariates

```
formula_va_1pl_cov2 <- bf(
  r2 ~ btype + situ + mode * Anger + Gender +
    (0 + Gender | item) + (0 + mode | id)
)
```

```
fit_va_1pl_cov2 <- brm(
  formula = formula_va_1pl_cov2,
  data = VerbAgg,
  family = brmsfamily("bernoulli", "logit"),
  prior = prior_va_1pl
)
```

Visualizing Covariate Effects

```
conditional_effects(fit_va_1pl_cov2, "mode:Gender")
```

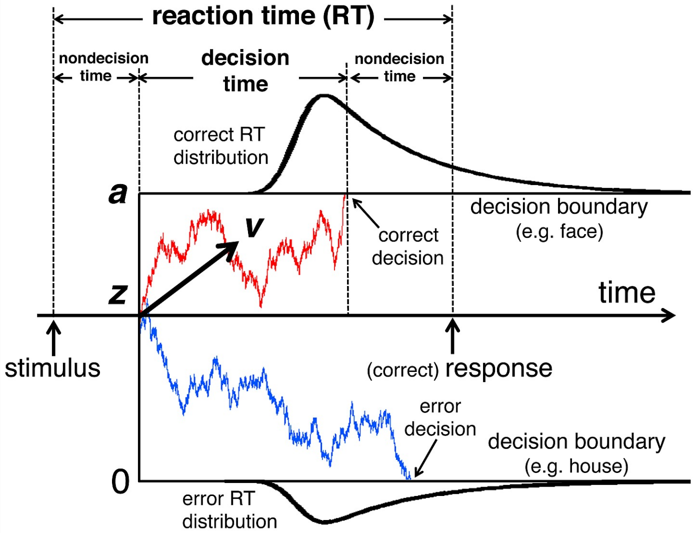


Case Study: The Rotation Data Set

```r
data("rotation", package = "diffIRT")
```

person	item	time	resp	rotate
1	1	4.444	1	150
1	10	5.447	1	100
1	2	2.328	1	50
1	3	3.408	1	100
1	4	5.134	1	150
1	5	2.653	1	50
1	6	2.607	1	100
1	7	3.126	1	150
1	8	2.869	1	50
1	9	3.271	1	150

Wiener Drift Diffusion Models

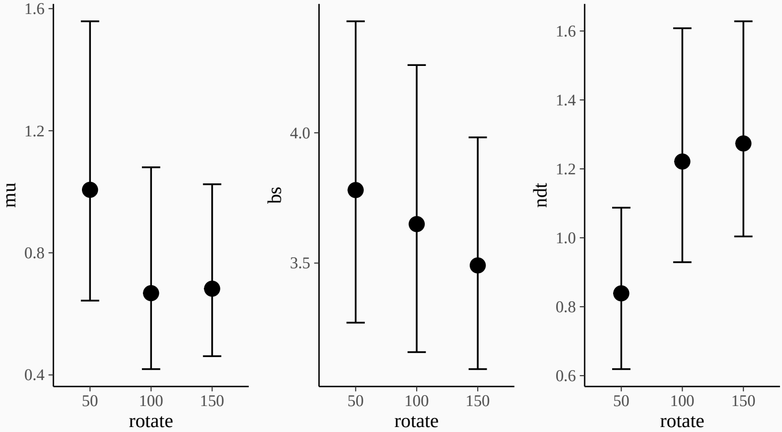


Fitting IRT Diffusion Models

```r
bform_drift1 <- bf(
  time | dec(resp) ~ rotate + (1 |p| person) + (1 |i| item),
  bs ~ rotate + (1 |p| person) + (1 |i| item),
  ndt ~ rotate + (1 |p| person) + (1 |i| item),
  bias = 0.5
)
```

```r
fit_drift1 <- brm(
  formula = bform_drift1,
  data = rotation,
  family = brmsfamily(
    "wiener", "log", link_bs = "log",
    link_ndt = "log"
  )
)
```

Diffusion Model: Results



Learn More about brms and Stan

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