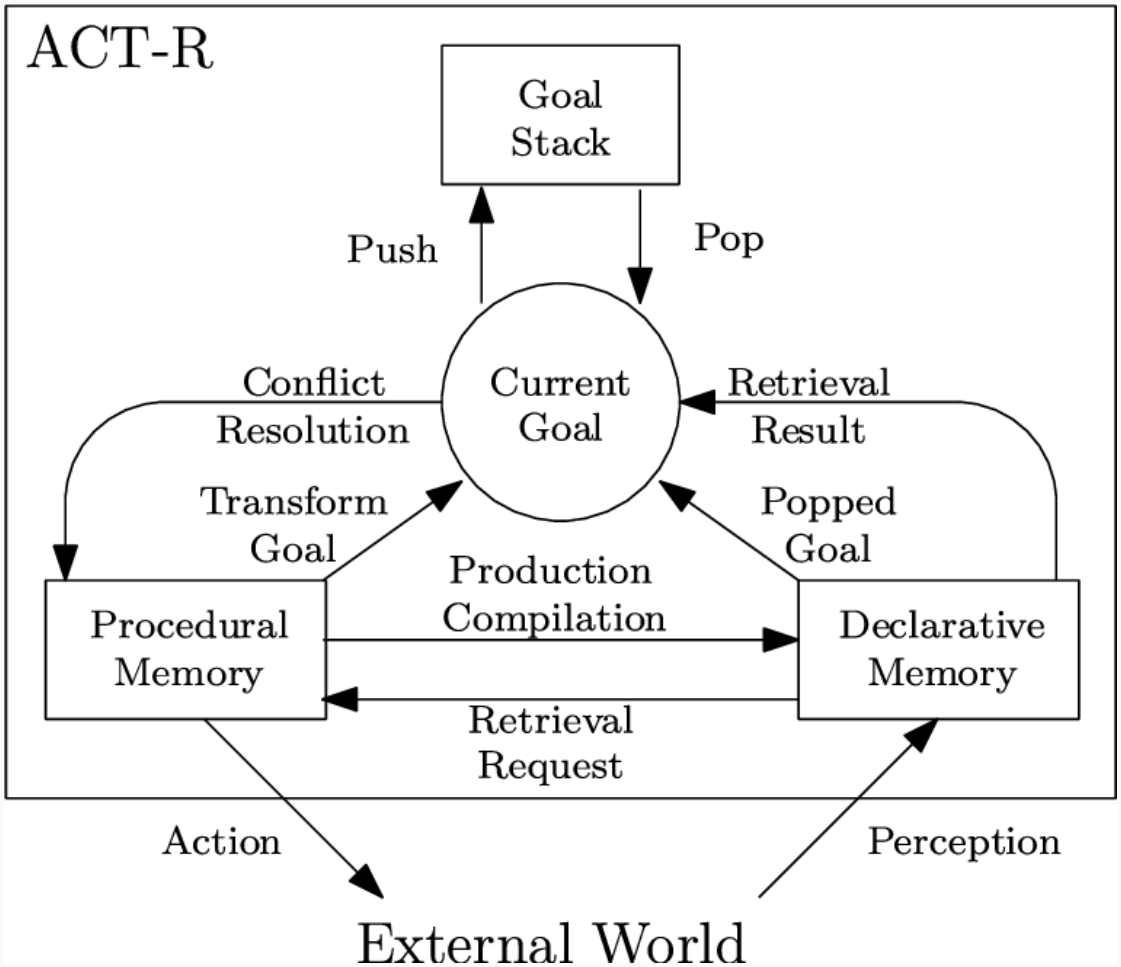
Bayesian distributional regression models for cognitive science

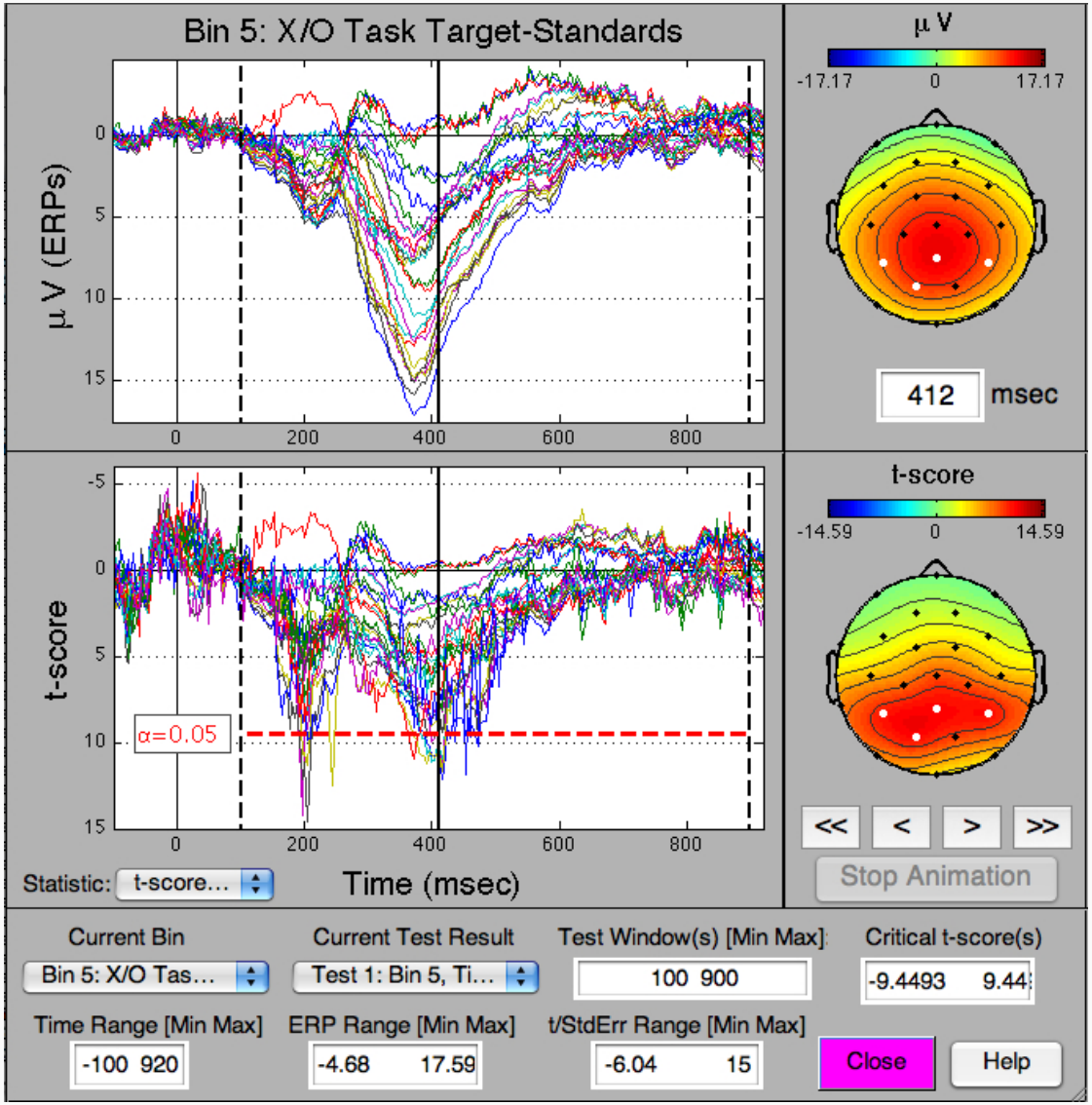
Paul Bürkner

Low-Level Perspectives on Cognitive Modeling



Source: Meyer B. (2008). *The effects of structural and group knowledge on complex problem solving performance.*

High-Level Perspectives on Cognitive Modeling



Source: Matlab Mass Univariate ERP Toolbox

Perspectives on Cognitive Modeling

Low-level generative models of neuro- or behavioral processes

...

...

(Abstraction level of the data and the generative process)

...

...

Mass univariate tests of neuro- or behavioral data

Perspectives on Cognitive Modeling

Low-level generative models of neuro- or behavioral processes

...

...

Distributional regression modeling

...

...

Mass univariate tests of neuro- or behavioral data

Bayesian (Estimation of) Cognitive Models

Whose learning and decision making are we considering?

- Those of participants: Bayesian models of cognition
- Those of researchers: Bayesian estimation of cognitive models

This talk focuses more on the second perspective

Cognitive-IRT Models

For IRT modeling we need:

- A set of parameters ξ_i for item (stimulus/input) 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)

So do you mean?

Bayesian Cognitive-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):

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

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

Distributional Models

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 stimulus and person parameters via response functions f_k:

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

Generalized Additive Models of Distributional Parameters

A lot of cognitive processes are of unknown non-linear form

At some point we have to settle for a convenient approximation

Generalized Additive Model of ψ_k:

$$\psi_k = f_k(\xi_i, \theta_p) = g_k\left(\sum_j h_{kj}(\xi_{ik}, \theta_{pk})\right)$$

Generalized Linear Model of ψ_k:

$$\psi_k = f_k(\xi_i, \theta_p) = g_k\left(\sum_{j_1} \xi_{ikj_2} x_{kj_1} + \sum_{j_2} \theta_{pkj_2} x_{kj_2}\right)$$

Can be realized as a one-step or two-step procedure

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:

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

What's a distributional parameter exactly?

Binary two-parameter logistic (2PL) model:

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

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

Alternative formulation:

$$y \sim \text{likelihood}(\psi_1, \psi_2) = f(\psi_1, \psi_2)^y \ (1 - f(\psi_1, \psi_2))^{1-y},$$

with

$$f(\psi_1, \psi_2) = \frac{\exp(\psi_2 \psi_1)}{1 + \exp(\psi_2 \psi_1)}$$

$$\psi_1 = \theta_p + \xi_i \qquad \psi_2 = \alpha_i$$

Making it more complicated

Binary four-parameter logistic (4PL) Model:

$$\begin{aligned} y &\sim \text{likelihood}(\psi_1, \psi_2, \psi_3, \psi_4) \\ &= f(\psi_1, \psi_2, \psi_3, \psi_4)^y \, (1 - f(\psi_1, \psi_2, \psi_3, \psi_4))^{1-y}, \end{aligned}$$

with

$$f(\psi_1, \psi_2, \psi_3, \psi_4) = \psi_3 + (1 - \psi_3 - \psi_4) \, \frac{\exp(\psi_2 \psi_1)}{1 + \exp(\psi_2 \psi_1)}$$

- location ψ_1
- scale ψ_2
- lower asymptote ψ_3 (guessing)
- upper asymptote ψ_4 (lapse)

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:

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

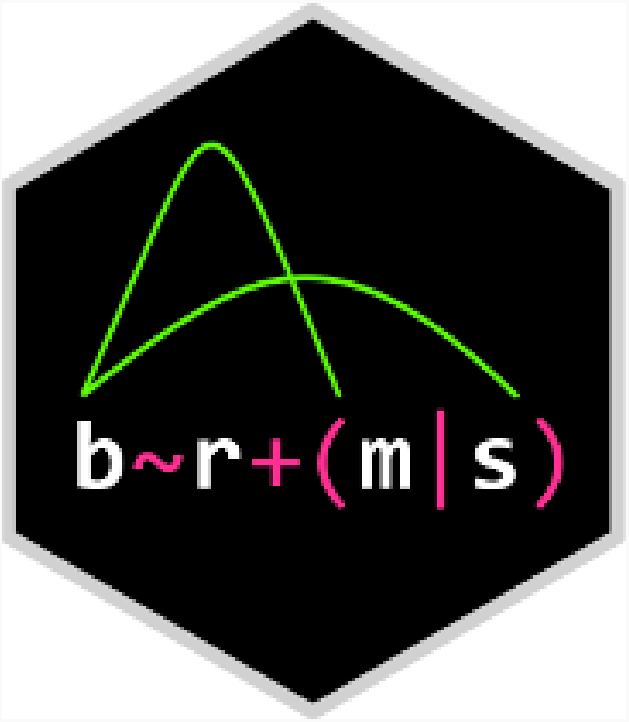
Decompose the covariance matrix Σ_ξ as:

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

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

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

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 Rotation Data Set

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

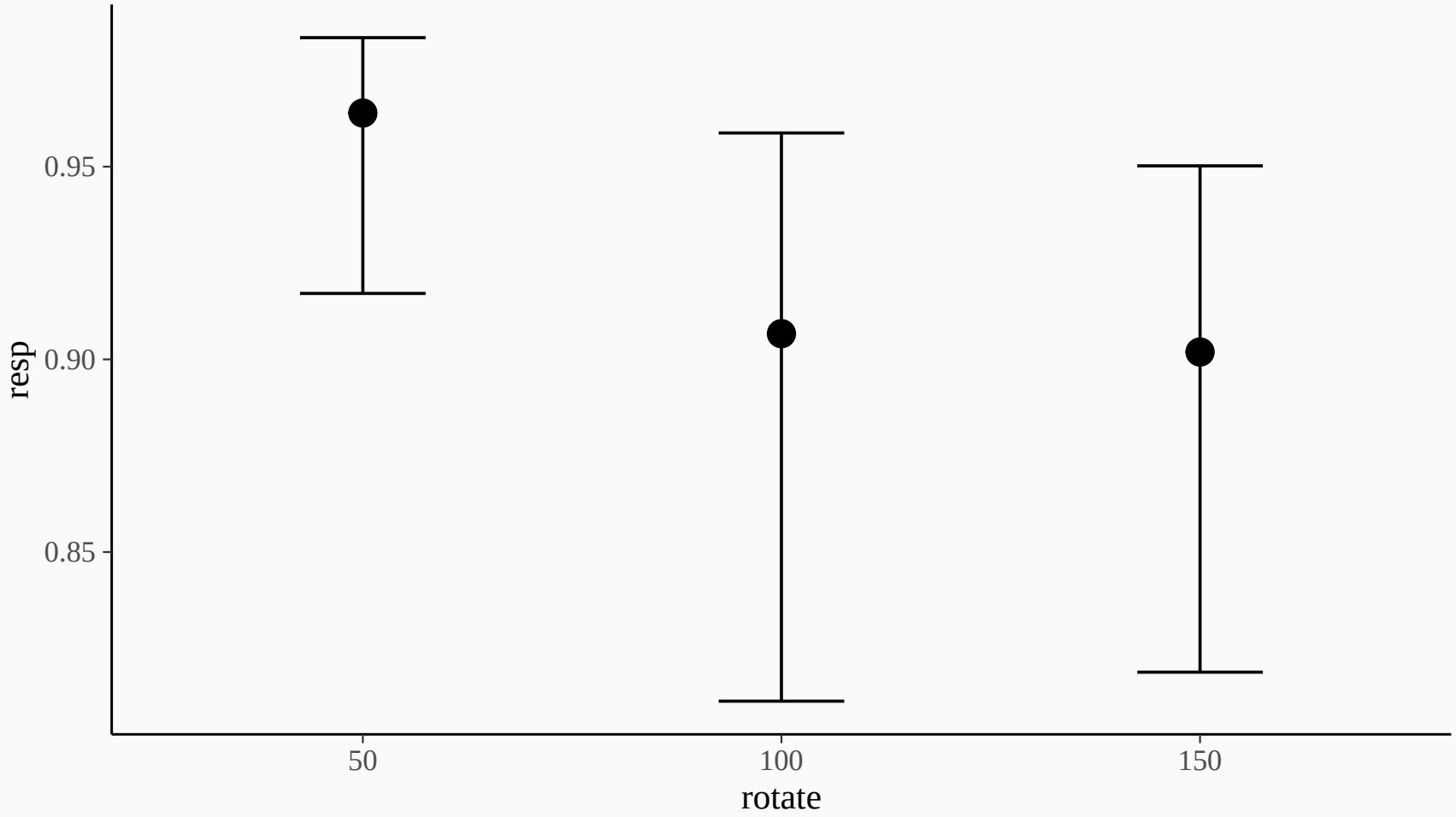
person	item	time	resp	rotate
1	1	4.444	1	150
1	10	5.447	1	100
1	2	2.328	1	50
1	3	3.408	1	100
1	4	5.134	1	150
1	5	2.653	1	50
1	6	2.607	1	100
1	7	3.126	1	150
1	8	2.869	1	50
1	9	3.271	1	150

Models of Binary Decisions: 1PL Model

```
bform_binary1 <- bf(resp ~ rotate + (1 | person) + (1 | item))
```

```
fit_binary1 <- brm(
  formula = bform_binary1,
  data = rotation,
  family = brmsfamily("bernoulli", "logit")
)
```

1PL Model: Results

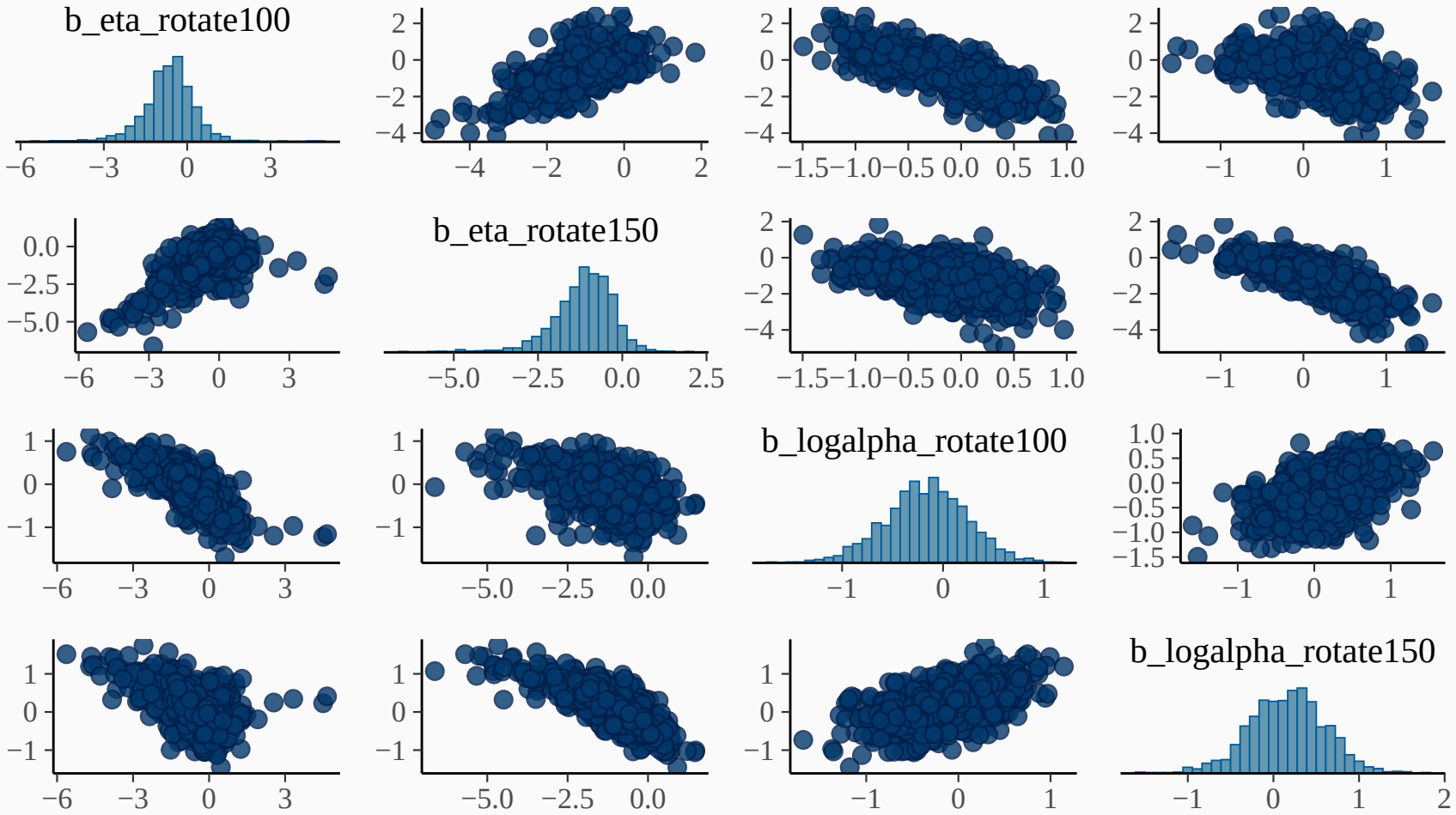


Models of Binary Decisions: 2PL Model

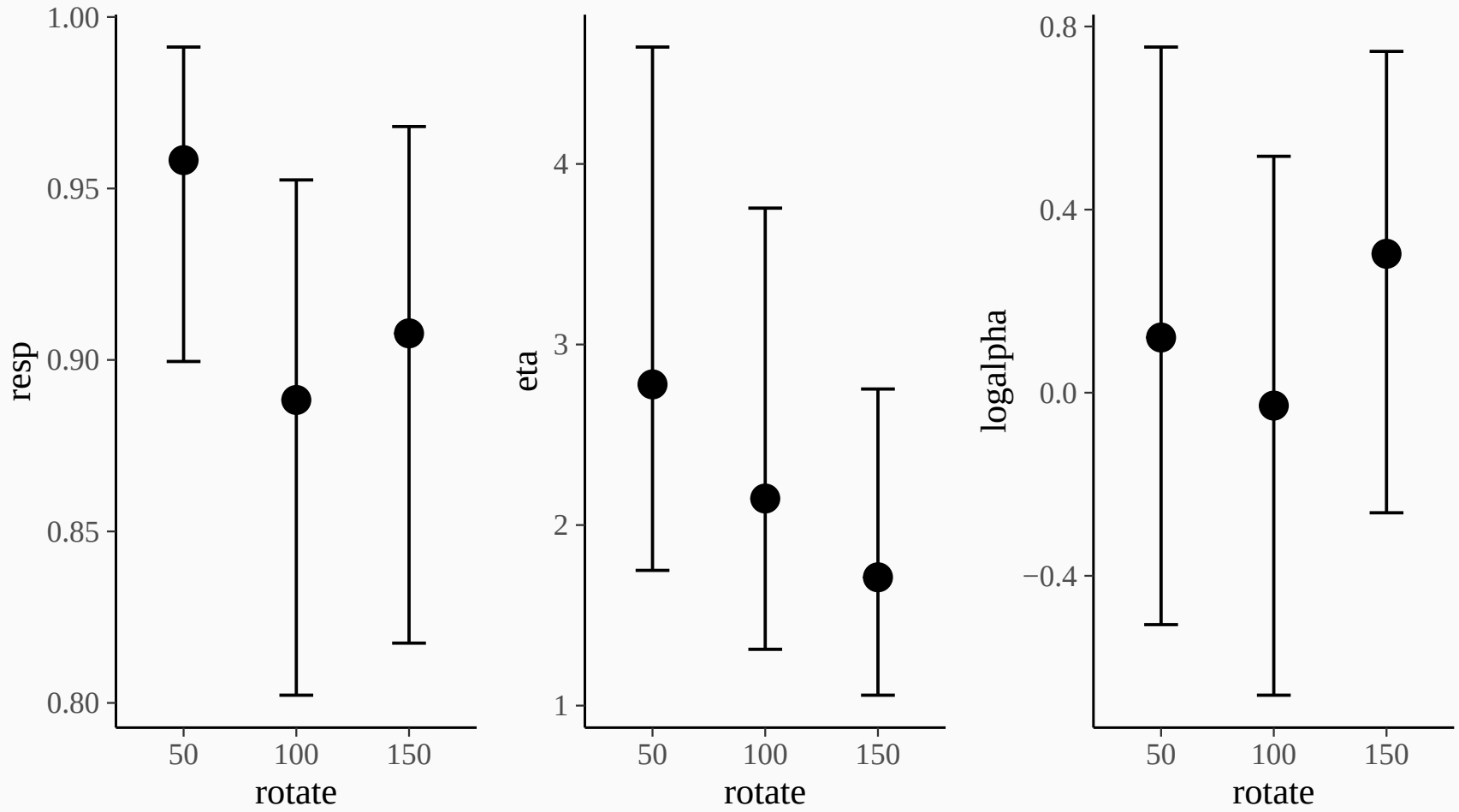
```
bform_binary2 <- bf(
  resp ~ exp(logalpha) * eta,
  eta ~ rotate + (1 |i| item) + (1 | person),
  logalpha ~ rotate + (1 |i| item),
  nl = TRUE
)
```

```
fit_binary2 <- brm(
  formula = bform_binary2,
  data = rotation,
  family = brmsfamily("bernoulli", "logit"),
  prior = prior_binary2
)
```

2PL Model: Pairs Plots



2PL Model: Results



Model Comparison

```
loo_compare(loo(fit_binary1), loo(fit_binary2))
```

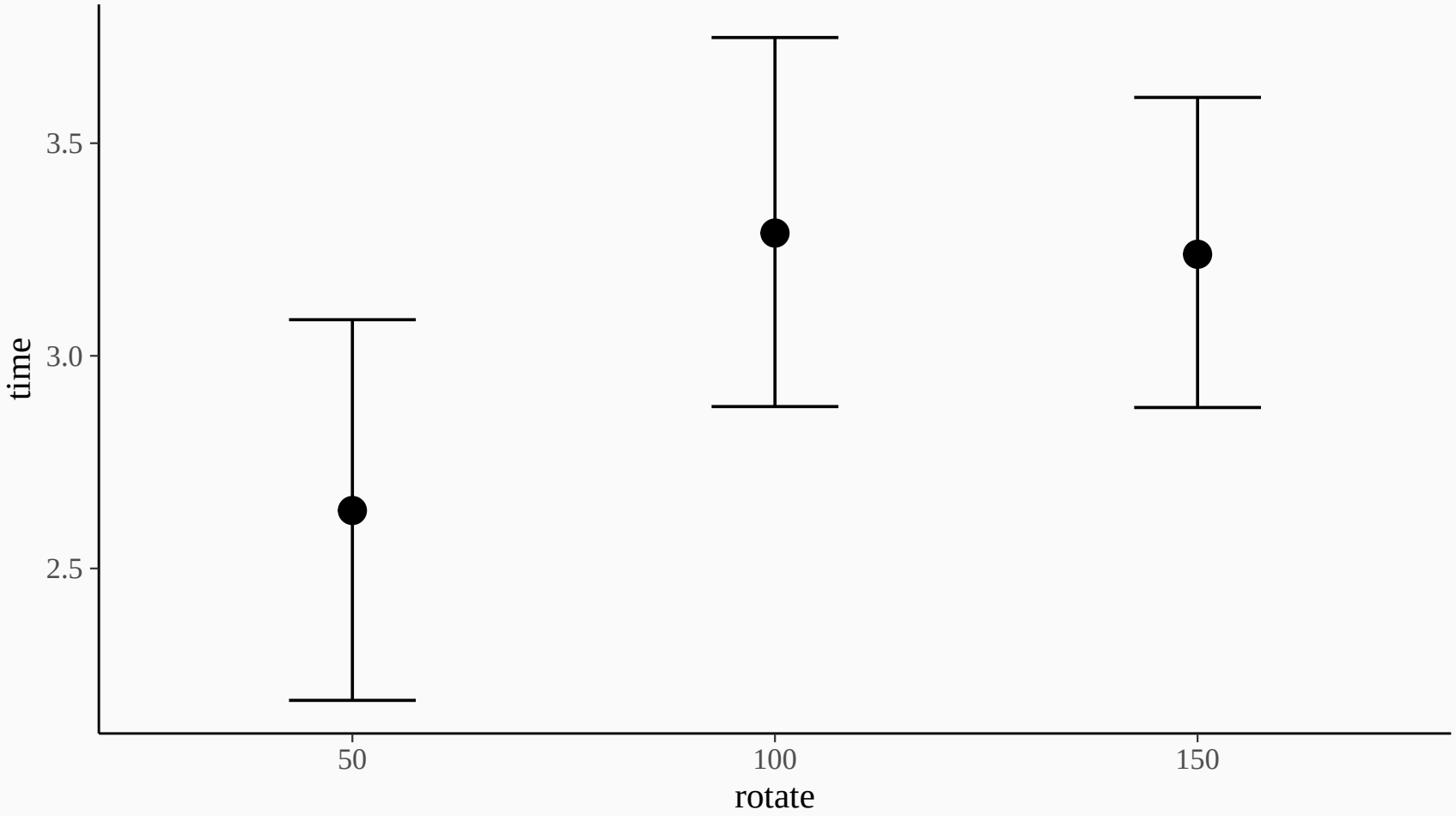
```
##             elpd_diff se_diff
## fit_binary1  0.0       0.0
## fit_binary2 -4.5       1.8
```

Simple Model of Reaction Times

```
bform_time1 <- bf(time ~ rotate + (1 | person) + (1 | item))
```

```
fit_time1 <- brm(
  formula = bform_time1,
  data = rotation,
  family = brmsfamily("exgaussian")
)
```

Simple ExGaussian Model: Results

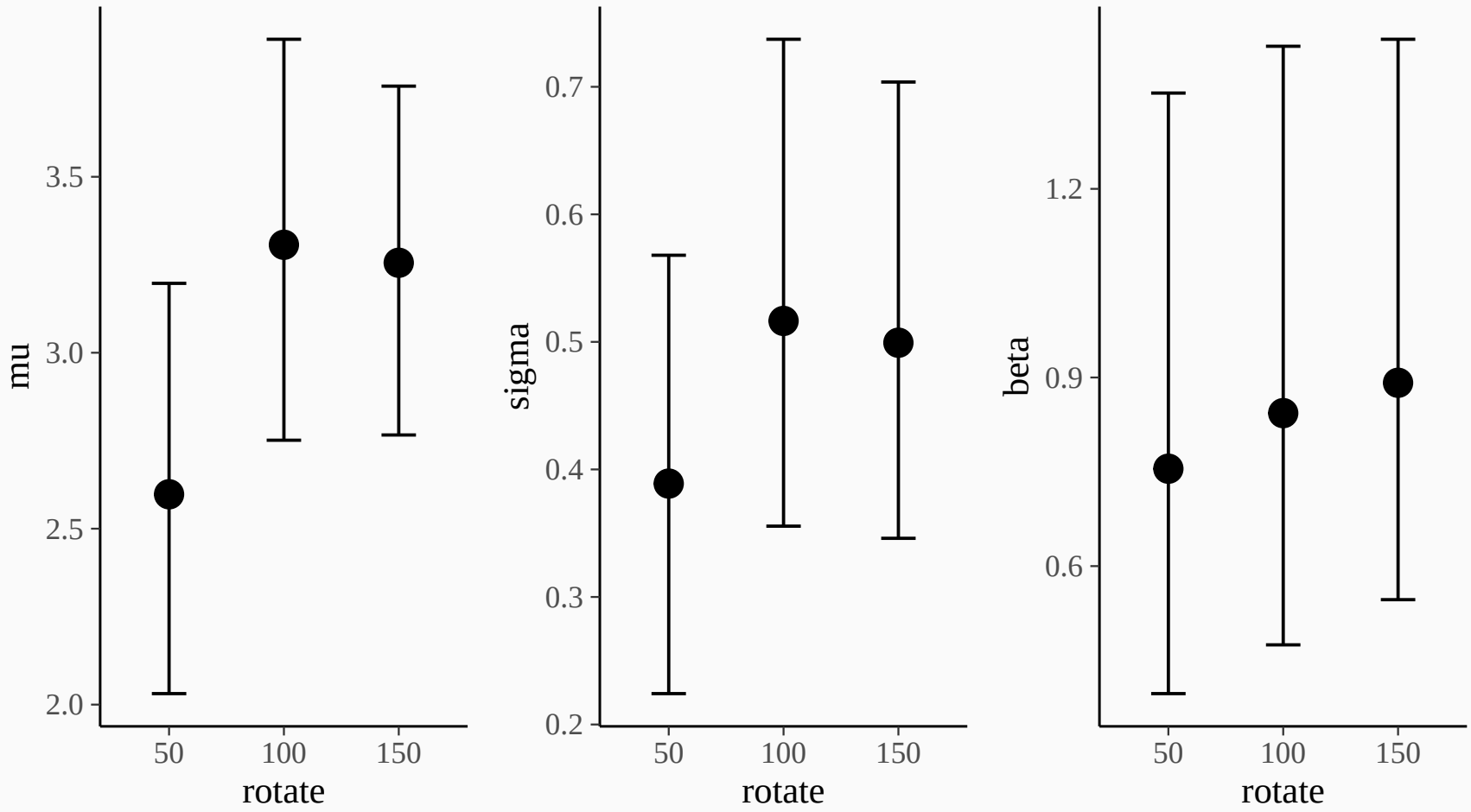


Distributional Model of Reaction Times

```
bform_time2 <- bf(
  time ~ rotate + (1 | person) + (1 | item),
  sigma ~ rotate + (1 | person) + (1 | item),
  beta ~ rotate + (1 | person) + (1 | item)
)
```

```
fit_time2 <- brm(
  formula = bform_time2,
  data = rotation,
  family = brmsfamily("exgaussian")
)
```

Distributional ExGaussian Model: Results

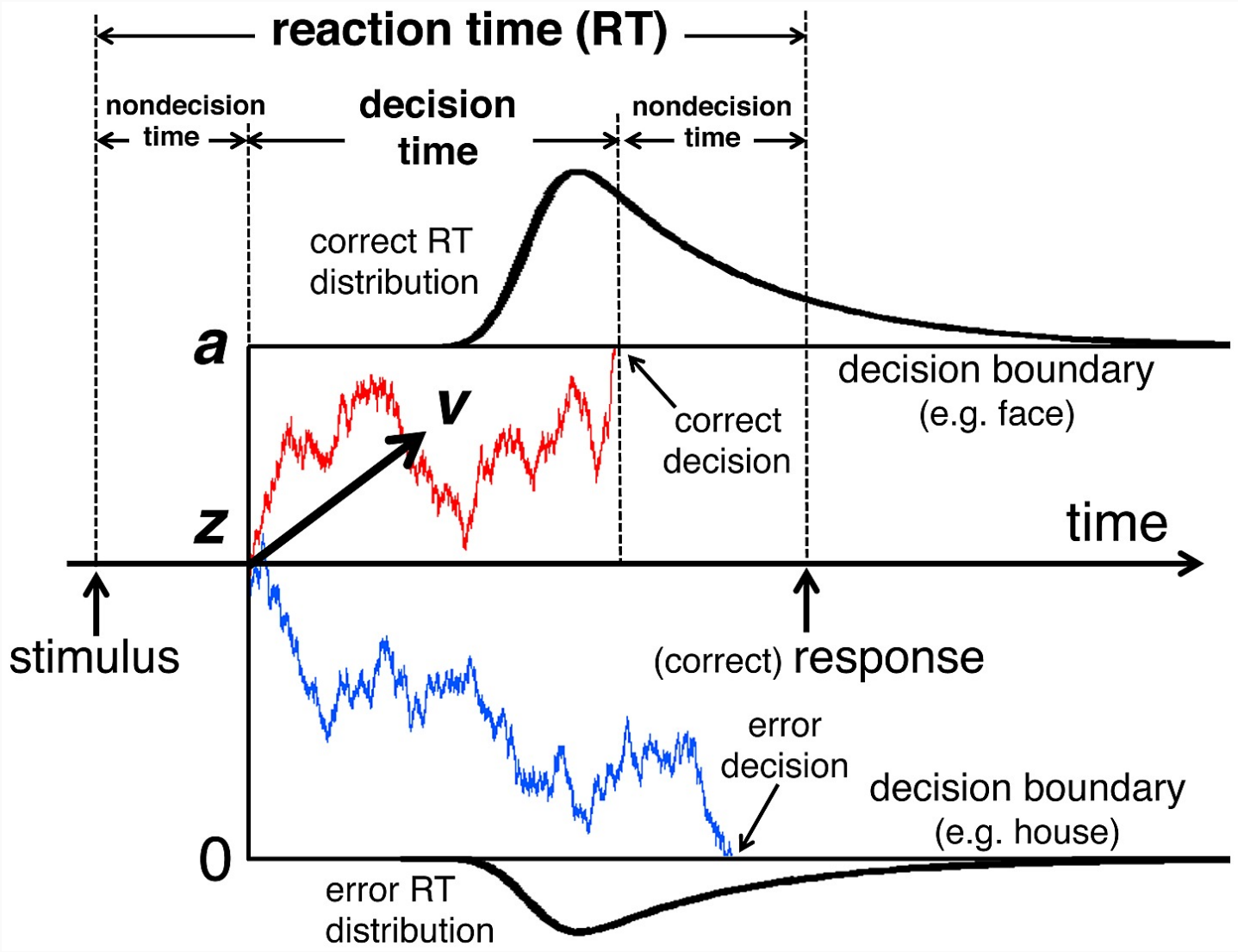


Model Comparison

```
loo_compare(loo(fit_time1), loo(fit_time2))
```

```
##            elpd_diff se_diff
## fit_time2   0.0       0.0
## fit_time1 -64.1      12.1
```

Wiener Drift Diffusion Models

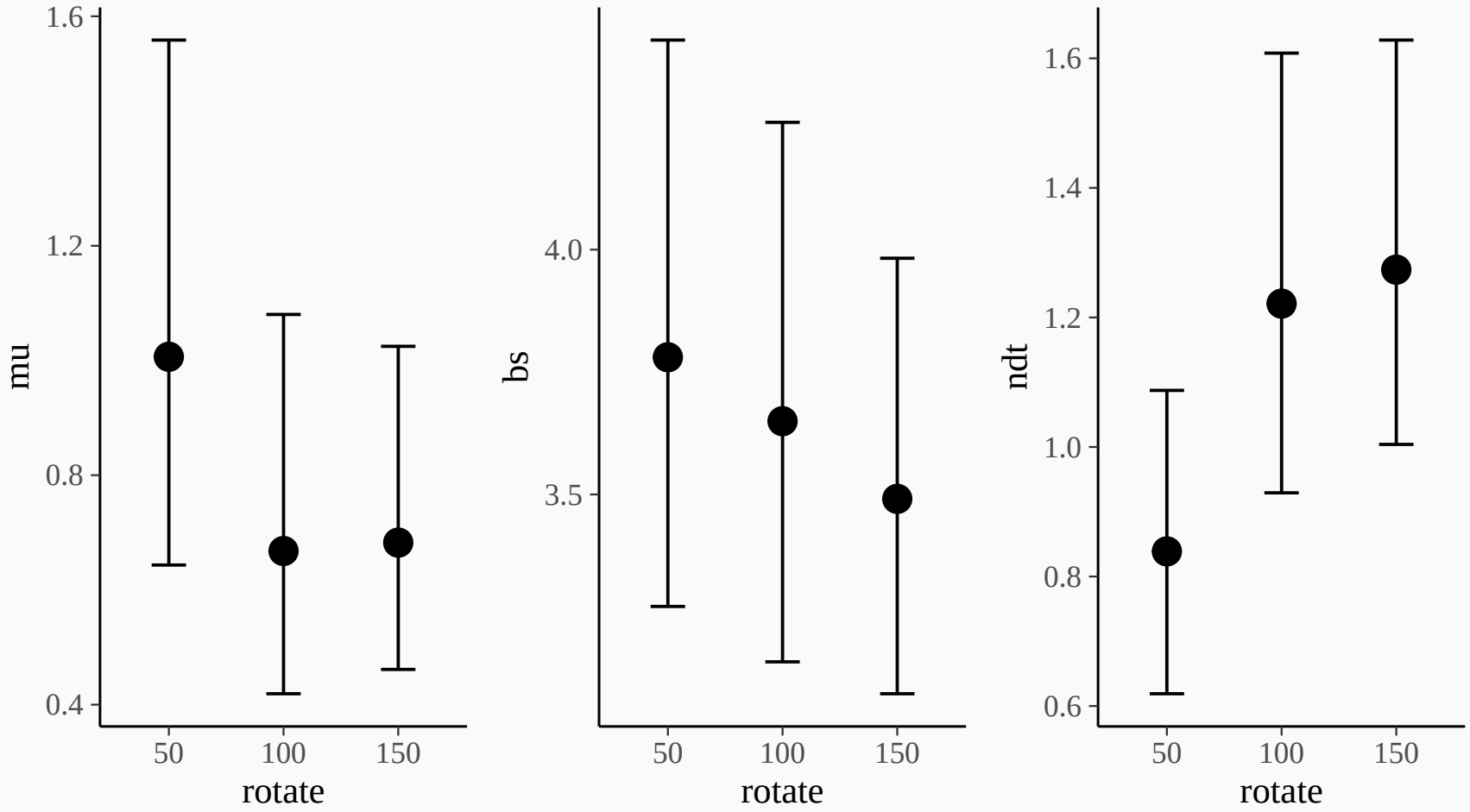


Fitting Diffusion Models

```
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
)
```

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

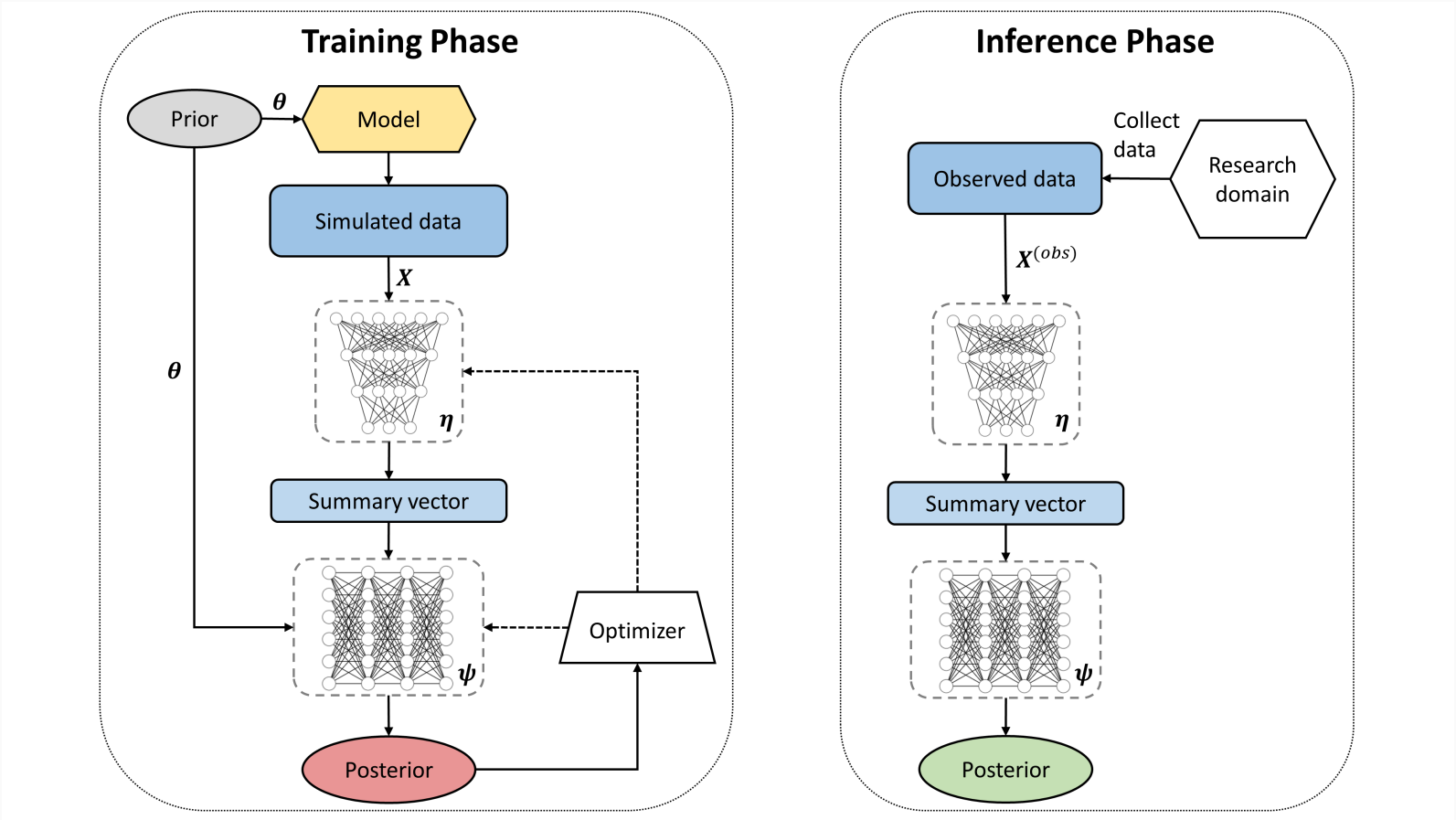
Diffusion Model: Results



Learn More about brms and Stan

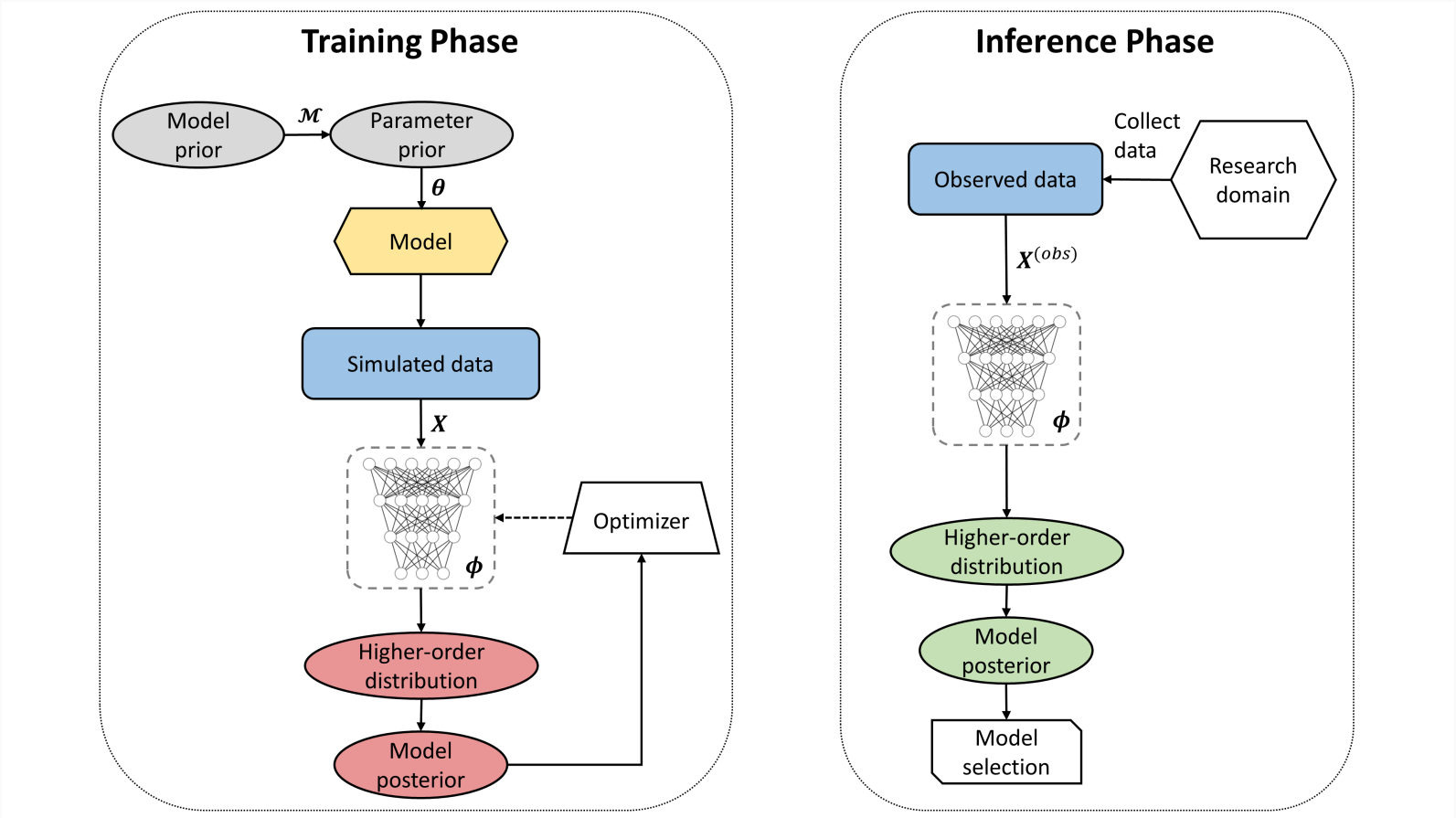
- Help within R: `help("brms")`
- Overview of vignettes: `vignette(package = "brms")`
- List of all methods: `methods(class = "brmsfit")`
- Website of brms: https://github.com/paul-buerkner/brms
- Website of Stan: http://mc-stan.org/
- Contact me: `paul.buerkner@gmail.com`
- Twitter: @paulbuerkner

Outlook: Simulation Based Inference (Estimation)



Source: Radev, S. T., Mertens, U. K., Voss, A., Ardizzone, L., & Köthe, U. (2020). BayesFlow: Learning complex stochastic models with invertible neural networks. *IEEE Transactions on Neural Networks and Learning Systems.*

Outlook: Simulation Based Inference (Model Comparison)



Source: Radev, S. T., D'Alessandro, M., Mertens, U. K., Voss, A., Köthe, U., & Bürkner, P. C. (2020). Amortized Bayesian model comparison with evidential deep learning. *arXiv preprint*.

References

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

Bürkner P. C. (2017). brms: An R Package for Bayesian Multilevel Models using Stan. *Journal of Statistical Software.*

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., ... & Riddell, A. Stan: A Probabilistic Programming Language. *Journal of Statistical Software.*

Radev S., Wieschen E. M., Voss A., & Bürkner P. C. (2020). Amortized Bayesian Inference for Models of Cognition. *International Conference on Cognitive Modelling (ICCM) Conference Proceedings.*