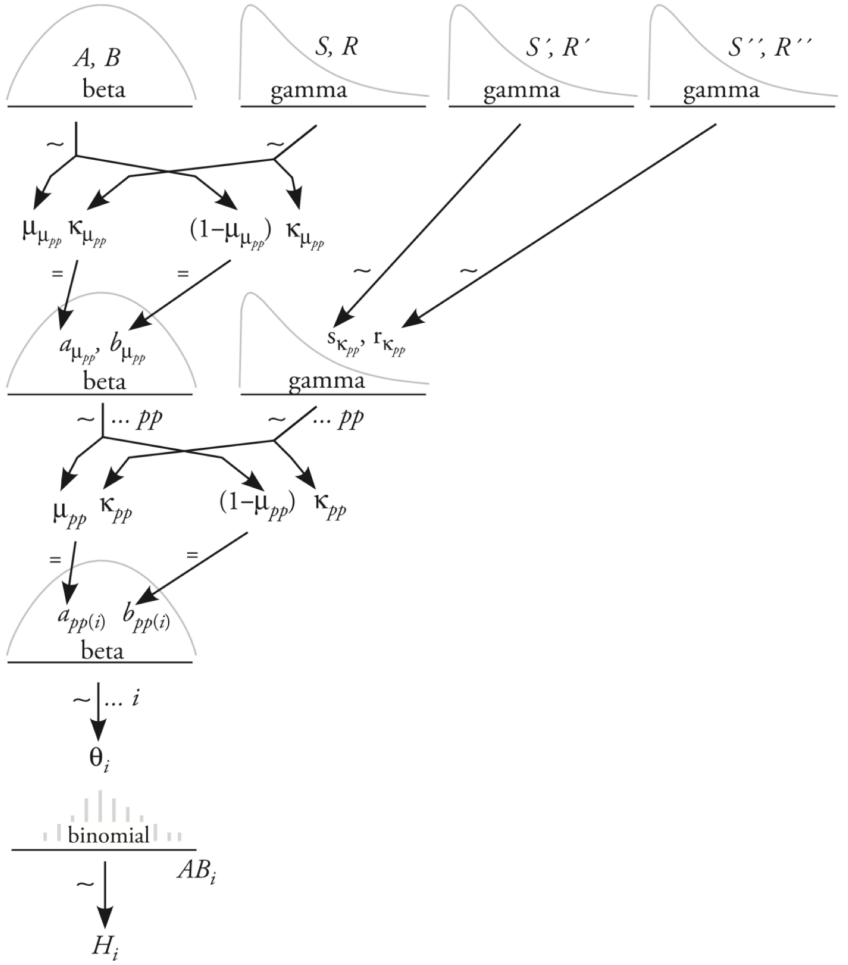
# Multilevel Modeling

## in a

# Bayesian Framework

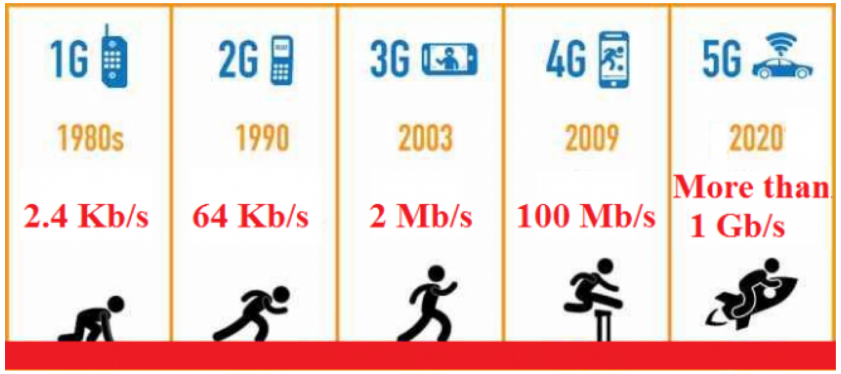
**Clinton Brownley**



# We care about groups (aka clusters)

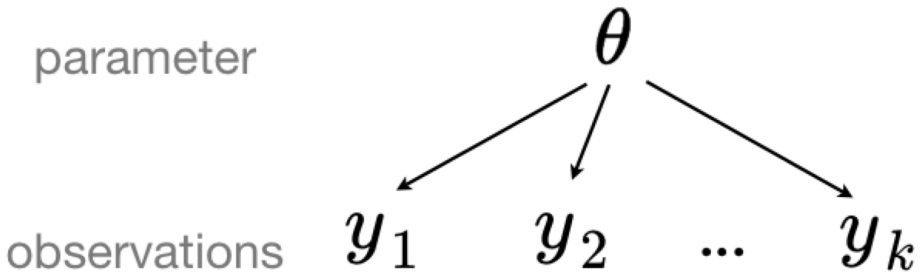




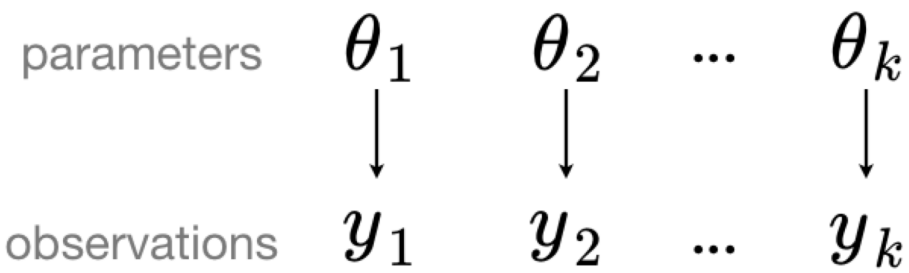


# Two unsatisfactory options for handling groups

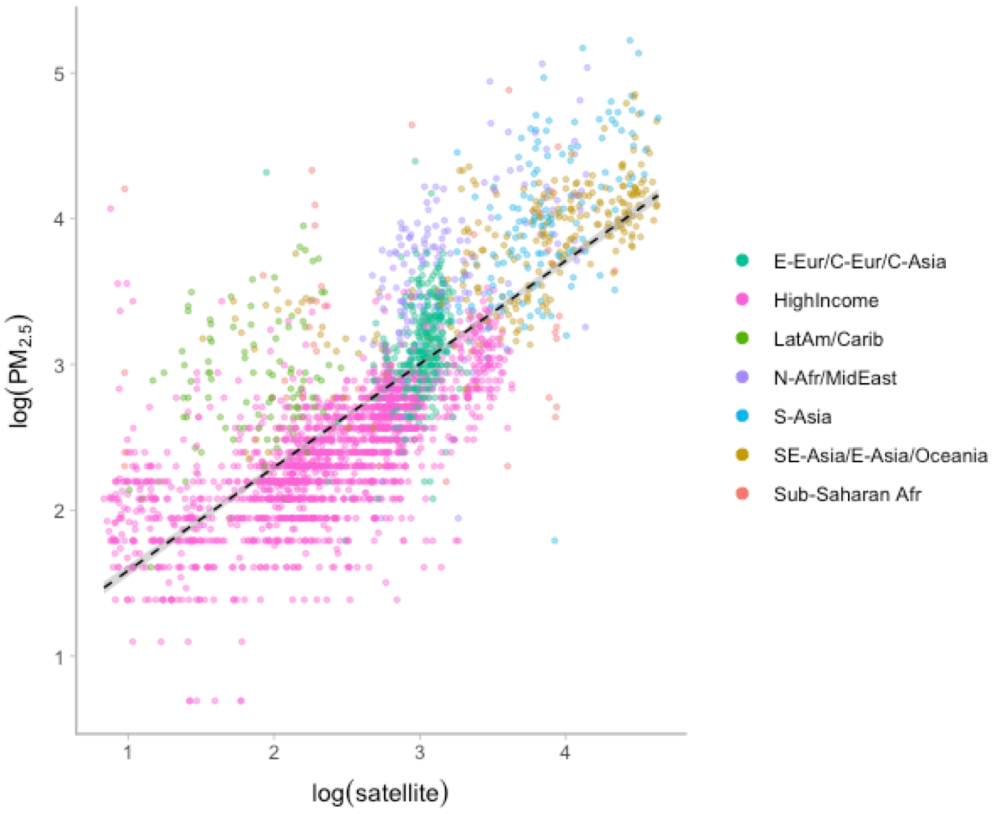
- Complete pooling

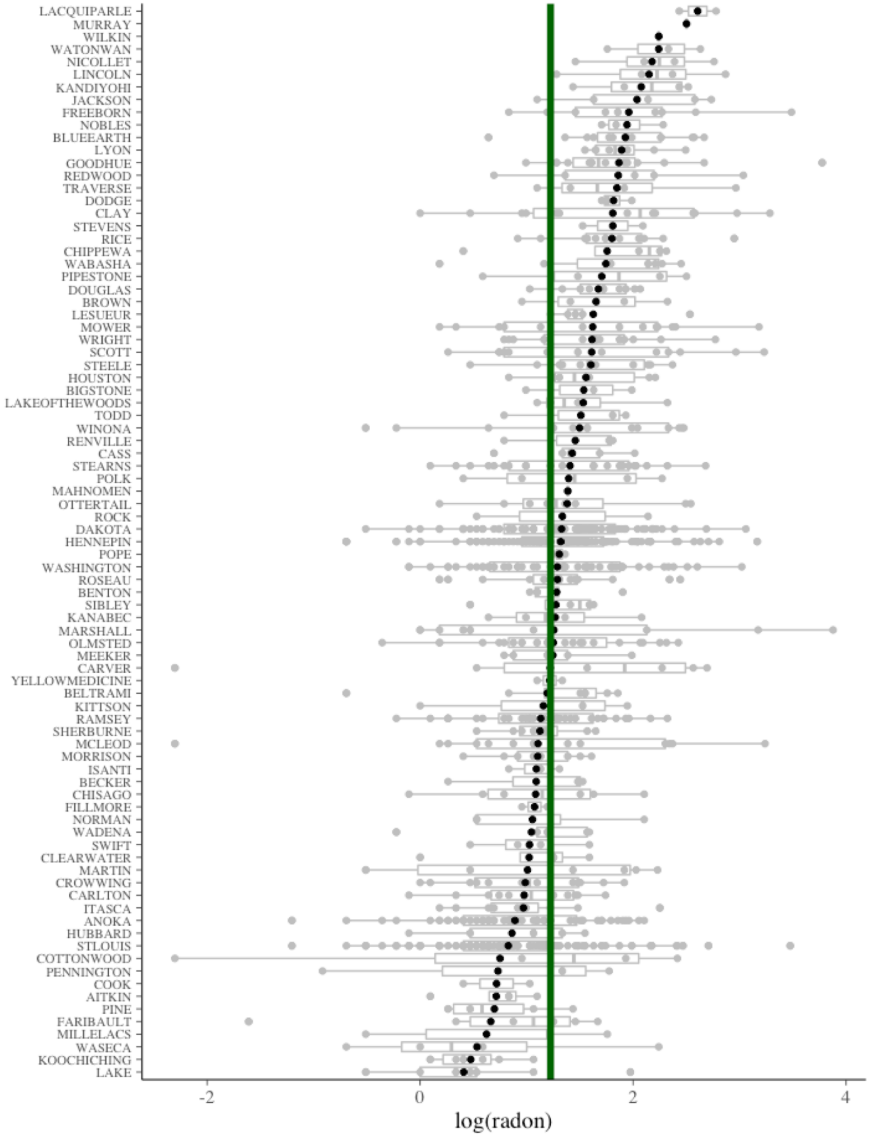



- No pooling



# Complete Pooling ignores groups

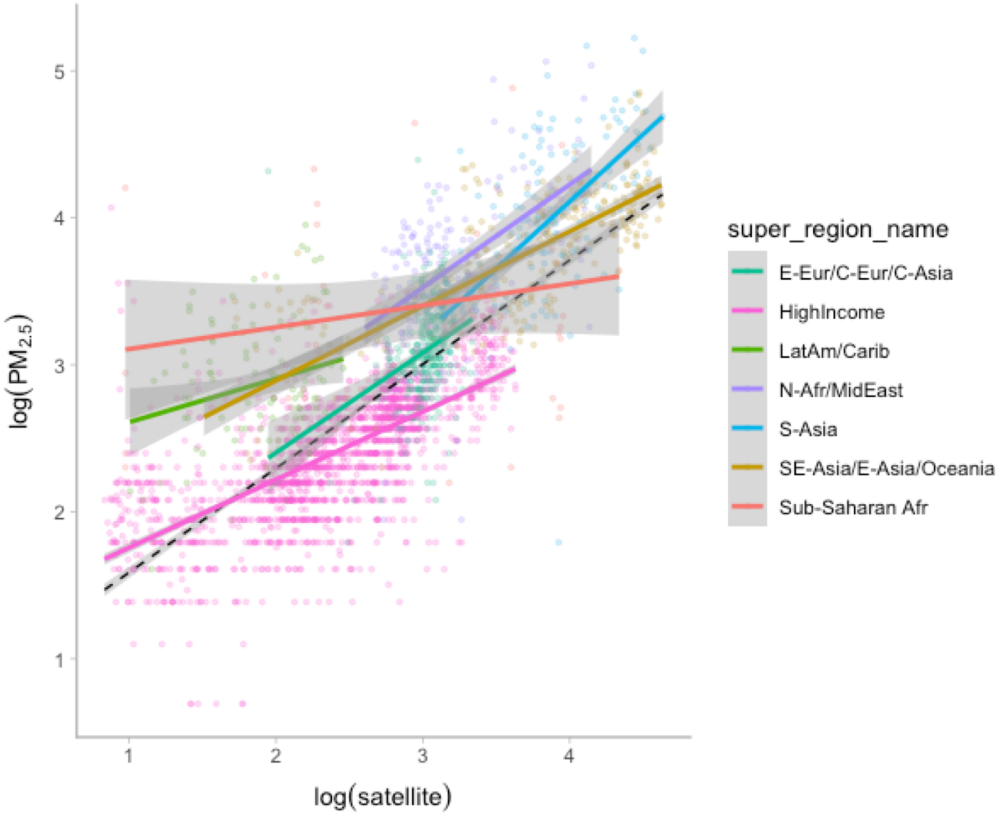


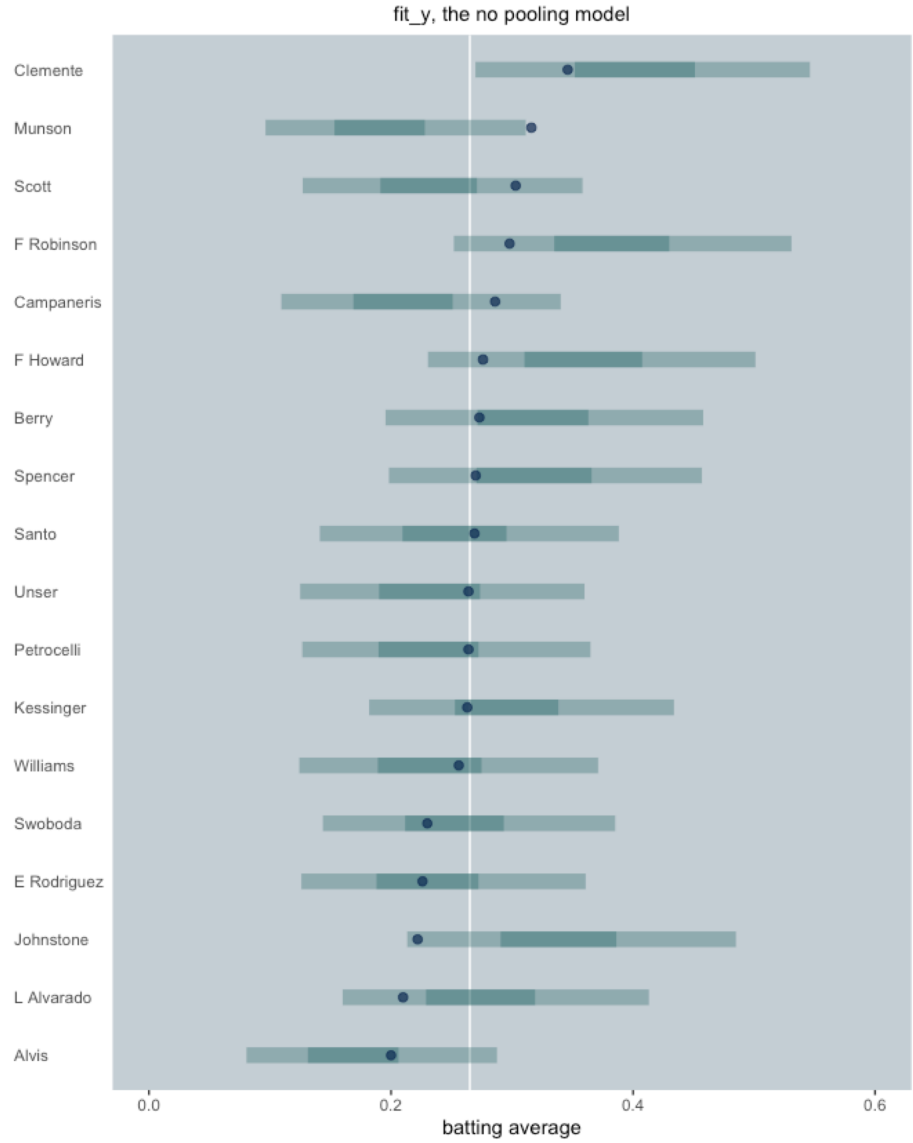



# Complete Pooling: Weaknesses

- Assumes all group members are identical (i.e. variation is zero)
- Grand mean is unlikely to match any particular group member mean
- Underfits the data

# No Pooling ignores commonality



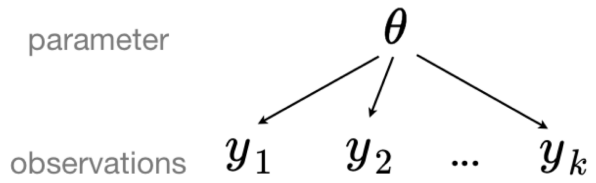



# No Pooling: Weaknesses

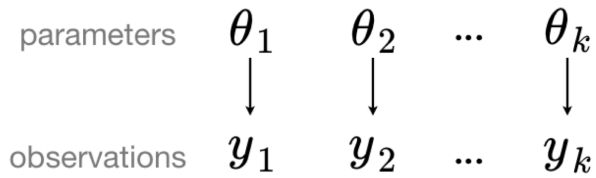
- Assumes all group members are different (i.e. variation is infinite)
- Assumes learning about a group member tells you nothing about the others
- Less data in each group → larger standard errors for each
- Overfits the data

# Partial Pooling: A Compromise

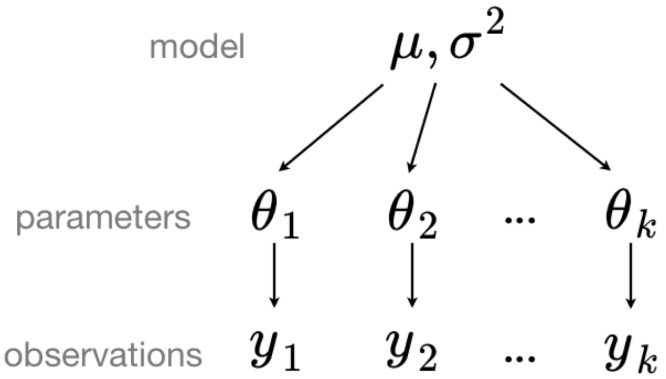
- Complete pooling



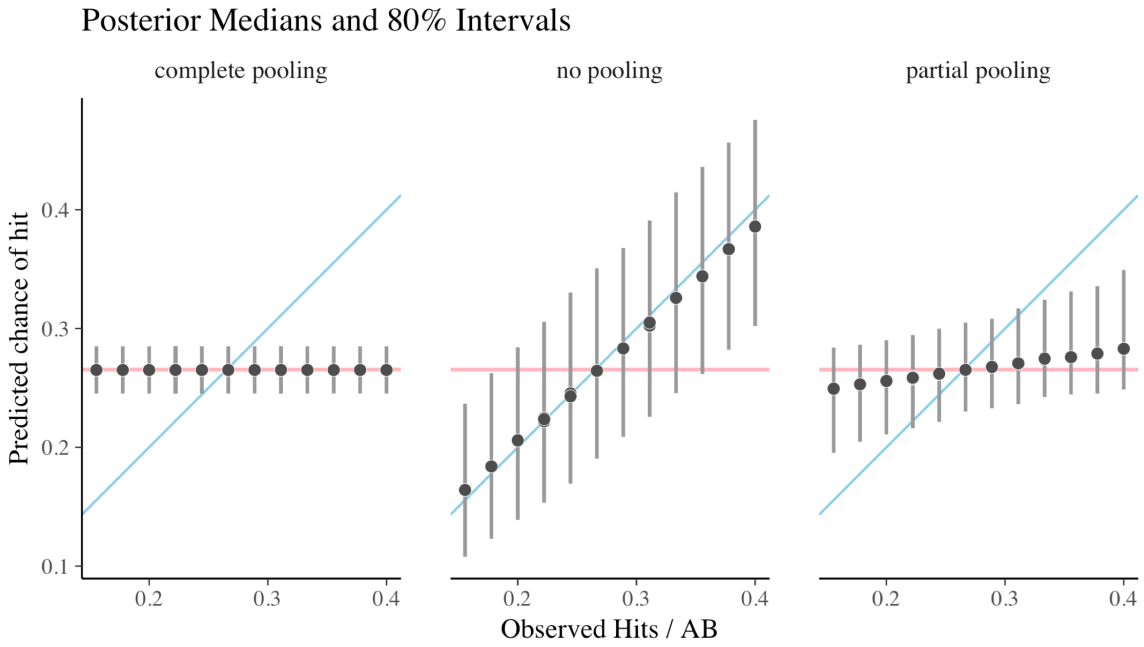

- No pooling

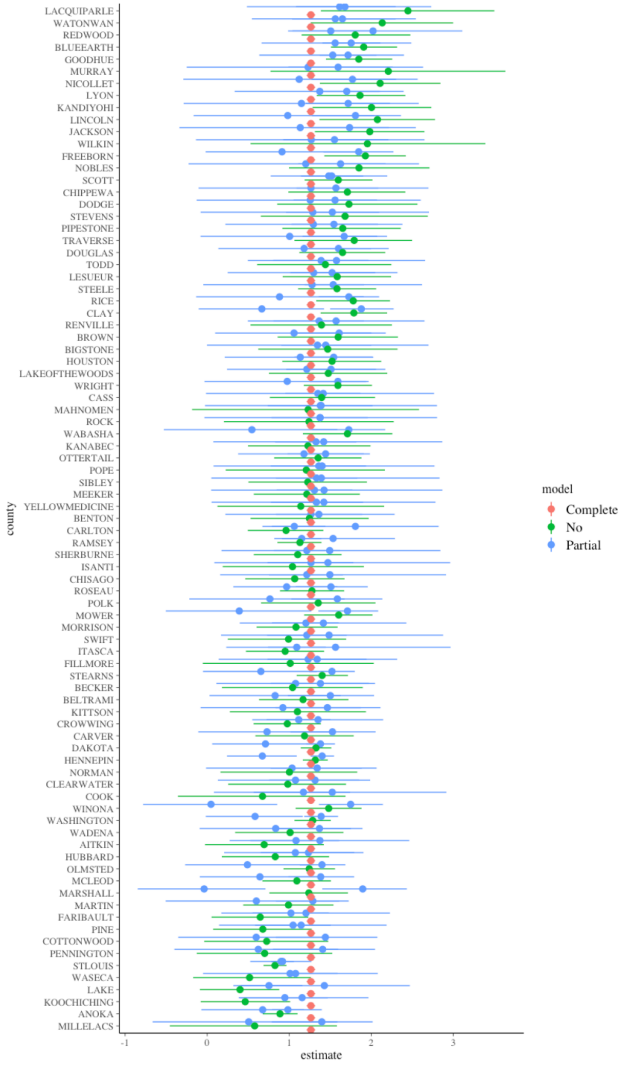



- Partial pooling



# Partial Pooling compromises between Complete and No Pooling

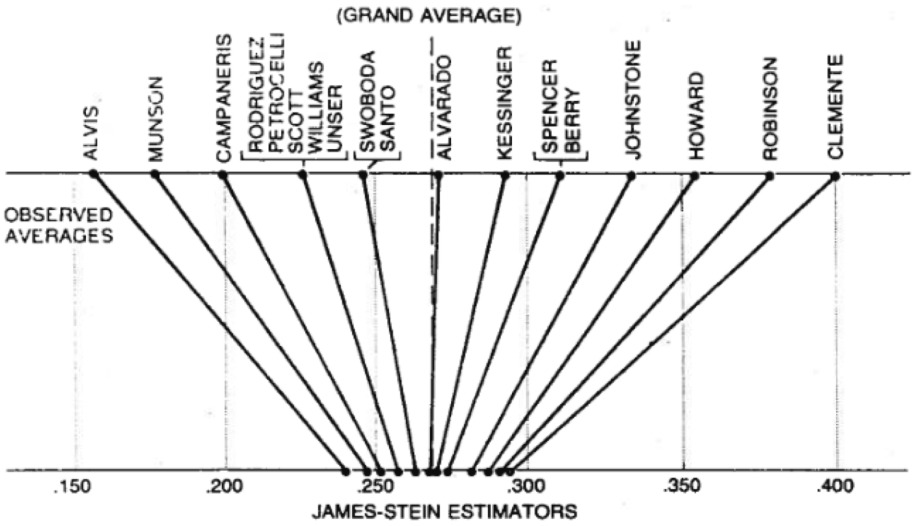


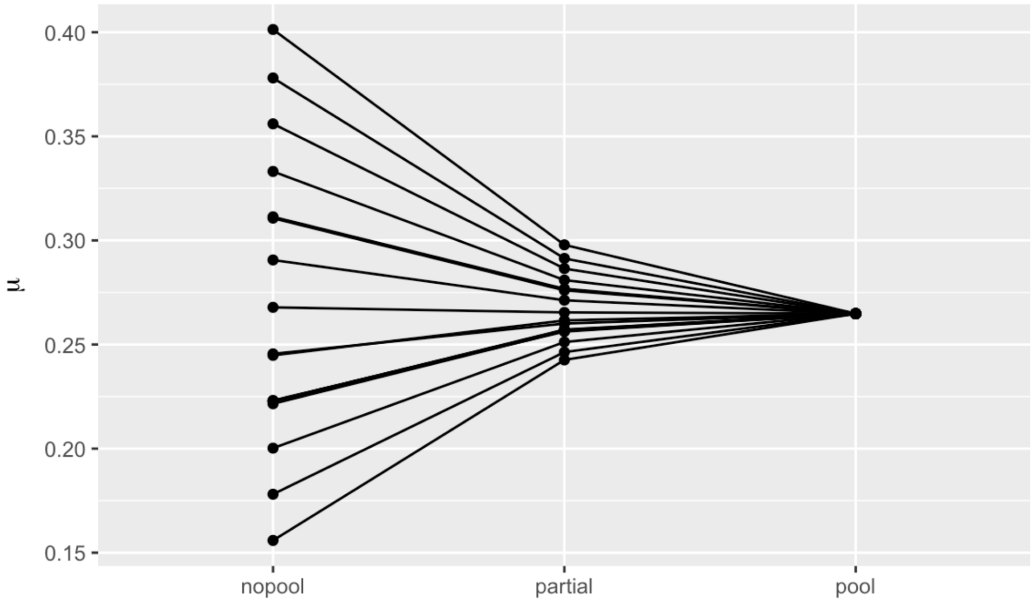
# Stein's Paradox in Statistics

*The best guess about the future is usually obtained by computing the average of past events. Stein's paradox defines circumstances in which there are estimators better than the arithmetic average*

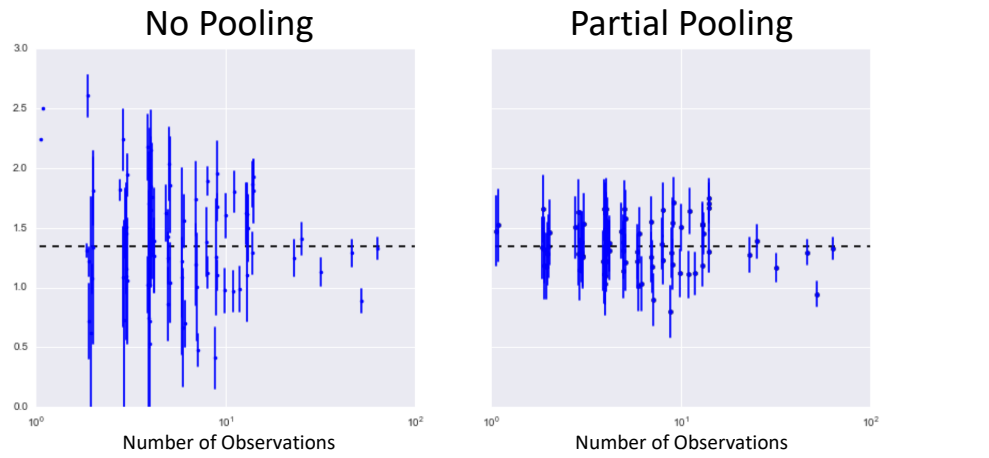
by Bradley Efron and Carl Morris



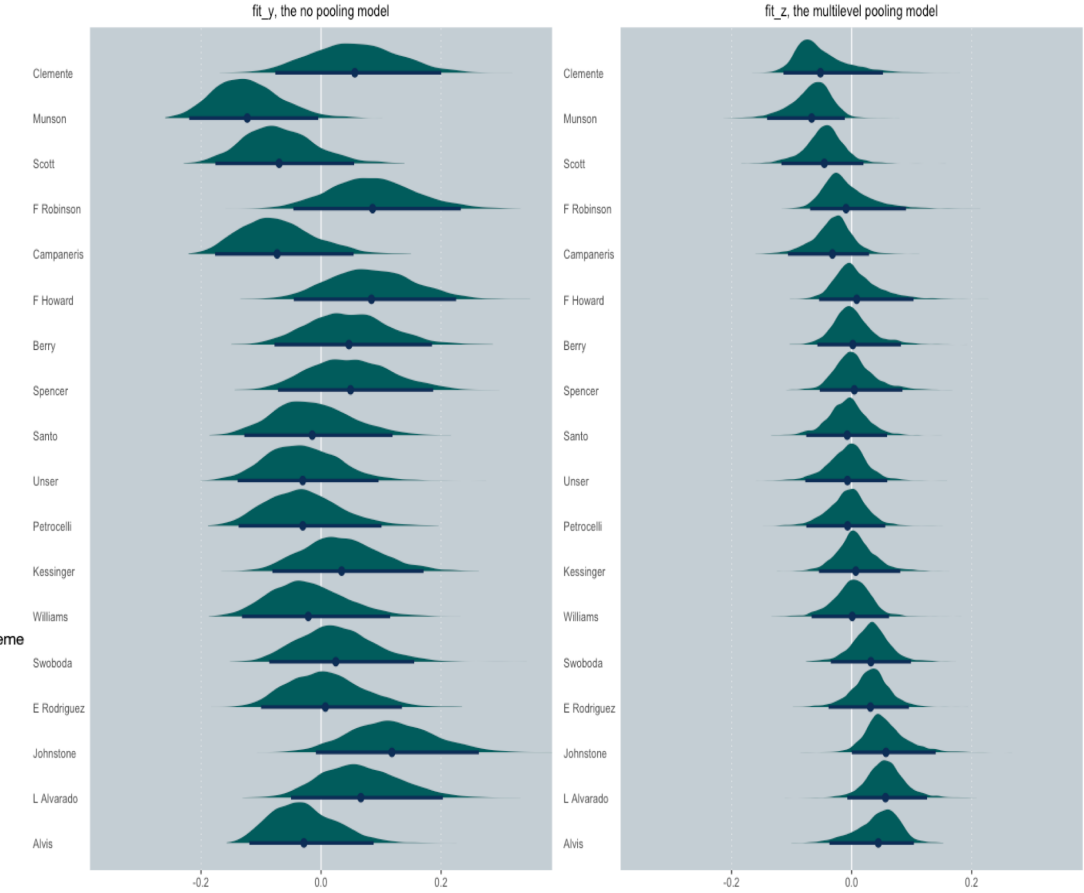

**JAMES-STEIN ESTIMATORS for the 18 baseball players were calculated by "shrinking" the individual batting averages toward the overall "average of the averages." In this case the grand average is .265 and each of the averages is shrunk about 80 percent of the distance to this value. Thus the theorem on which Stein's method is based asserts that the true batting abilities are more tightly clustered than the preliminary batting averages would seem to suggest they are.**



# Partial Pooling generates more accurate, precise predictions

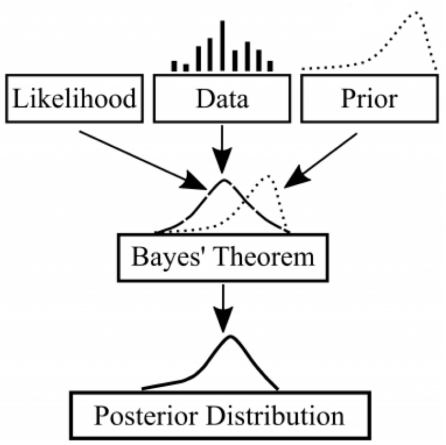


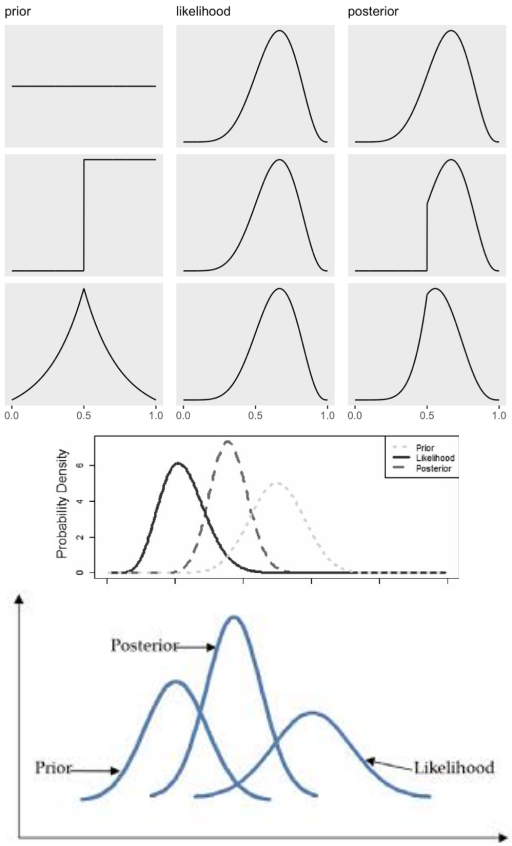

Notice the difference between the unpooled and partially-pooled estimates, particularly at smaller sample sizes. The former are both more extreme imprecise.

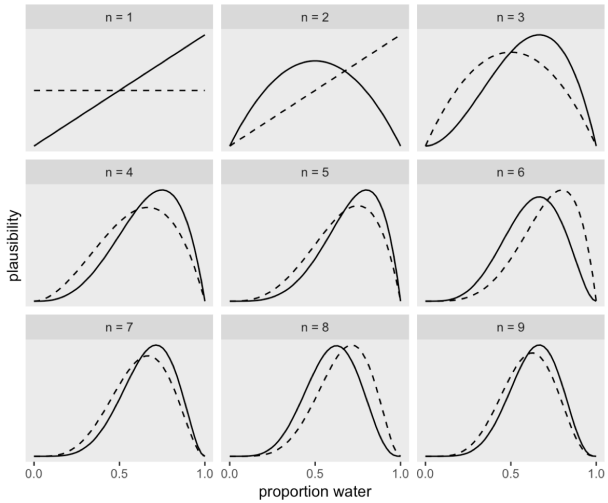


# Bayesian Framework

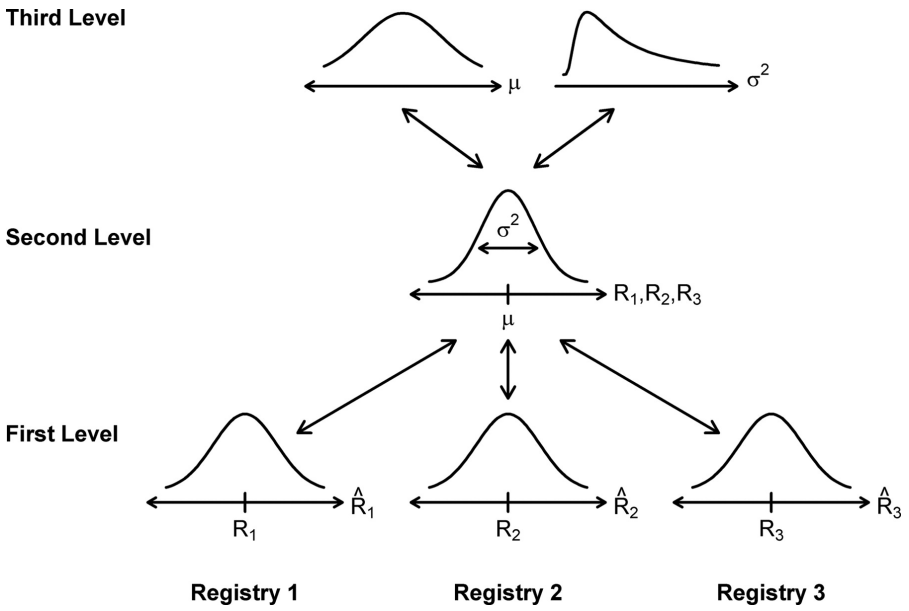
$$P(\Theta|data) \propto P(data|\Theta) \times P(\Theta)$$

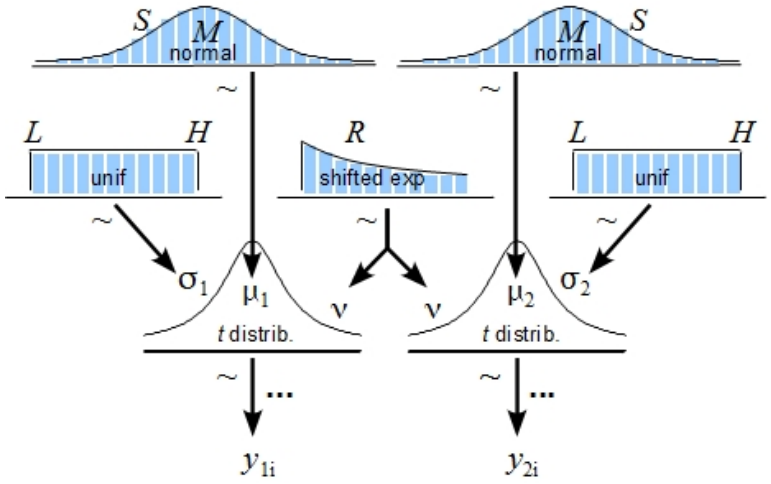


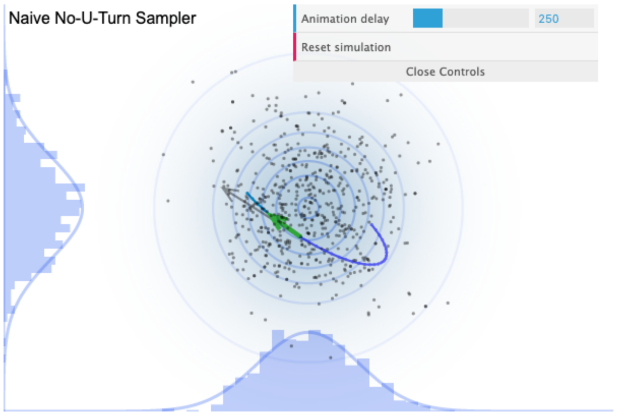



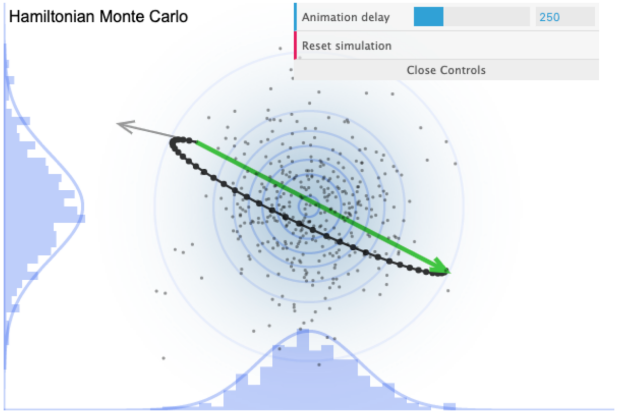
# Multilevel models account for dependencies among parameters

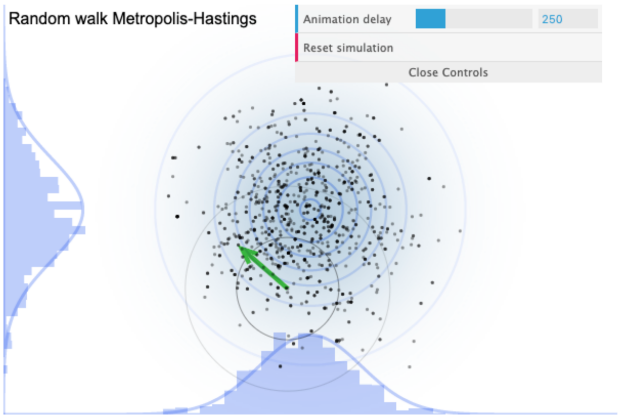


# Markov chain Monte Carlo (MCMC) methods sample from a probability distribution




brms

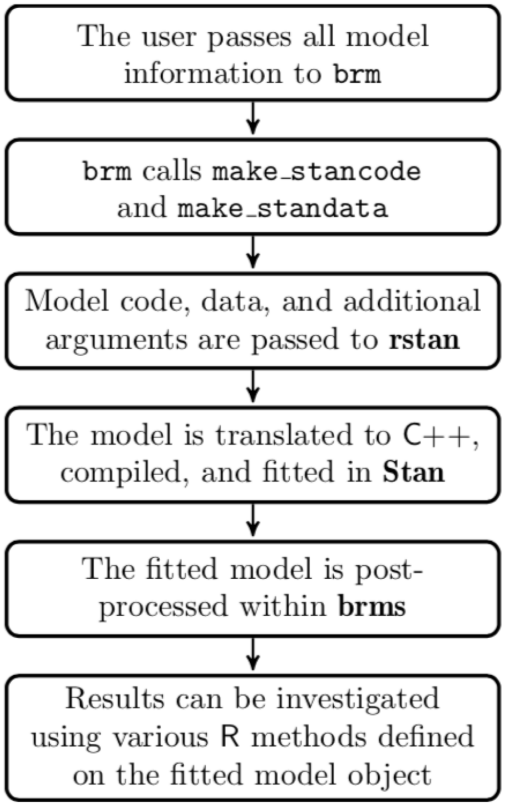
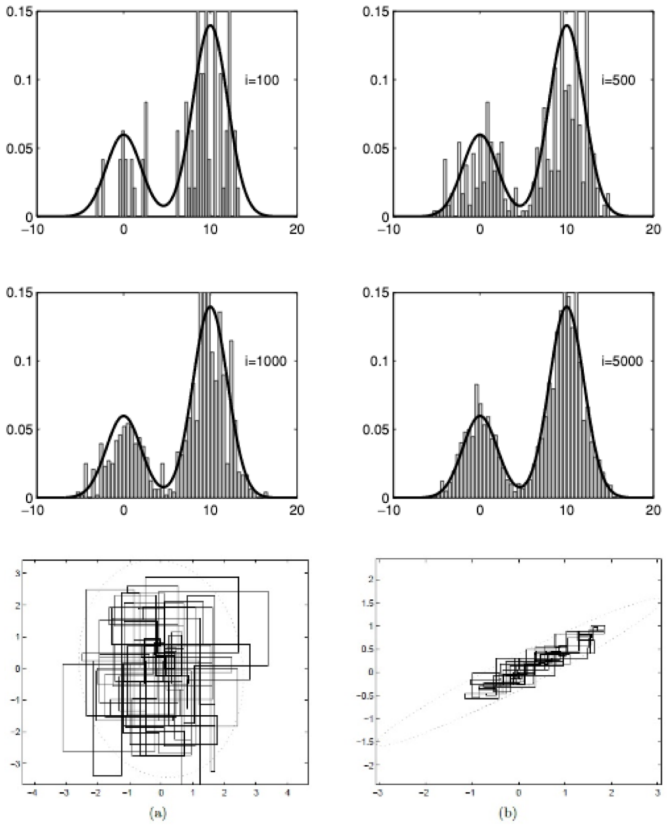



Figure 1: High level description of the model fitting procedure used in **brms**.

# With a sufficient number of steps, MCMC samples approx. the distribution



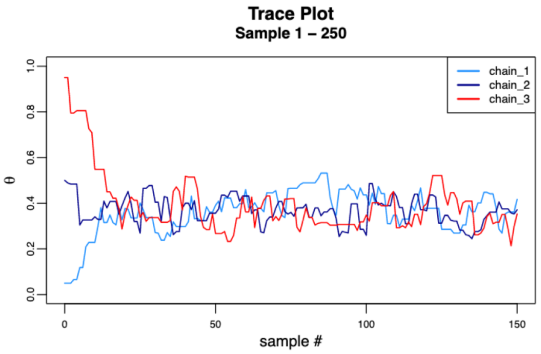




Figure 11: Trace plots: First 150 samples from three chains

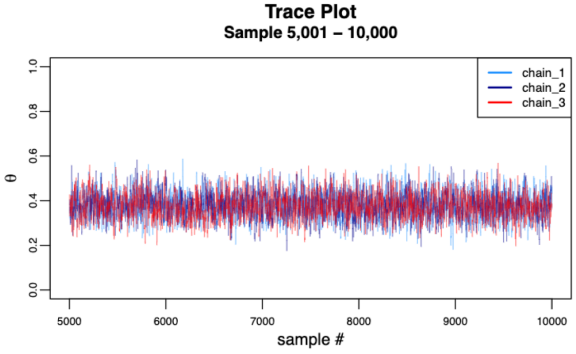
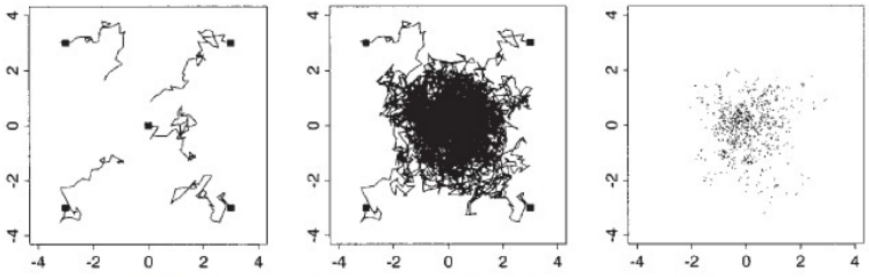
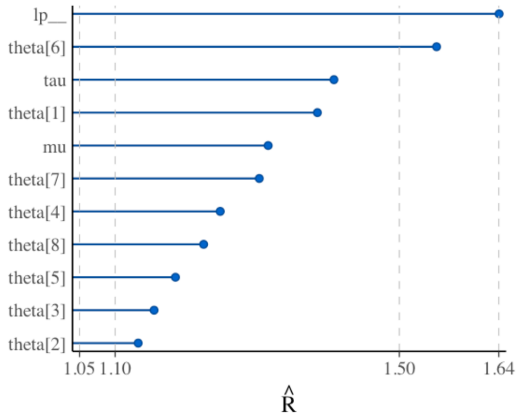



Figure 12: Trace plots: Last 5,000 samples from three chains



# MCMC Diagnostics

**Rhat:** potential scale reduction statistic

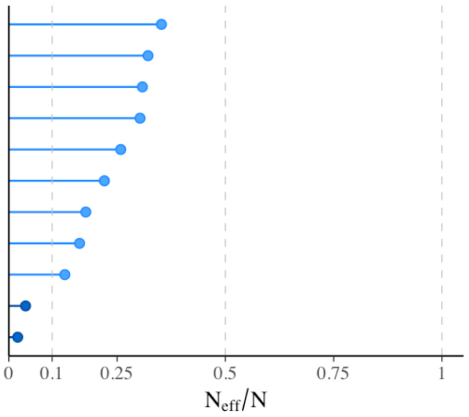



**Guidelines**

- *light*: below 1.05 (good)
- *mid*: between 1.05 and 1.1 (ok)
- *dark*: above 1.1 (too high)

**better**

**$N_{eff}$:** Effective sample size

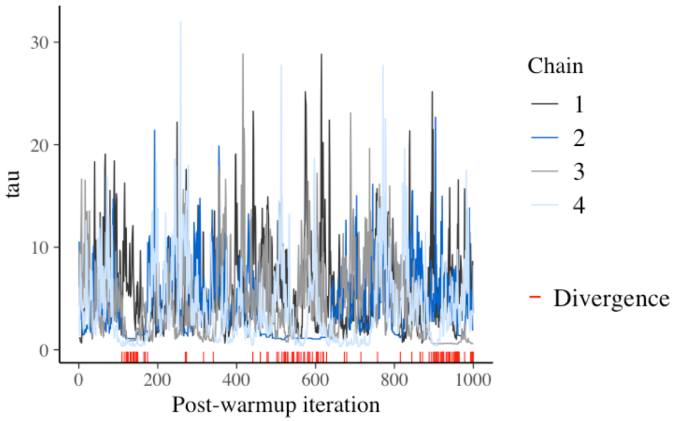



**Guidelines**

- *light*: between 0.5 and 1 (high)
- *mid*: between 0.1 and 0.5 (good)
- *dark*: below 0.1 (low)

**better**

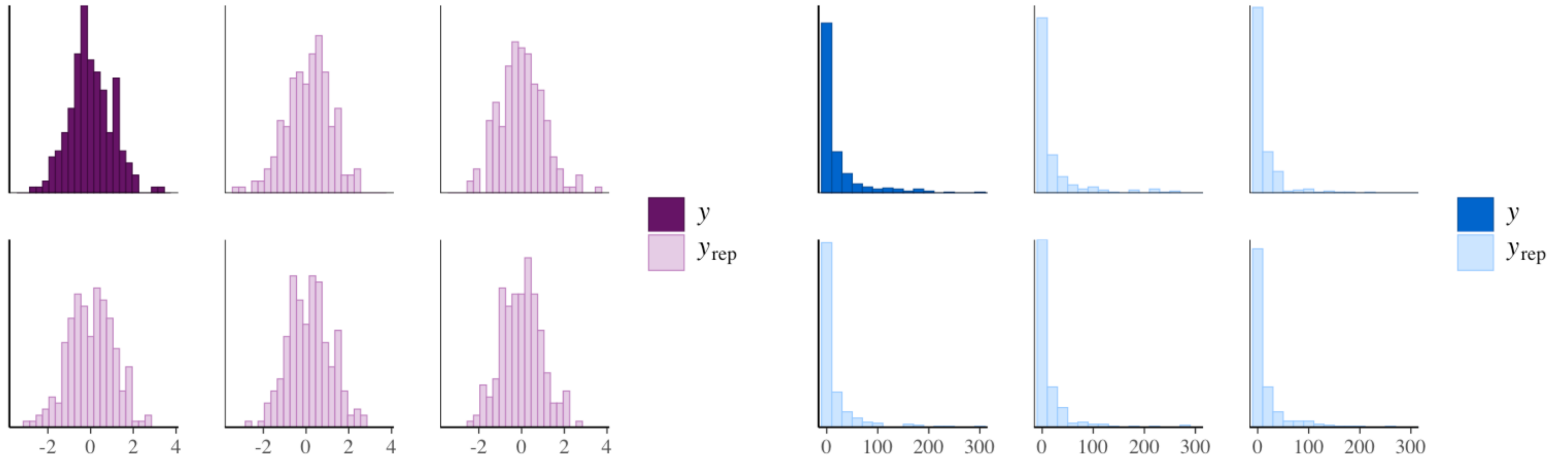
**Trace plot:** a time series plot of the Markov chains



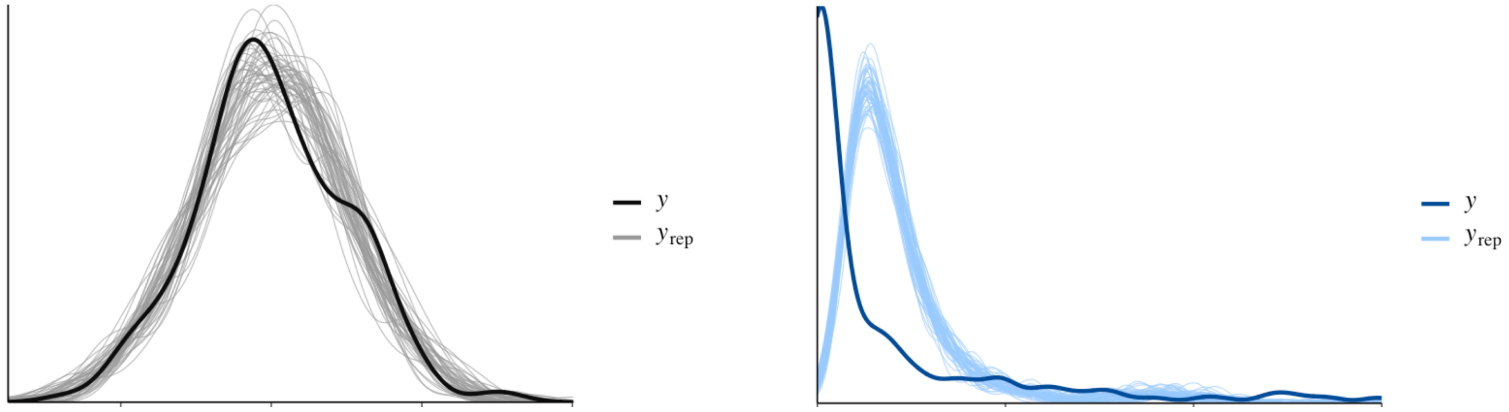

**Guidelines**

# Model Diagnostics: posterior predictive checks

**ppc_hist**




**ppc_dens_overlay**

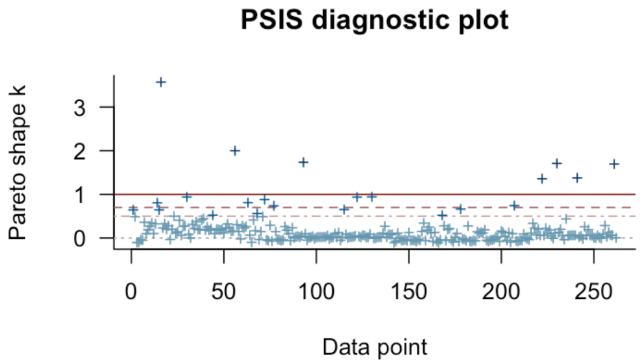


# Model Diagnostics:
## Pareto-smoothed importance sampling (PSIS) Leave-one-out (LOO) cross-validation (PSIS-LOO)

### Pareto *k* diagnostics

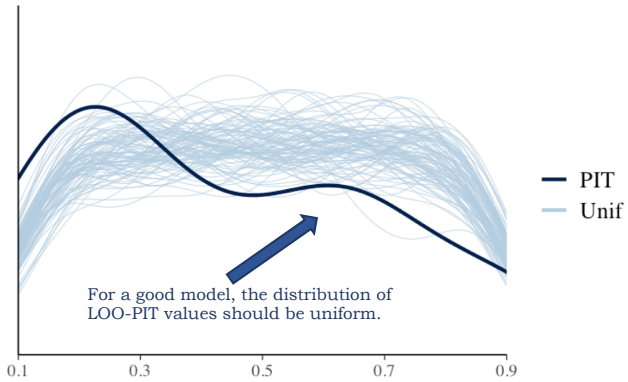
```
Computed from 4000 by 262 log-likelihood matrix

         Estimate     SE
elpd_loo  -6236.9  725.4
p_loo       284.9   69.1
looic     12473.8 1450.7
------
Monte Carlo SE of elpd_loo is NA.

Pareto k diagnostic values:
                         Count Pct.    Min. n_eff
(-Inf, 0.5]   (good)     240   91.6%   206
 (0.5, 0.7]   (ok)         7    2.7%   48
   (0.7, 1]   (bad)        8    3.1%   7
   (1, Inf)   (very bad)   7    2.7%   1
See help('pareto-k-diagnostic') for details.
```

A matrix with two columns (`Estimate`, `SE`) and four rows (`elpd_loo`, `mcse_elpd_loo`, `p_loo`, `looic`). This contains point estimates and standard errors of the expected log pointwise predictive density (`elpd_loo`), the Monte Carlo standard error of `elpd_loo` (`mcse_elpd_loo`), the effective number of parameters (`p_loo`) and the LOO information criterion `looic` (which is just `-2 * elpd_loo`, i.e., converted to deviance scale).




Uh no! Pareto k values in the bad and very bad ranges:
Suggests model misspecification
Consider a more-robust model!

### LOO-PIT plot



# Summarize Results

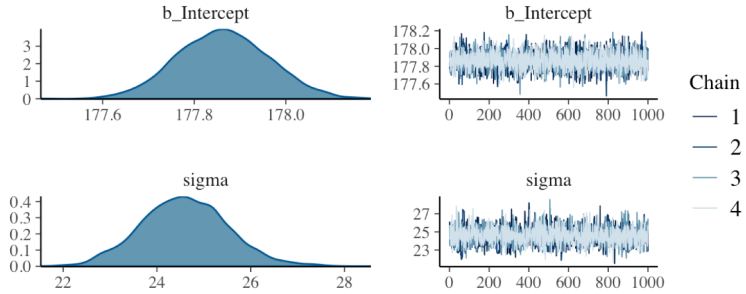
## Intercept, with offsets

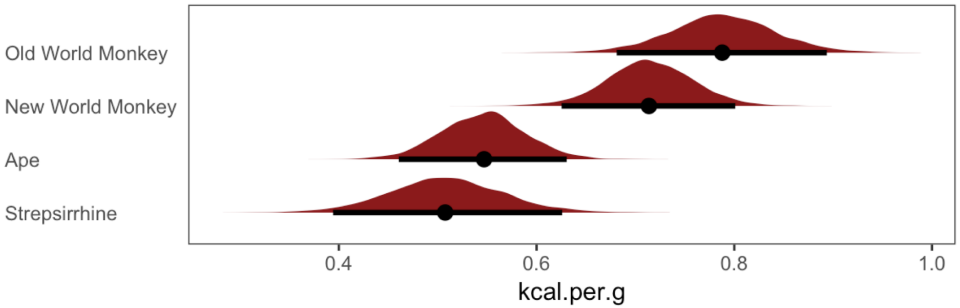
```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: kcal.per.g ~ 1 + clade_nwm + clade_owm + clade_s
##    Data: d (Number of observations: 29)
## Samples: 4 chains, each with iter = 2000; warmup = 500; thin = 1;
##          total post-warmup samples = 6000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept     0.55      0.04     0.46     0.63       4771 1.00
## clade_nwm     0.17      0.06     0.05     0.29       4874 1.00
## clade_owm     0.24      0.07     0.11     0.38       5031 1.00
## clade_s      -0.04      0.07    -0.18     0.11       5308 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.13      0.02     0.10     0.17       5397 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

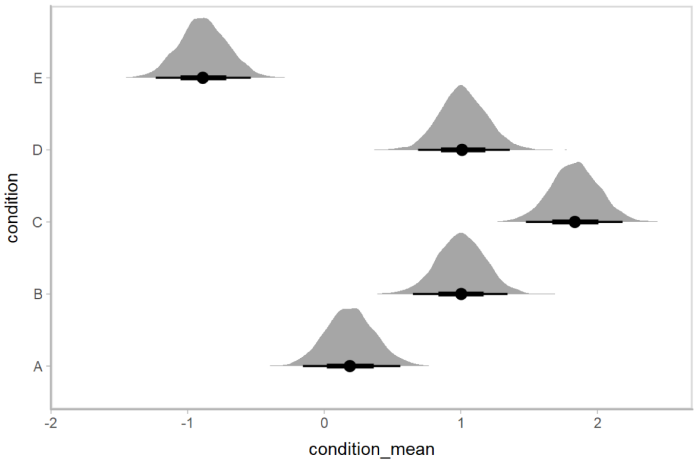
## All Intercepts

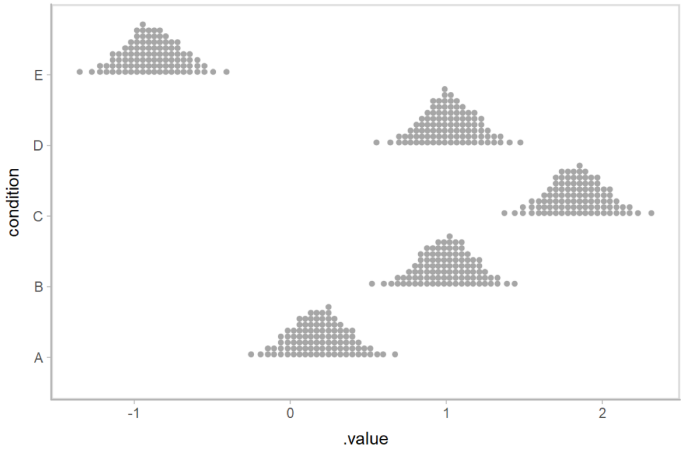
```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: kcal.per.g ~ 0 + clade
##    Data: d (Number of observations: 29)
## Samples: 4 chains, each with iter = 2000; warmup = 500; thin = 1;
##          total post-warmup samples = 6000
##
## Population-Level Effects:
##                     Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## cladeApe                0.55      0.04     0.46     0.64       7023 1.00
## cladeNewWorldMonkey     0.71      0.04     0.63     0.80       7420 1.00
## cladeOldWorldMonkey     0.79      0.05     0.68     0.89       6222 1.00
## cladeStrepsirrhine      0.51      0.06     0.39     0.63       7233 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.13      0.02     0.10     0.17       4777 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

# Visualize Results

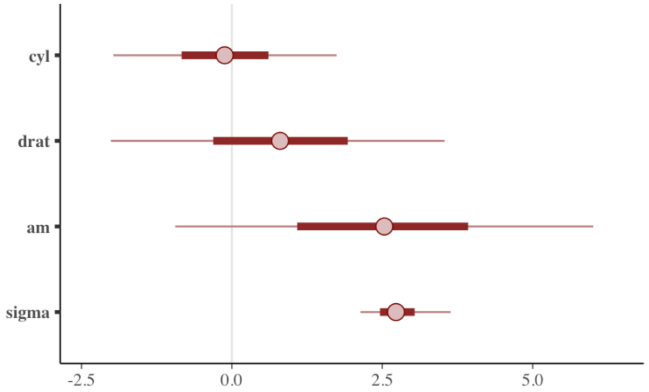


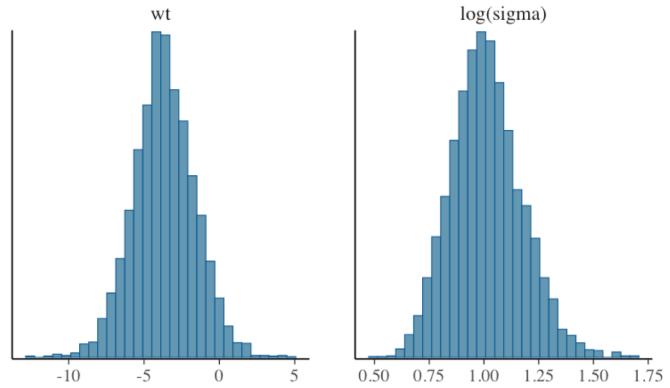



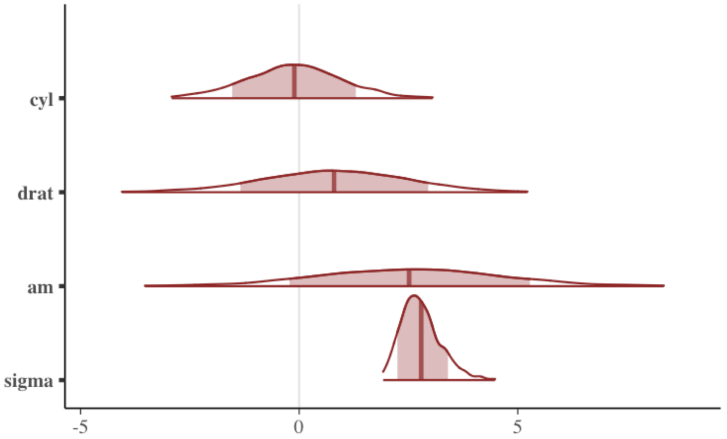



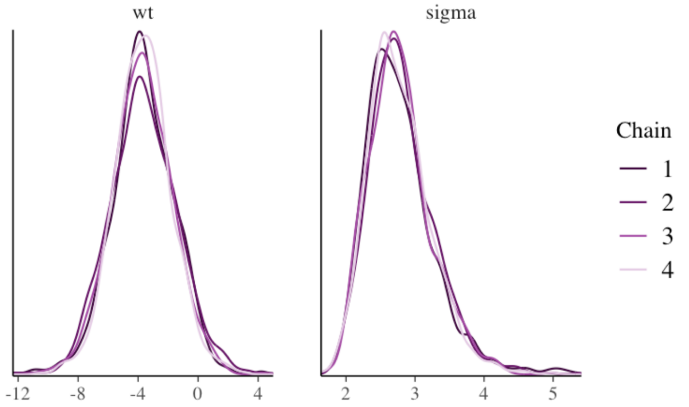



# Visualize Results



# Multilevel modeling: One cluster (varying-intercepts)

## No Pooling

$$\begin{aligned} \text{surv}_i &\sim \text{Binomial}(n_i, p_i) \\ \text{logit}(p_i) &= \alpha_{\text{tank}_i} \\ \alpha_{\text{tank}} &\sim \text{Normal}(0, 5) \end{aligned}$$

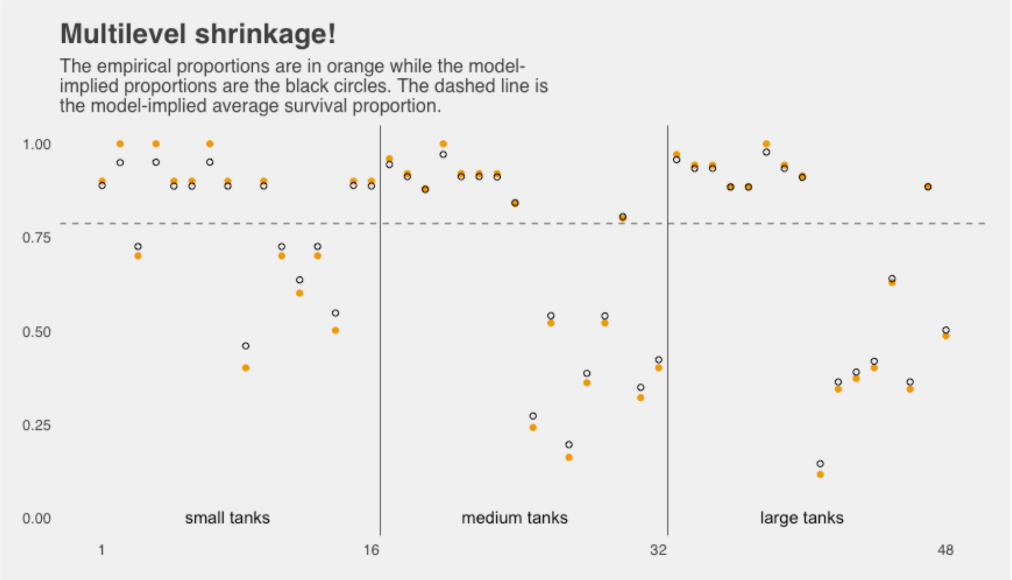
```
b12.1 <-
  brm(data = d, family = binomial,
      surv | trials(density) ~ 0 + factor(tank),
      prior(normal(0, 5), class = b),
      iter = 2000, warmup = 500, chains = 4, cores = 4,
      seed = 12)
```

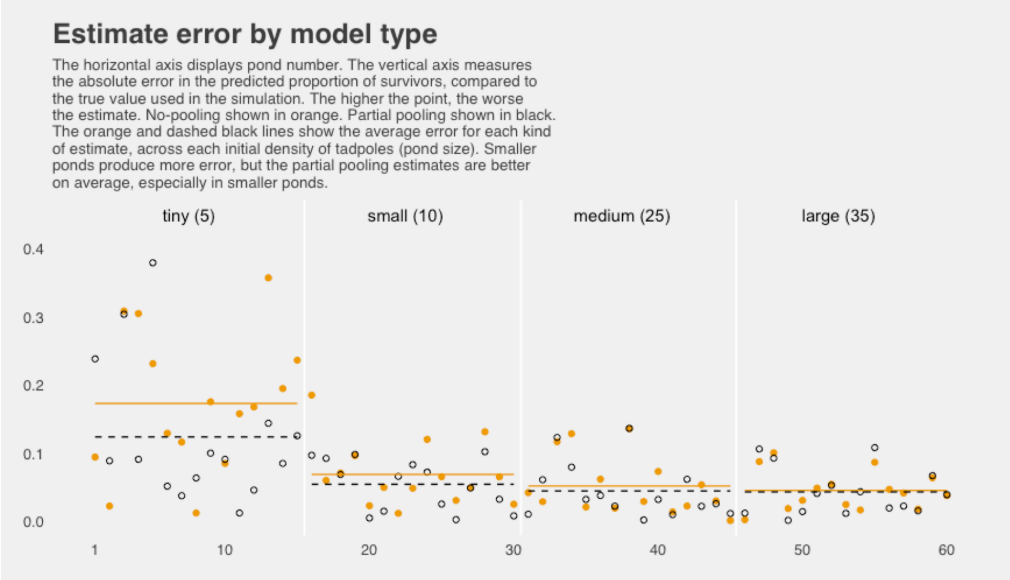
## Multilevel Model

$$\begin{aligned} \text{surv}_i &\sim \text{Binomial}(n_i, p_i) \\ \text{logit}(p_i) &= \alpha_{\text{tank}_i} \\ \alpha_{\text{tank}} &\sim \text{Normal}(\alpha, \sigma) \\ \alpha &\sim \text{Normal}(0, 1) \\ \sigma &\sim \text{HalfCauchy}(0, 1) \end{aligned}$$

```
b12.2 <-
  brm(data = d, family = binomial,
      surv | trials(density) ~ 1 + (1 | tank),
      prior = c(prior(normal(0, 1), class = Intercept),
                prior(cauchy(0, 1), class = sd)),
      iter = 4000, warmup = 1000, chains = 4, cores = 4,
      seed = 12)
```

# The multilevel model's partially pooled estimates are better on average




```
## # A tibble: 2 x 3
##   key            mean_error median_error
##   <chr>               <dbl>        <dbl>
## 1 nopool_error        0.078        0.05
## 2 partpool_error      0.067        0.051
```

# Multilevel modeling:
# More than one cluster (varying-intercepts)

## One Cluster

$$
\begin{aligned}
\text{left\_pull}_i &\sim \text{Binomial}(n_i = 1, p_i) \\
\text{logit}(p_i) &= \alpha + \alpha_{\text{actor}_i} + (\beta_1 + \beta_2 \text{condition}_i)\text{prosoc\_left}_i \\
\alpha_{\text{actor}} &\sim \text{Normal}(0, \sigma_{\text{actor}}) \\
\alpha &\sim \text{Normal}(0, 10) \\
\beta_1 &\sim \text{Normal}(0, 10) \\
\beta_2 &\sim \text{Normal}(0, 10) \\
\sigma_{\text{actor}} &\sim \text{HalfCauchy}(0, 1)
\end{aligned}
$$

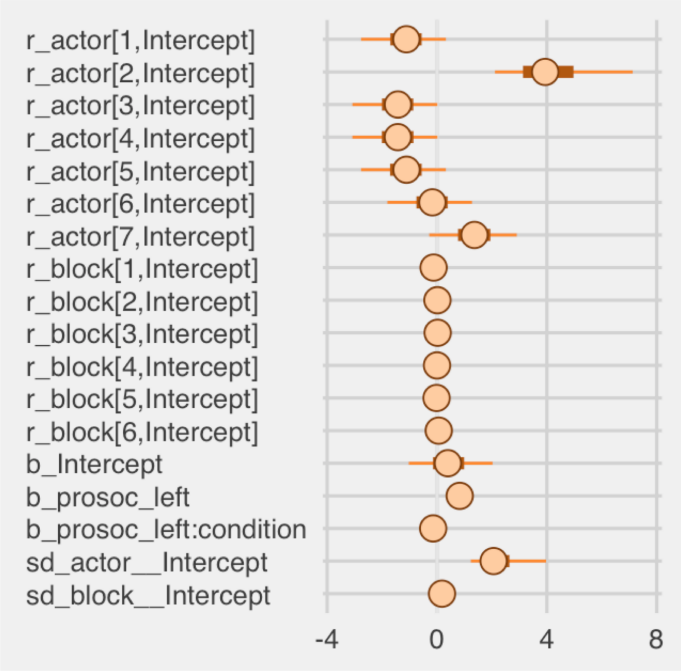
```
b12.4 <-
  brm(data = d, family = binomial,
      pulled_left | trials(1) ~ 1 + prosoc_left + prosoc_left:condition + (1 | actor),
      prior = c(prior(normal(0, 10), class = Intercept),
                prior(normal(0, 10), class = b),
                prior(cauchy(0, 1), class = sd)),
      # I'm using 4 cores, instead of the `cores=3` in McElreath's code
      iter = 5000, warmup = 1000, chains = 4, cores = 4,
      control = list(adapt_delta = 0.95),
      seed = 12)
```

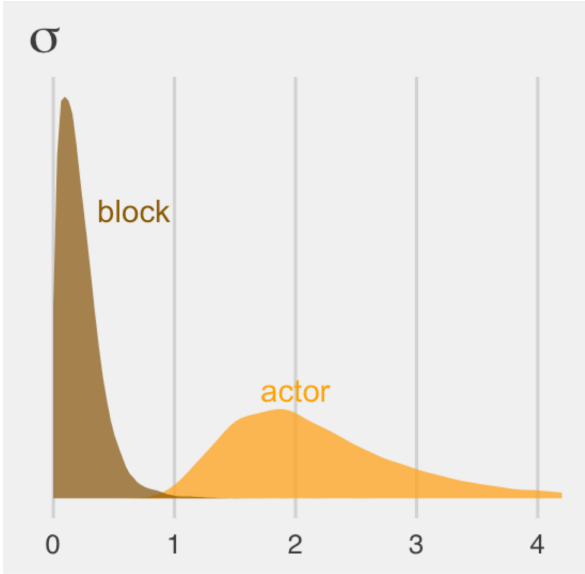
## Two Clusters

$$
\begin{aligned}
\text{left\_pull}_i &\sim \text{Binomial}(n_i = 1, p_i) \\
\text{logit}(p_i) &= \alpha + \alpha_{\text{actor}_i} + \alpha_{\text{block}_i} + (\beta_1 + \beta_2 \text{condition}_i)\text{prosoc\_left}_i \\
\alpha_{\text{actor}} &\sim \text{Normal}(0, \sigma_{\text{actor}}) \\
\alpha_{\text{block}} &\sim \text{Normal}(0, \sigma_{\text{actor}}) \\
\alpha &\sim \text{Normal}(0, 10) \\
\beta_1 &\sim \text{Normal}(0, 10) \\
\beta_2 &\sim \text{Normal}(0, 10) \\
\sigma_{\text{actor}} &\sim \text{HalfCauchy}(0, 1) \\
\sigma_{\text{block}} &\sim \text{HalfCauchy}(0, 1)
\end{aligned}
$$

```
b12.5 <-
  update(b12.4,
         newdata = d,
         formula = pulled_left | trials(1) ~ 1 + prosoc_left + prosoc_left:condition +
           (1 | actor) + (1 | block),
         iter = 6000, warmup = 1000, cores = 4, chains = 4,
         control = list(adapt_delta = 0.99),
         seed = 12)
```

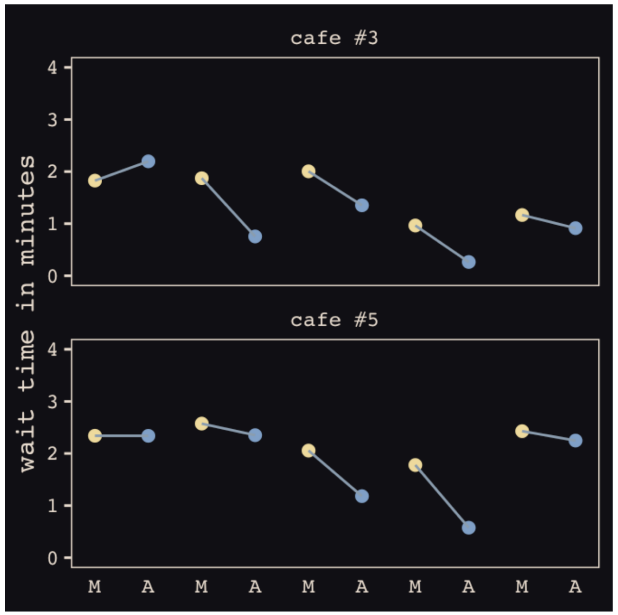
# Multilevel modeling:
## Compare variation between clusters



# Multilevel modeling: (varying-intercepts, varying-slopes)

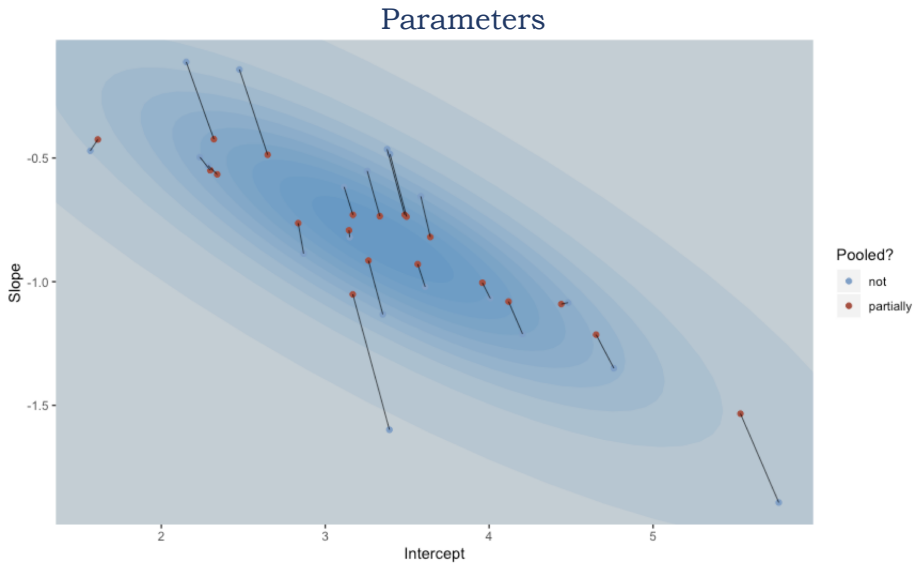
"In the population of interest, intercepts and slopes covary."

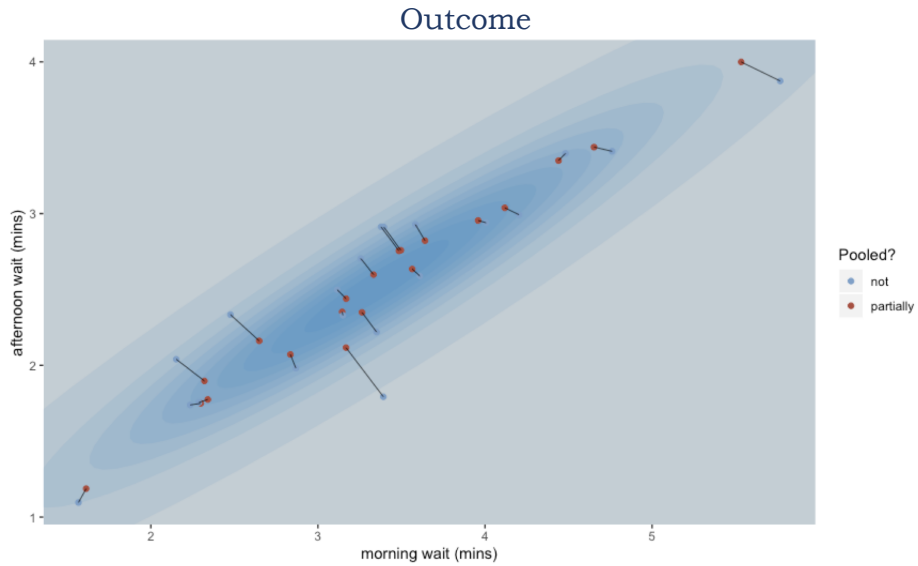



$$
\begin{aligned}
\text{wait}_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{cafe}_i} + \beta_{\text{cafe}_i}\text{afternoon}_i \\
\begin{bmatrix} \alpha_{\text{cafe}} \\ \beta_{\text{cafe}} \end{bmatrix} &\sim \text{MVNormal}\left( \begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \mathbf{S} \right) \\
\mathbf{S} &= \begin{pmatrix} \sigma_\alpha & 0 \\ 0 & \sigma_\beta \end{pmatrix} \mathbf{R} \begin{pmatrix} \sigma_\alpha & 0 \\ 0 & \sigma_\beta \end{pmatrix} \\
\alpha &\sim \text{Normal}(0, 10) \\
\beta &\sim \text{Normal}(0, 10) \\
\sigma &\sim \text{HalfCauchy}(0, 1) \\
\sigma_\alpha &\sim \text{HalfCauchy}(0, 1) \\
\sigma_\beta &\sim \text{HalfCauchy}(0, 1) \\
\mathbf{R} &\sim \text{LKJcorr}(2)
\end{aligned}
$$

```
b13.1 <-
 brm(data = d, family = gaussian,
     wait ~ 1 + afternoon + (1 + afternoon | cafe),
     prior = c(prior(normal(0, 10), class = Intercept),
               prior(normal(0, 10), class = b),
               prior(cauchy(0, 2), class = sd),
               prior(cauchy(0, 2), class = sigma),
               prior(lkj(2), class = cor)),
     iter = 5000, warmup = 2000, chains = 2, cores = 2,
     seed = 13)
```

# Shrinkage on the parameter scale produces shrinkage on the outcome scale



# Information Criteria:
## Measures of out-of-sample deviance

**Pareto-smoothed importance sampling (PSIS)**
**Leave-one-out (LOO) cross-validation (PSIS-LOO)**

The PSIS estimate of the LOO expected log pointwise predictive density is

$$\widehat{\text{elpd}}_{\text{psis-loo}} = \sum_{i=1}^{n} \log\left(\frac{\sum_{s=1}^{S} w_i^s \, p(y_i|\theta^s)}{\sum_{s=1}^{S} w_i^s}\right)$$

The estimated shape parameter $\hat{k}$ of the generalized Pareto distribution can be used to assess the reliability of the estimate:

- If $k < \frac{1}{2}$, the variance of the raw importance ratios is finite, the central limit theorem holds, and the estimate converges quickly.
- If $k$ is between $\frac{1}{2}$ and 1, the variance of the raw importance ratios is infinite but the mean exists, the generalized central limit theorem for stable distributions holds, and the convergence of the estimate is slower. The variance of the PSIS estimate is finite but may be large.
- If $k > 1$, the variance and the mean of the raw ratios distribution do not exist. The variance of the PSIS estimate is finite but may be large.

**Widely Applicable Information Criterion (WAIC)**

$$\text{WAIC} = -2(\text{lppd} - p_{\text{WAIC}})$$

$$\text{lppd} = \sum_{i=1}^{N} \log \Pr(y_i)$$

$$p_{\text{WAIC}} = \sum_{i=1}^{N} V(y_i)$$

# Compare models:
## PSIS-LOO

### PSIS-LOO Output for a Model

```
## Computed from 4000 by 17 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo      4.4 1.9
## p_loo         1.3 0.3
## looic        -8.8 3.8
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     16    94.1%   3013
##  (0.5, 0.7]   (ok)        1     5.9%   2041
##    (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.
```

A matrix with two columns (`Estimate`, `SE`) and four rows (`elpd_loo`, `mcse_elpd_loo`, `p_loo`, `looic`). This contains point estimates and standard errors of the expected log pointwise predictive density (`elpd_loo`), the Monte Carlo standard error of `elpd_loo` (`mcse_elpd_loo`), the effective number of parameters (`p_loo`) and the LOO information criterion `looic` (which is just `-2 * elpd_loo`, i.e., converted to deviance scale).

Nice. No pareto k values in the bad or very bad ranges.

### Compare Models based on PSIS-LOO

First two models are basically equivalent and preferred over the third.

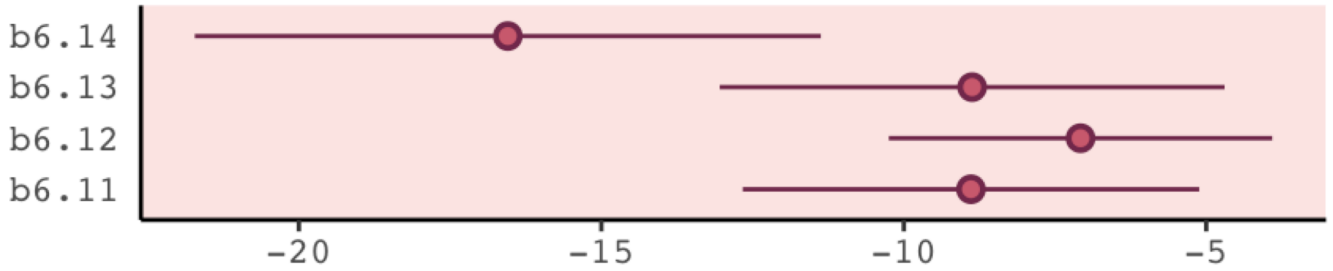
```
Model comparison:
(ordered by highest ELPD)
```

| | elpd_diff | elpd_loo | se_elpd_loo | p_loo | se_p_loo | looic | se_looic |
|---|---|---|---|---|---|---|---|
| fit_pool | 0.0 | -46.5 | 2.7 | 1.1 | 0.3 | 92.9 | 5.3 |
| fit_partialpool | -0.2 | -46.6 | 2.2 | 3.6 | 0.8 | 93.3 | 4.3 |
| fit_nopool | -6.2 | -52.7 | 0.7 | 10.9 | 0.4 | 105.3 | 1.5 |

When comparing two fitted models, we can estimate the difference in their expected predictive accuracy by the difference in `elpd_loo` or `elpd_waic` (or multiplied by −2, if desired, to be on the deviance scale).

When using `loo_compare()`, the returned matrix will have one row per model and several columns of estimates. The values in the `elpd_diff` and `se_diff` columns of the returned matrix are computed by making pairwise comparisons between each model and the model with the largest ELPD (the model in the first row). For this reason the `elpd_diff` column will always have the value `0` in the first row (i.e., the difference between the preferred model and itself) and negative values in subsequent rows for the remaining models.

# Compare models:
## WAIC




First model preferred over the others.

**Compare Models based on WAIC**

| ## | elpd_diff | se_diff | elpd_waic | se_elpd_waic | p_waic | se_p_waic | waic | se_waic |
|---|---|---|---|---|---|---|---|---|
| ## b6.14 | 0.0 | 0.0 | 8.3 | 2.6 | 3.2 | 0.9 | −16.5 | 5.2 |
| ## b6.11 | −3.8 | 2.5 | 4.4 | 1.9 | 1.3 | 0.3 | −8.9 | 3.8 |
| ## b6.13 | −3.8 | 1.8 | 4.4 | 2.1 | 2.0 | 0.4 | −8.9 | 4.2 |
| ## b6.12 | −4.7 | 2.5 | 3.5 | 1.6 | 2.0 | 0.3 | −7.1 | 3.2 |

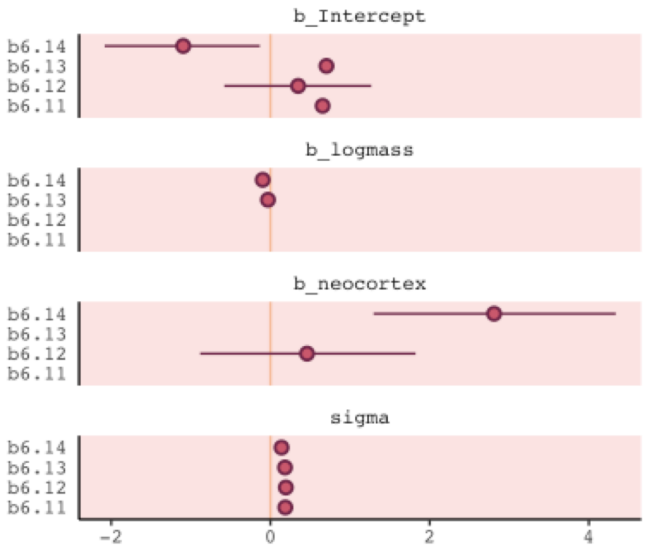
Same substantive conclusion using WAIC.

| ## | waic_diff | se |
|---|---|---|
| ## b6.14 | 0.000000 | 0.000000 |
| ## b6.11 | 7.657211 | 5.013383 |
| ## b6.13 | 7.676240 | 3.599457 |
| ## b6.12 | 9.468233 | 5.061744 |

**waic_diff = -2 * elpd_diff**
**se = se_diff * 2**

# Compare models: Coefficients

```
## # A tibble: 4 x 5
##   term        b6.11 b6.12  b6.13 b6.14
##   <chr>       <dbl> <dbl>  <dbl> <dbl>
## 1 b_Intercept  0.66  0.35   0.7  -1.1
## 2 b_logmass   NA    NA     -0.03 -0.1
## 3 b_neocortex NA     0.46  NA     2.81
## 4 sigma        0.19  0.19   0.18  0.14
```



# Additional Benefits

- Incorporate Measurement Error
  - Outcome and Predictors

Incorporate measurement error on the outcome and predictor variables during modeling.

```
b14.2_mi <-
  brm(data = dlist, family = gaussian,
      div_obs | mi(div_sd) ~ 0 + intercept + me(mar_obs, mar_sd) + A,
      prior = c(prior(normal(0, 10), class = b),
                prior(cauchy(0, 2.5), class = sigma)),
      iter = 5000, warmup = 1000, cores = 2, chains = 2,
      seed = 14,
      control = list(adapt_delta = 0.99,
                     max_treedepth = 12),
      save_mevars = TRUE,
      inits = inits_list)
```

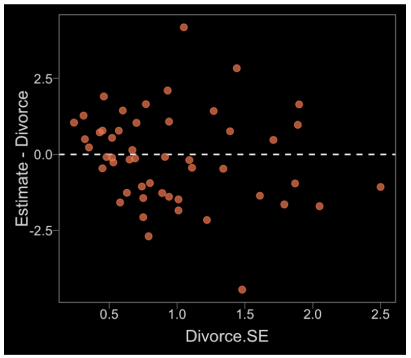
- Impute Missing Values

Impute missing values “on-the-fly”, with models for the missing values, during modeling.

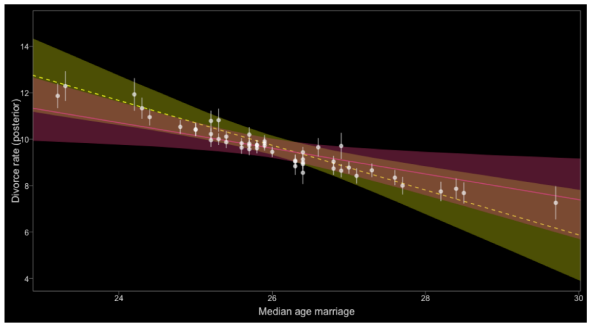
```
# define the model
b_model <-
  bf(kcal ~ 1 + mi(neocortex) + logmass) +
  bf(neocortex | mi() ~ 1 + logmass) + # here's the big difference
  set_rescor(FALSE)

# fit the model
b14.4 <-
  brm(data = data_list,
      family = gaussian,
      b_model,
      prior = c(prior(normal(0, 100), class = Intercept, resp = kcal),
                prior(normal(0.5, 1), class = Intercept, resp = neocortex),
                prior(normal(0, 10),  class = b),
                prior(cauchy(0, 1),   class = sigma,     resp = kcal),
                prior(cauchy(0, 1),   class = sigma,     resp = neocortex)),
      iter = 1e4, chains = 2, cores = 2,
      seed = 14)
```

# Measurement Error

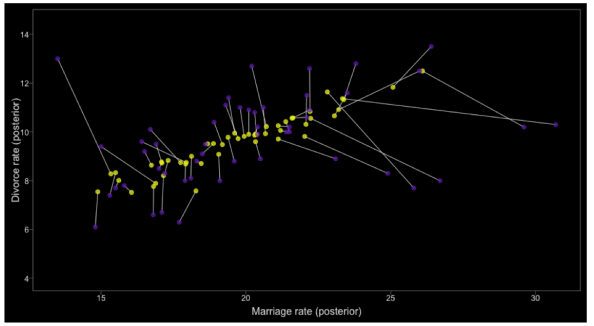
**Shrinkage:** States with more **uncertain** divorce rates have estimates more different from observed. Less **certain** estimates are improved by pooling info from more certain estimates.




**Shrinkage:** The estimates and regression trend have moved. For a State with an **uncertain** divorce rate, the trend has influenced the estimate. For a State with a more **certain** divorce rate, the State has influenced the regression trend.

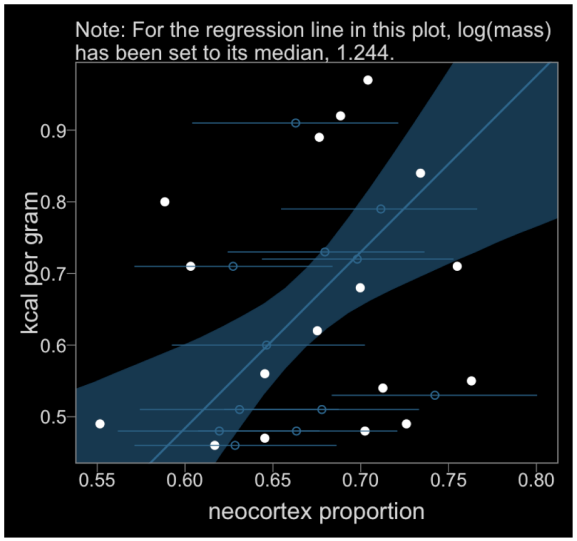



**Shrinkage:** Observed (**purple**) and estimated (**yellow**) divorce and marriage rates. States with **uncertain** marriage rates tend to be small States with high rates. Pooling results in smaller estimates for these States.

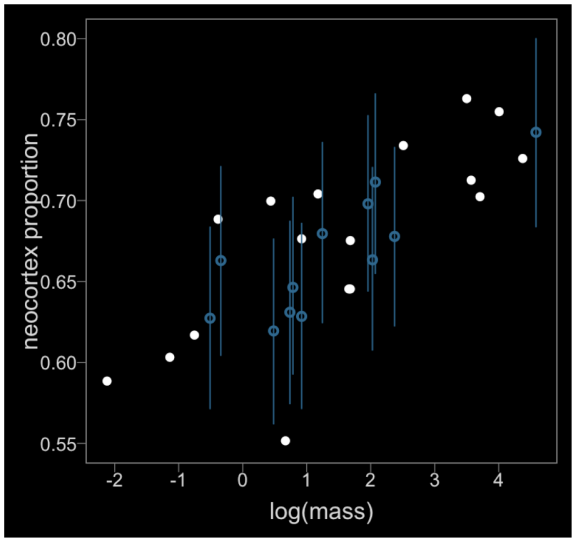


# Missing Values

Inferred relationship between outcome and predictor, with imputed values shown as open points




Inferred relationship between the two predictors, with imputed values shown as open points



# Advantages and Disadvantage of Methodology

- **Advantages**
  - Better uncertainty estimates
  - Incorporate prior information
  - Incorporate measurement error
  - Model and impute missing data values

- **Disadvantage**
  - Speed

# Some Resources

- **Statistical Rethinking recoded:** https://bookdown.org/connect/#/apps/1850/access
- **Statistical Rethinking:** https://xcelab.net/rm/statistical-rethinking/
- **Stein's Paradox:** https://solomonkurz.netlify.com/post/stein-s-paradox-and-what-partial-pooling-can-do-for-you/
- **Bayesian Notes:** https://jrnold.github.io/bayesian_notes/multilevel-models.html
- **Bayesian Linear Mixed Models:** https://willhipson.netlify.com/post/bayesian_mlm/bayesian_mlm/
- **Bayesian Linear Mixed Models using Stan:** http://www.ling.uni-potsdam.de/~vasishth/statistics/BayesLMMs.html
- **Doing Bayesian Data Analysis:** https://sites.google.com/site/doingbayesiandataanalysis/software-installation
- **brms:** https://paul-buerkner.github.io/brms/index.html
- **rstanarm:** http://mc-stan.org/rstanarm/articles/index.html
- **bayesplot:** https://cran.r-project.org/web/packages/bayesplot/vignettes/visual-mcmc-diagnostics.html
- **tidybayes:** https://mjskay.github.io/tidybayes/articles/tidy-brms.html
- **PyMC3:** https://docs.pymc.io/nb_examples/index.html
- **ArviZ:** https://arviz-devs.github.io/arviz/
- **Stan:** https://mc-stan.org/

# Appendix