Chapter 5

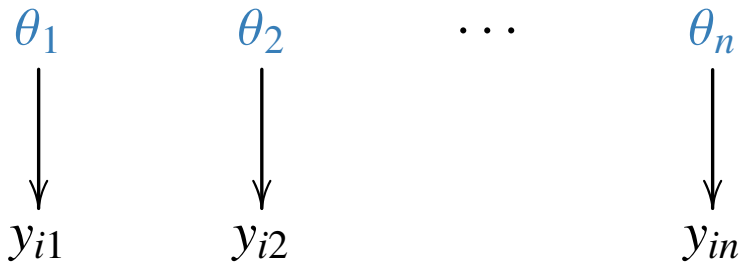
- 5.1 Lead-in to hierarchical models
- 5.2 Exchangeability (useful concept)
- 5.3 Bayesian analysis of hierarchical models (we use Stan/brms for computation)
- 5.4 Hierarchical normal model (we use Stan/brms for computation)
- 5.5 Example: parallel experiments in eight schools (useful discussion on benefits of hierarchical model)
- 5.6 Meta-analysis (can be skipped)
- 5.7 Weakly informative priors for hierarchical variance parameters

Hierarchical model

- In simple model: posterior for the parameters
- In hierarchical model: posterior for the prior parameters

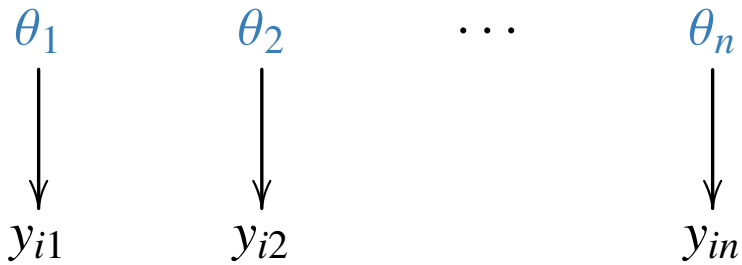
Hierarchical model

- Example: CVD treatment effectiveness
  - in hospital j the survival probability is θ_j
  - observations y_{ij} tell whether patient i survived in hospital j

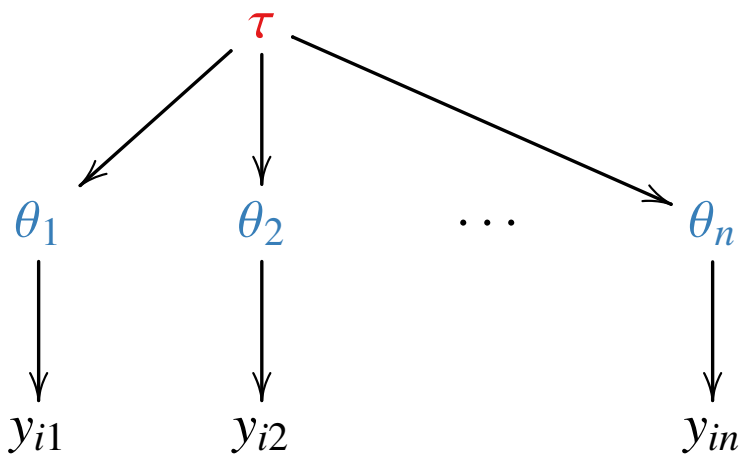


Hierarchical model

- Example: CVD treatment effectiveness
  - in hospital j the survival probability is θ_j
  - observations y_{ij} tell whether patient i survived in hospital j



  - sensible to assume that θ_j are similar



  - natural to think that θ_j have common population distribution
  - θ_j is not directly observed and the population distribution is unknown

Hierarchical model: terms

Level 1: observations given parameters $p(y_{ij} \mid \theta_j)$

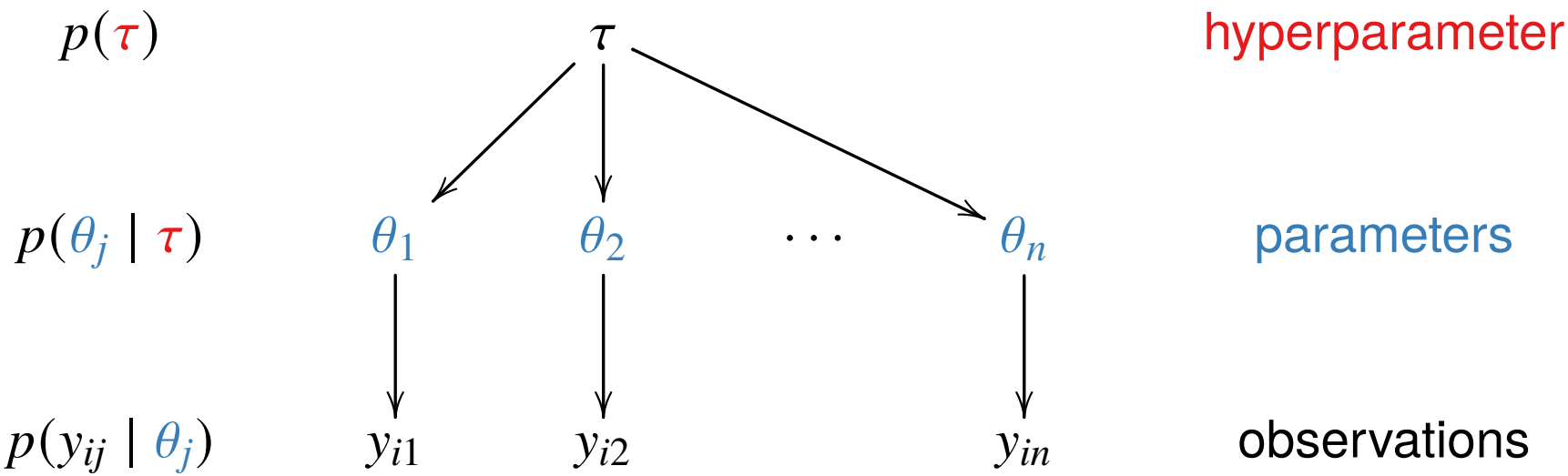
Joint posterior

$$\begin{aligned} p(\theta, \tau \mid y) &\propto p(y \mid \theta, \tau) p(\theta, \tau) \\ &\propto p(y \mid \theta) p(\theta \mid \tau) p(\tau) \end{aligned}$$

Hierarchical model: terms

Level 1: observations given parameters $p(y_{ij} \mid \theta_j)$

Level 2: parameters given hyperparameters $p(\theta_j \mid \tau)$

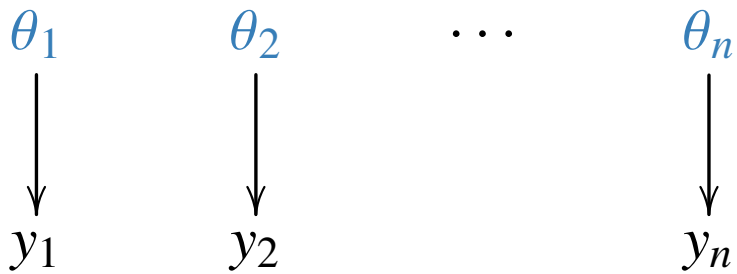


Joint posterior

$$\begin{aligned} p(\theta, \tau \mid y) &\propto p(y \mid \theta, \tau) p(\theta, \tau) \\ &\propto p(y \mid \theta) p(\theta \mid \tau) p(\tau) \end{aligned}$$

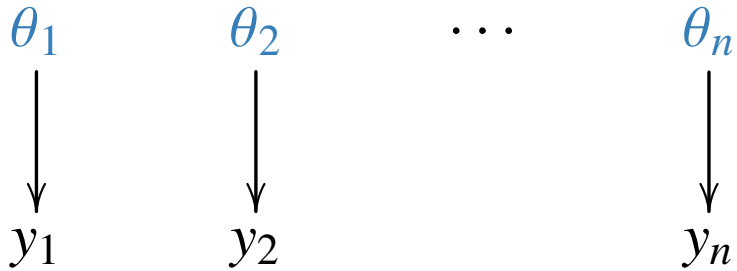
Compare

- "Separate model" (model with separate/independent effects)

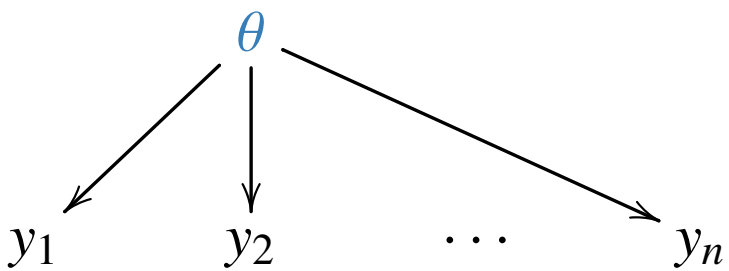


Compare

- "Separate model" (model with separate/independent effects)

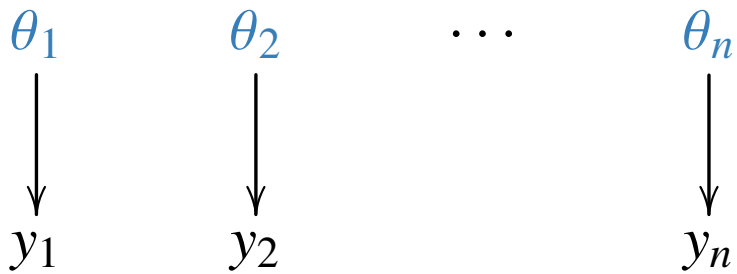


- "Joint model" (model with a common effect / pooled model)

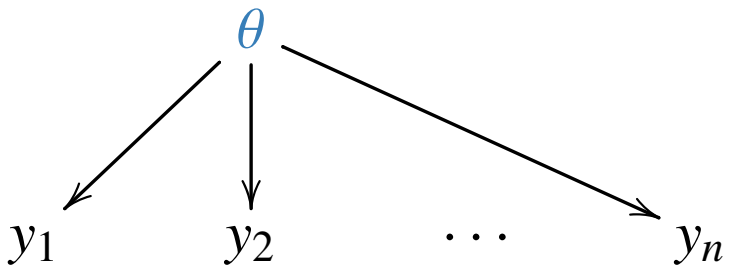


Compare

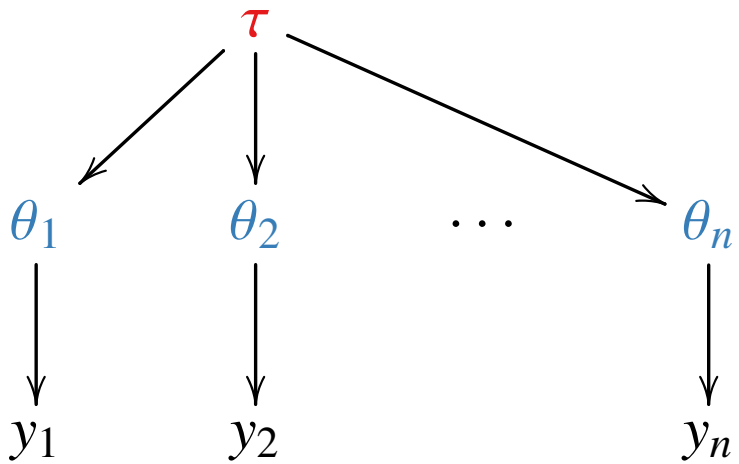
- "Separate model" (model with separate/independent effects)



- "Joint model" (model with a common effect / pooled model)



- Hierarchical model



Hierarchical binomial model: rats

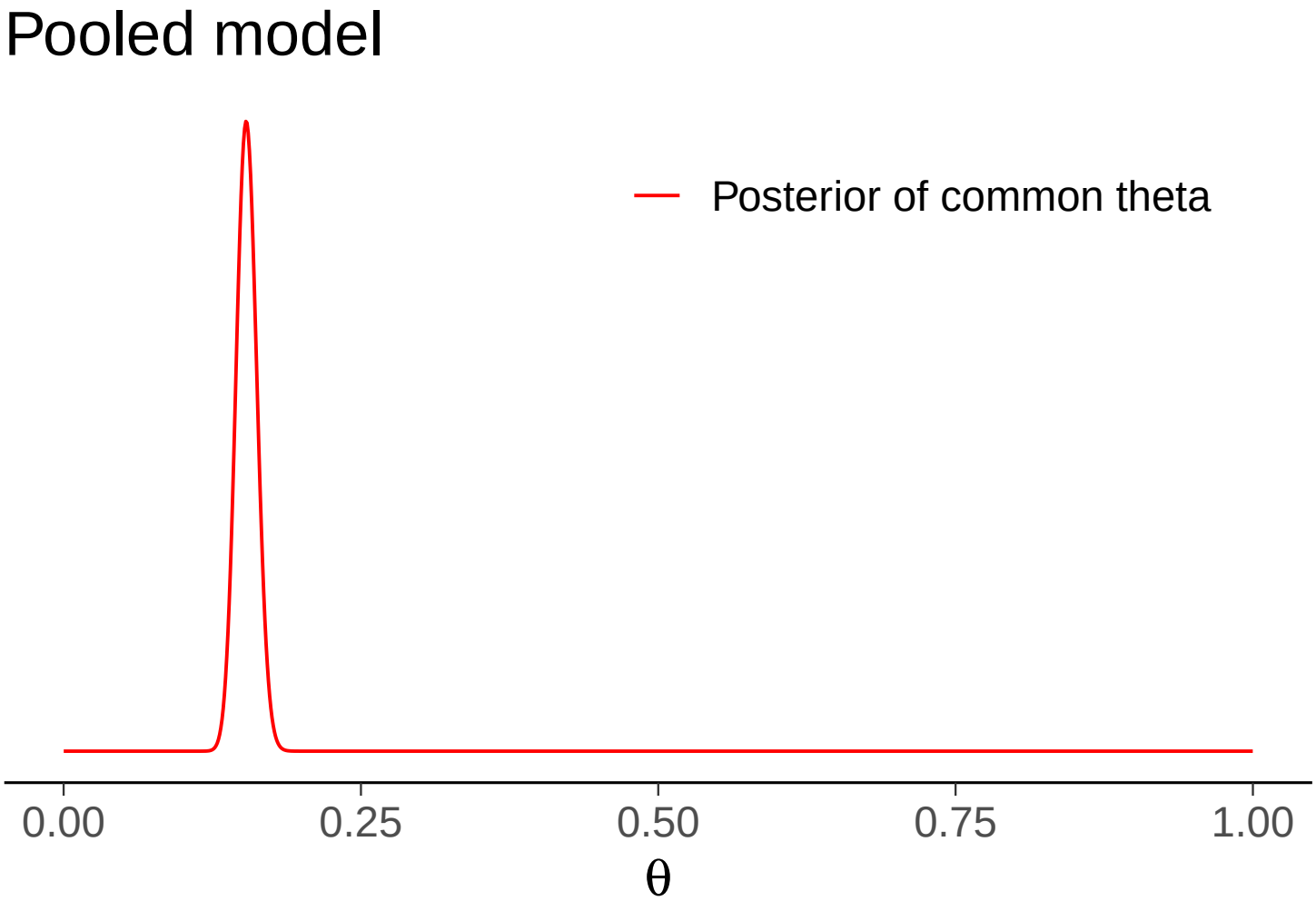
- Medicine testing
- Type F344 female rats in control group given placebo
  - count how many get endometrial stromal polyps
  - familiar binomial model example

Hierarchical binomial model: rats

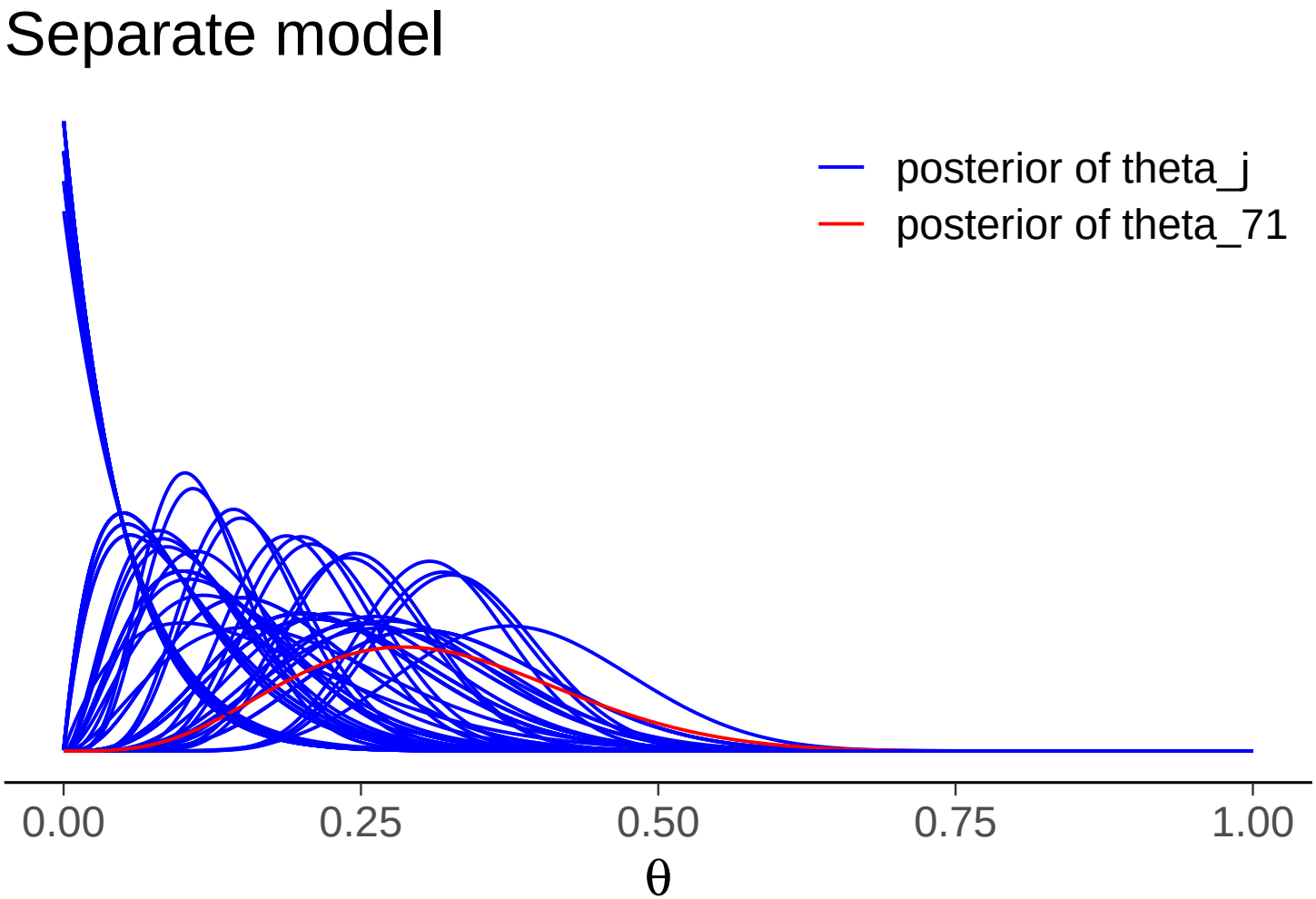
- Medicine testing
- Type F344 female rats in control group given placebo
  - count how many get endometrial stromal polyps
  - familiar binomial model example
- Experiment has been repeated 71 times

0/20	0/20	0/20	0/20	0/20	0/20	0/20	0/19	0/19	0/19
0/19	0/18	0/18	0/17	1/20	1/20	1/20	1/20	1/19	1/19
1/18	1/18	2/25	2/24	2/23	2/20	2/20	2/20	2/20	2/20
2/20	1/10	5/49	2/19	5/46	3/27	2/17	7/49	7/47	3/20
3/20	2/13	9/48	10/50	4/20	4/20	4/20	4/20	4/20	4/20
4/20	10/48	4/19	4/19	4/19	5/22	11/46	12/49	5/20	5/20
6/23	5/19	6/22	6/20	6/20	6/20	16/52	15/46	15/47	9/24
4/14									

Hierarchical binomial model: rats



Hierarchical binomial model: rats

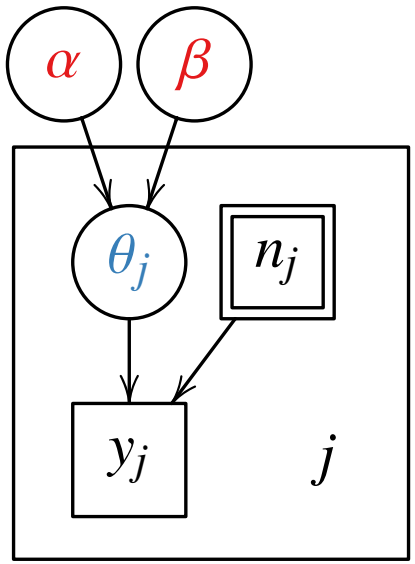


Hierarchical binomial model: rats

- Hierarchical binomial model for rats
  prior parameters α and β are unknown

$$\theta_j \mid \alpha, \beta \sim \text{Beta}(\theta_j \mid \alpha, \beta)$$

$$y_j \mid n_j, \theta_j \sim \text{Bin}(y_j \mid n_j, \theta_j)$$



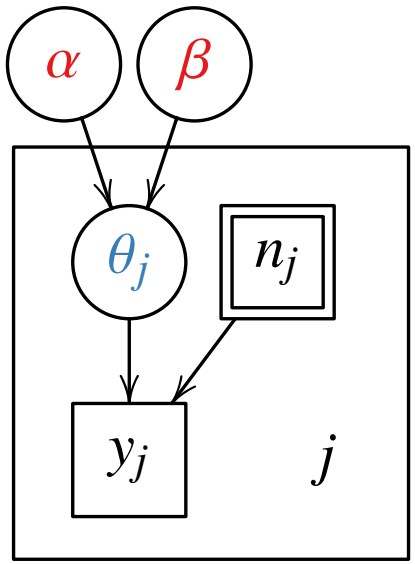
- Joint posterior $p(\theta_1, \ldots, \theta_J, \alpha, \beta \mid y)$
  - multiple parameters

Hierarchical binomial model: rats

- Hierarchical binomial model for rats
  prior parameters α and β are unknown

$$\theta_j \mid \alpha, \beta \sim \text{Beta}(\theta_j \mid \alpha, \beta)$$

$$y_j \mid n_j, \theta_j \sim \text{Bin}(y_j \mid n_j, \theta_j)$$

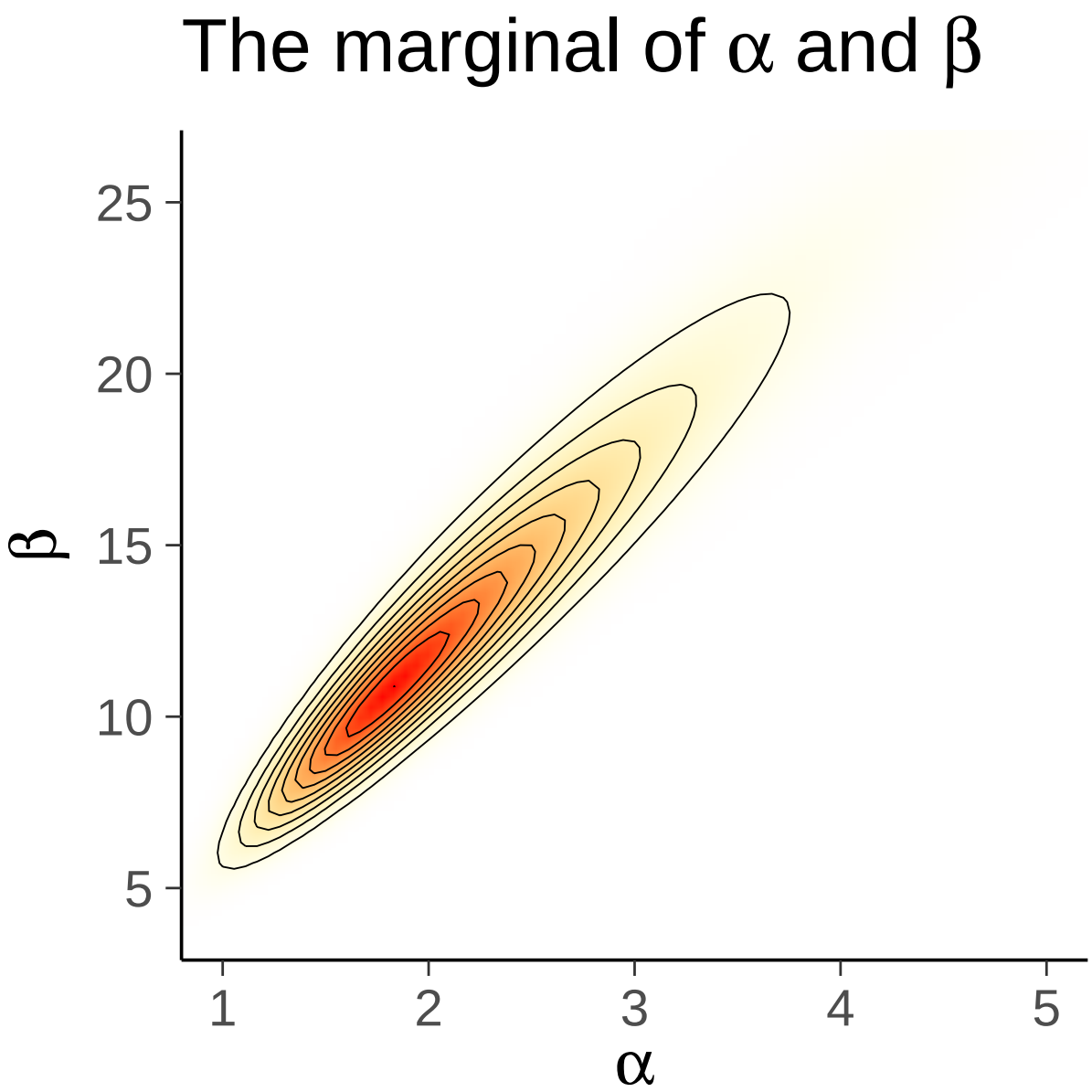


- Joint posterior $p(\theta_1, \ldots, \theta_J, \alpha, \beta \mid y)$
  - multiple parameters
  - factorize $\prod_{j=1}^{J} p(\theta_j \mid \alpha, \beta, y) p(\alpha, \beta \mid y)$

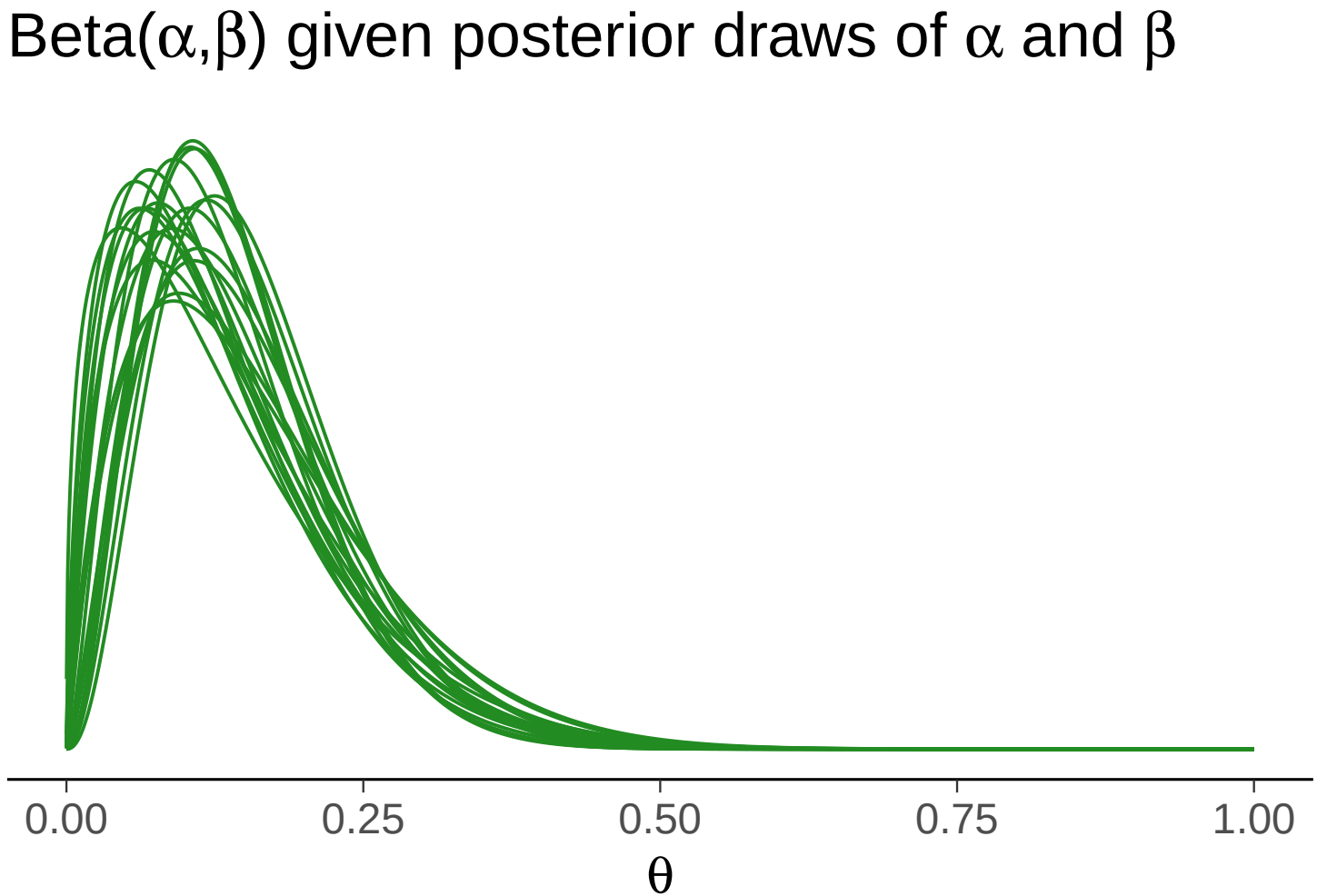
Hierarchical binomial model: rats

- Population prior $\mathrm{Beta}(\theta_j \mid \alpha, \beta)$
- Hyperprior $p(\alpha, \beta)$?
  - α, β both affect the location and scale
  - BDA3 has $p(\alpha, \beta) \propto (\alpha + \beta)^{-5/2}$
    - diffuse prior for location and scale (BDA3 p. 110)
- demo5_1

Hierarchical binomial model: rats

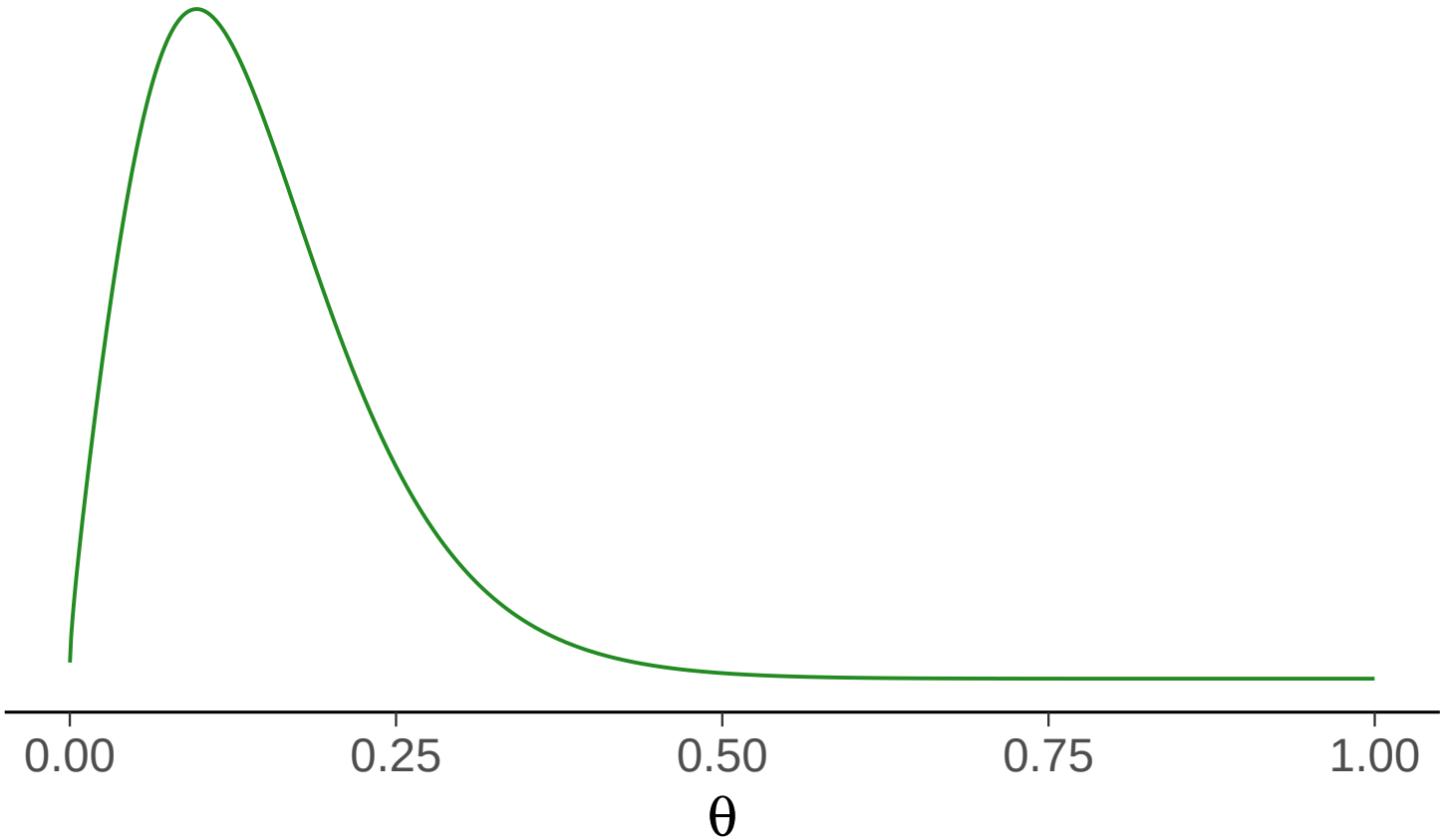


Hierarchical binomial model: rats



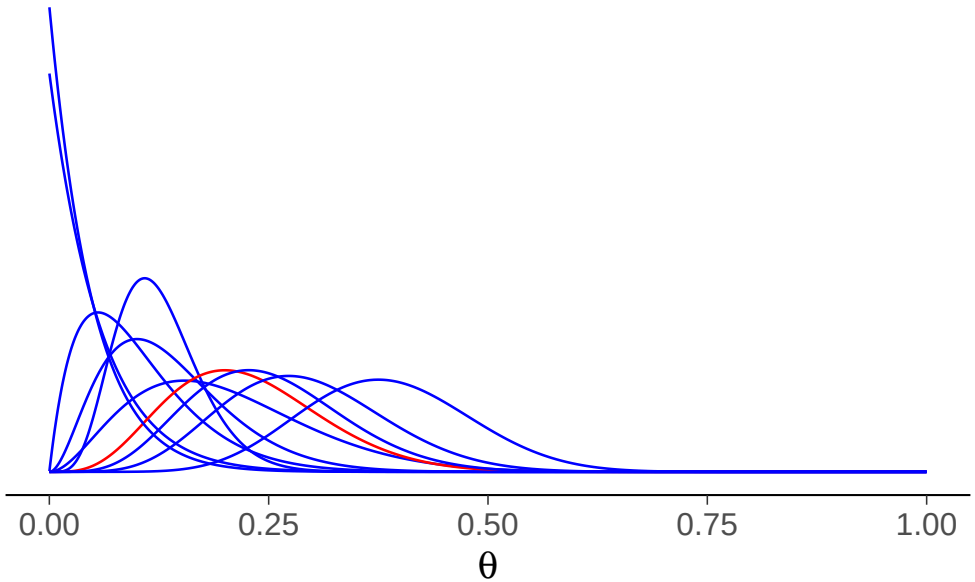
Hierarchical binomial model: rats

Population distribution (prior) for θ_j



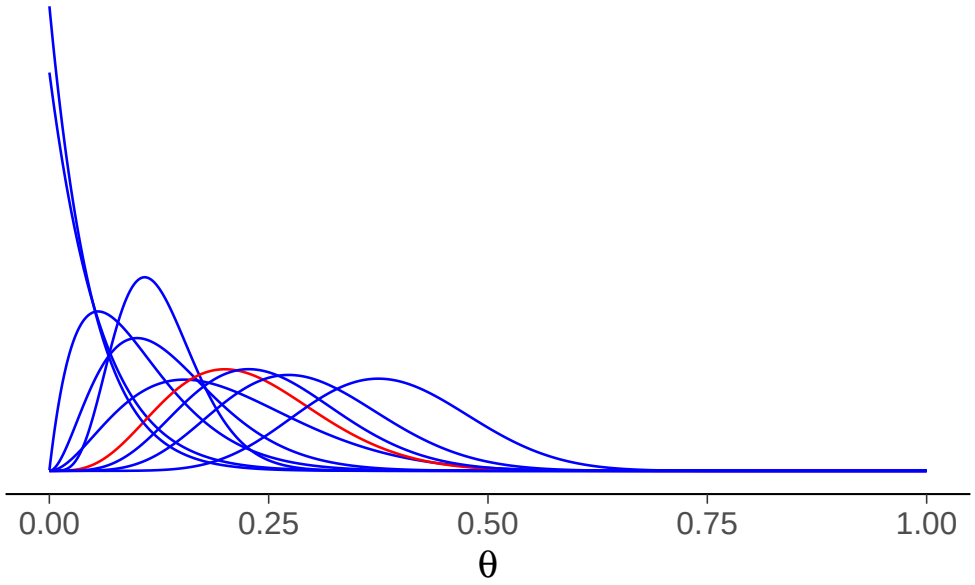
Hierarchical binomial model: rats

Separate model

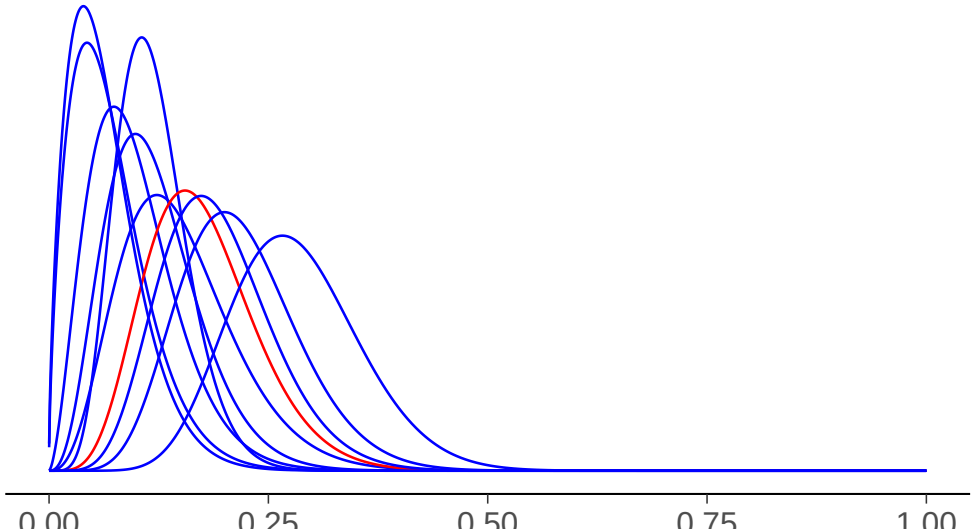


Hierarchical binomial model: rats

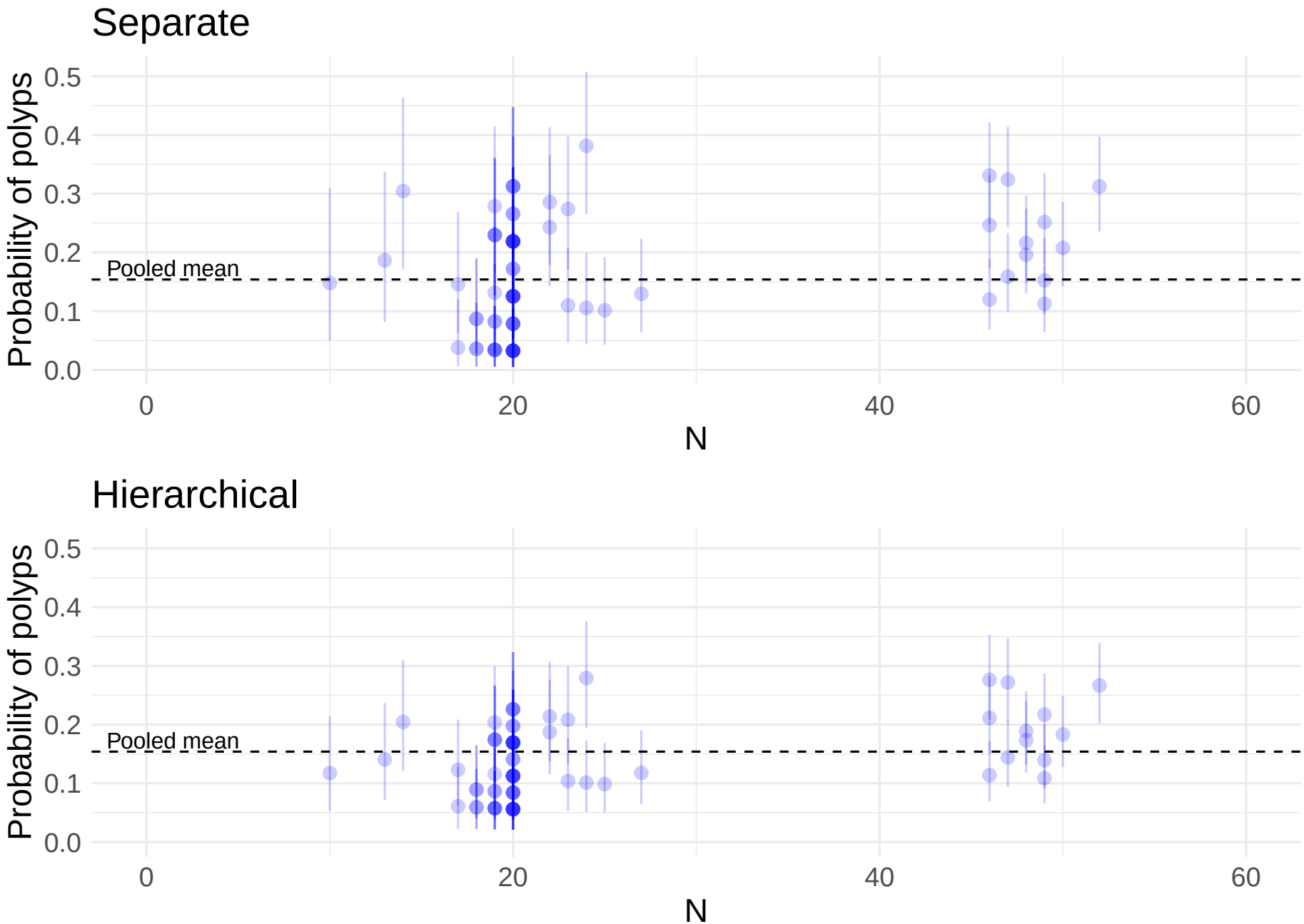
Separate model



Hierarchical model

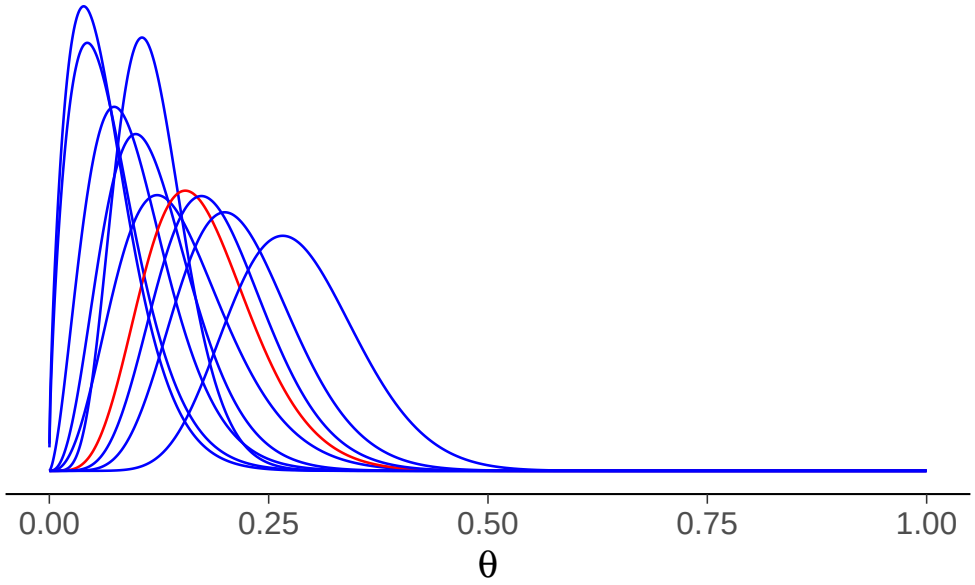


Hierarchical model and group size: Rats

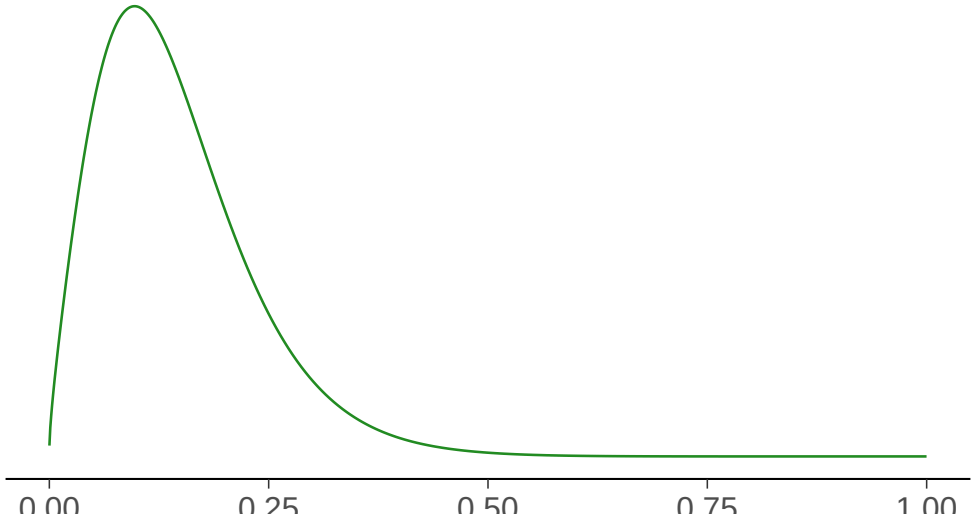


Hierarchical binomial model: rats

Hierarchical model



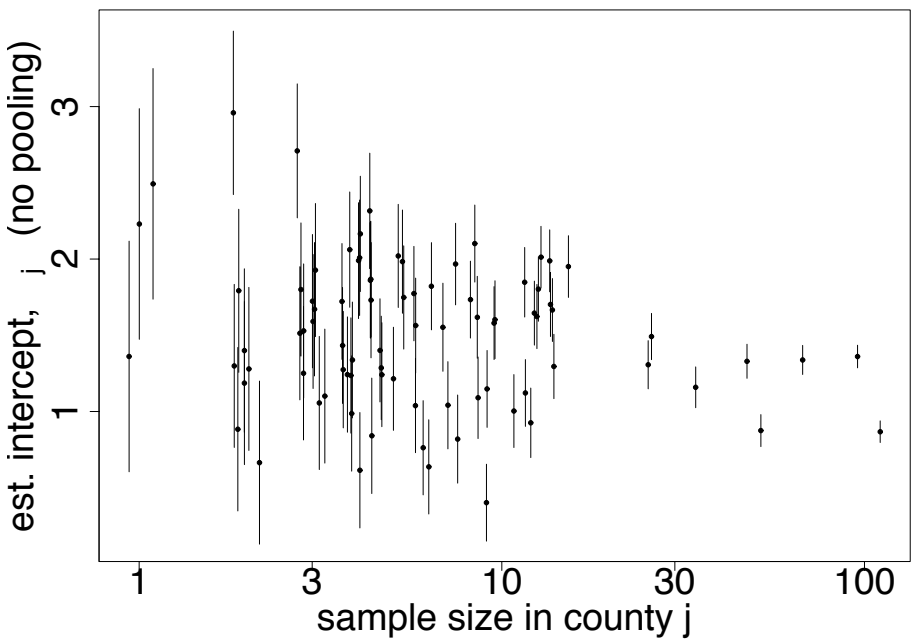
Population distribution (prior) for θ_j



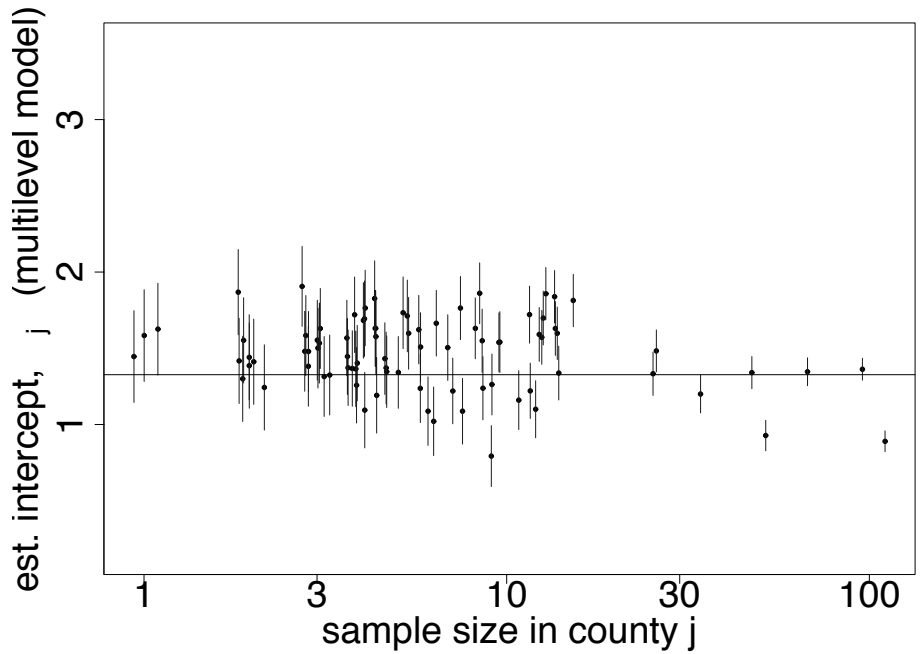
Hierarchical model and group size: Radon

919 home radon levels in 85 counties in Minnesota:

Separate



Hierarchical



Diet effect on chicken weights (at age 12 days)

- A typical treatment effect analysis
- Models
  - a separate model, in which each diet is modeled individually
  - a pooled model, in which all measurements are combined and there is no distinction between diets
  - a hierarchical model

Stan hierarchical model

`diet_idx` is a vector with each element indicating the group

```
model {
  // Priors
  for (diet in 1:N_diets) {
    mu_diet[diet] ~ normal(mu_0, sd);
  }
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  for (obs in 1:N_observations) {
    weight[obs] ~ normal(mu_diet[diet_idx[obs]], sigma);
  }
}
```

Stan hierarchical model

`mu_0` and `sd` are the population mean and sd

```
model {
  // Priors
  for (diet in 1:N_diets) {
    mu_diet[diet] ~ normal(mu_0, sd);
  }
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  for (obs in 1:N_observations) {
    weight[obs] ~ normal(mu_diet[diet_idx[obs]], sigma);
  }
}
```

Stan hierarchical model

sd is constrained to be positive and thus the prior is half-normal

```
model {
  // Priors
  for (diet in 1:N_diets) {
    mu_diet[diet] ~ normal(mu_0, sd);
  }
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  for (obs in 1:N_observations) {
    weight[obs] ~ normal(mu_diet[diet_idx[obs]], sigma);
  }
}
```

Stan hierarchical model

`sigma` is constrained to be positive and thus the prior is half-normal

```
model {
  // Priors
  for (diet in 1:N_diets) {
    mu_diet[diet] ~ normal(mu_0, sd);
  }
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  for (obs in 1:N_observations) {
    weight[obs] ~ normal(mu_diet[diet_idx[obs]], sigma);
  }
}
```

Stan without loops

Vectorized statements

```
model {
  // Priors
  mu_diet ~ normal(mu_0, sd);
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  weight ~ normal(mu_diet[diet_idx], sigma);
}
```

Stan vs brms

Stan

```
model {
  // Priors
  mu_diet ~ normal(mu_0, sd);
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  weight ~ normal(mu_diet[diet_idx], sigma);
}
```

brms formula

```
brm(weight ~ 1 + (1 | Diet),
```

Stan vs brms

Stan

```
model {
  // Priors
  mu_diet ~ normal(mu_0, sd);
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  weight ~ normal(mu_diet[diet_idx], sigma);
}
```

brms formula

```
brm(weight ~ 1 + (1 | Diet),
```

Stan vs brms

Stan

```
model {
  // Priors
  mu_diet ~ normal(mu_0, sd);
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  weight ~ normal(mu_diet[diet_idx], sigma);
}
```

brms formula

```
brm(weight ~ 1 + (1 | Diet),
```

Stan vs brms

Stan

```
model {
  // Priors
  mu_diet ~ normal(mu_0, sd);
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  weight ~ normal(mu_diet[diet_idx], sigma);
}
```

brms formula

```
brm(weight ~ 1 + (1 | Diet),
```

Stan vs brms

Stan

```
model {
  // Priors
  mu_diet ~ normal(mu_0, sd);
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  weight ~ normal(mu_diet[diet_idx], sigma);
}
```

brms formula

```
brm(weight ~ 1 + (1 | Diet), data=Chick12,
```

Stan vs brms

Stan

```
model {
  // Priors
  mu_diet ~ normal(mu_0, sd);
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  weight ~ normal(mu_diet[diet_idx], sigma);
}
```

brms formula

```
brm(weight ~ 1 + (1 | Diet), data=Chick12,
    prior=c(prior(normal(0,1), class="Intercept"), # p(mu_0)
            prior(normal(0,1), class="sd"),        # p(tau)
            prior(normal(0,1), class="sigma")))    # p(sigma)
```

Stan vs brms

Stan

```
model {
  // Priors
  mu_diet ~ normal(mu_0, sd);
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  weight ~ normal(mu_diet[diet_idx], sigma);
}
```

brms formula

```
brm(weight ~ 1 + (1 | Diet), data=Chick12,
    prior=c(prior(normal(0,1), class="Intercept"), # p(mu_0)
            prior(normal(0,1), class="sd"),        # p(tau)
            prior(normal(0,1), class="sigma")))    # p(sigma)
```

Stan vs brms

Stan

```
model {
  // Priors
  mu_diet ~ normal(mu_0, sd);
  mu_0 ~ normal(0, 1);
  sd ~ normal(0, 1)
  sigma ~ normal(0, 1);

  // Observation model
  weight ~ normal(mu_diet[diet_idx], sigma);
}
```

brms formula

```
brm(weight ~ 1 + (1 | Diet), data=Chick12,
    prior=c(prior(normal(0,1), class="Intercept"), # p(mu_0)
            prior(normal(0,1), class="sd"),        # p(tau)
            prior(normal(0,1), class="sigma")))    # p(sigma)
```

brms generated Stan code

```
// generated with brms 2.22.1
data {
  int<lower=1> N;  // total number of observations
  vector[N] Y;  // response variable
  // data for group-level effects of ID 1
  int<lower=1> N_1;  // number of grouping levels
  int<lower=1> M_1;  // number of coefficients per level
  array[N] int<lower=1> J_1;  // grouping indicator per observation
  // group-level predictor values
  vector[N] Z_1_1;
  int prior_only;  // should the likelihood be ignored?
}
```

brms generated Stan code

```
parameters {
  real Intercept;  // temporary intercept for centered predictors
  real<lower=0> sigma;  // dispersion parameter
  vector<lower=0>[M_1] sd_1;  // group-level standard deviations
  array[M_1] vector[N_1] z_1;  // standardized group-level effects
}
transformed parameters {
  vector[N_1] r_1_1;  // actual group-level effects
  real lprior = 0;  // prior contributions to the log posterior
  r_1_1 = (sd_1[1] * (z_1[1]));
  lprior += normal_lpdf(Intercept | 0, 1);
  lprior += normal_lpdf(sigma | 0, 1)
    - 1 * normal_lccdf(0 | 0, 1);
  lprior += normal_lpdf(sd_1 | 0, 1)
    - 1 * normal_lccdf(0 | 0, 1);
}
```

brms generated Stan code

```
parameters {
  real Intercept;  // temporary intercept for centered predictors
  real<lower=0> sigma;  // dispersion parameter
  vector<lower=0>[M_1] sd_1;  // group-level standard deviations
  array[M_1] vector[N_1] z_1;  // standardized group-level effects
}
transformed parameters {
  vector[N_1] r_1_1;  // actual group-level effects
  real lprior = 0;  // prior contributions to the log posterior
  r_1_1 = (sd_1[1] * (z_1[1]));
  lprior += normal_lpdf(Intercept | 0, 1);
  lprior += normal_lpdf(sigma | 0, 1)
    - 1 * normal_lccdf(0 | 0, 1);
  lprior += normal_lpdf(sd_1 | 0, 1)
    - 1 * normal_lccdf(0 | 0, 1);
}
```

brms generated Stan code

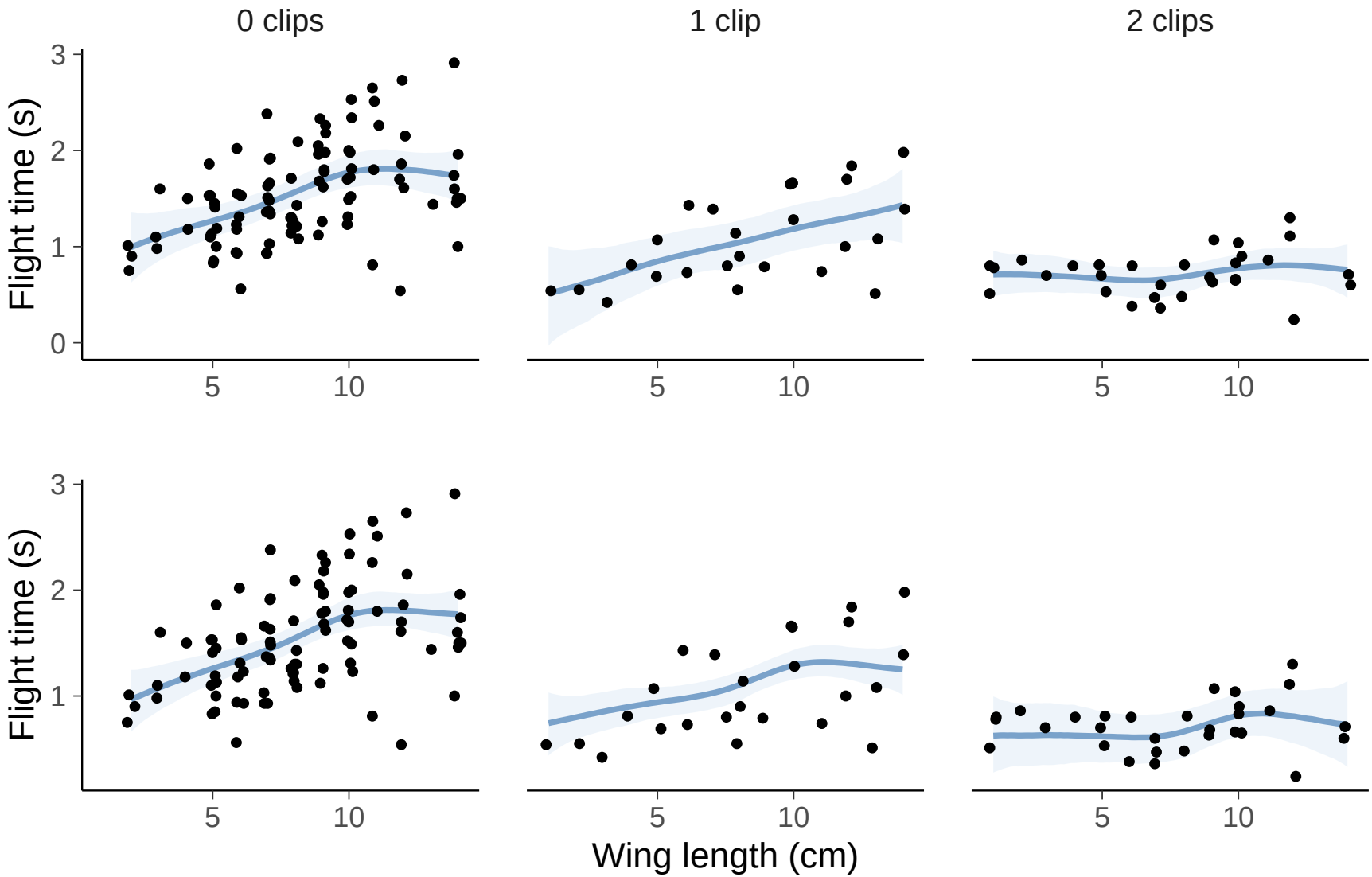
```
model {
  // likelihood including constants
  if (!prior_only) {
    // initialize linear predictor term
    vector[N] mu = rep_vector(0.0, N);
    mu += Intercept;
    for (n in 1:N) {
      // add more terms to the linear predictor
      mu[n] += r_1_1[J_1[n]] * Z_1_1[n];
    }
    target += normal_lpdf(Y | mu, sigma);
  }
  // priors including constants
  target += lprior;
  target += std_normal_lpdf(z_1[1]);
}
generated quantities {
  // actual population-level intercept
  real b_Intercept = Intercept;
}
```

brms generated Stan code

```
model {
  // likelihood including constants
  if (!prior_only) {
    // initialize linear predictor term
    vector[N] mu = rep_vector(0.0, N);
    mu += Intercept;
    for (n in 1:N) {
      // add more terms to the linear predictor
      mu[n] += r_1_1[J_1[n]] * Z_1_1[n];
    }
    target += normal_lpdf(Y | mu, sigma);
  }
  // priors including constants
  target += lprior;
  target += std_normal_lpdf(z_1[1]);
}
generated quantities {
  // actual population-level intercept
  real b_Intercept = Intercept;
}
```

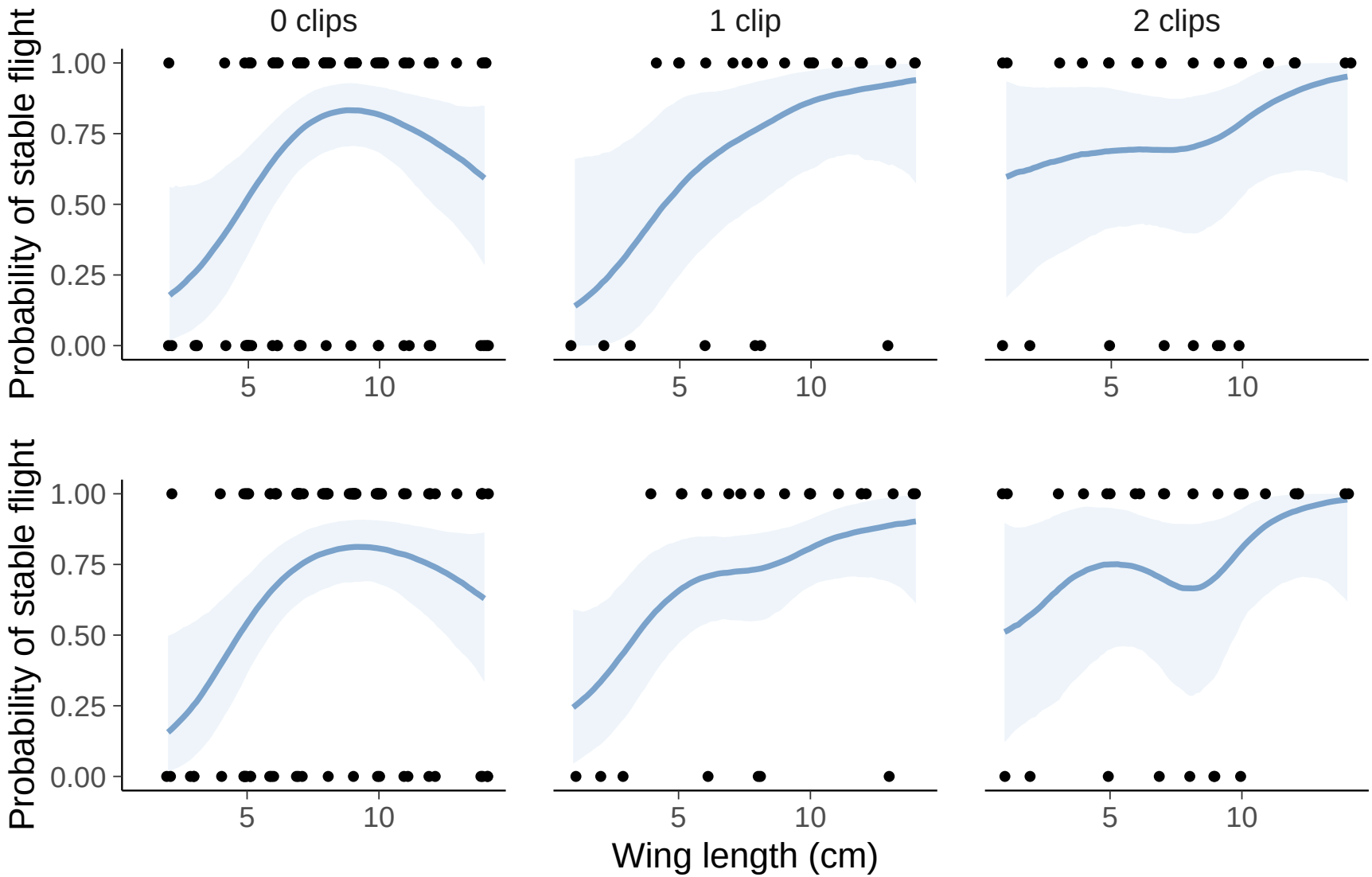
Paper helicopters: flight time

Separate model vs. hierarchical model



Paper helicopters: stability

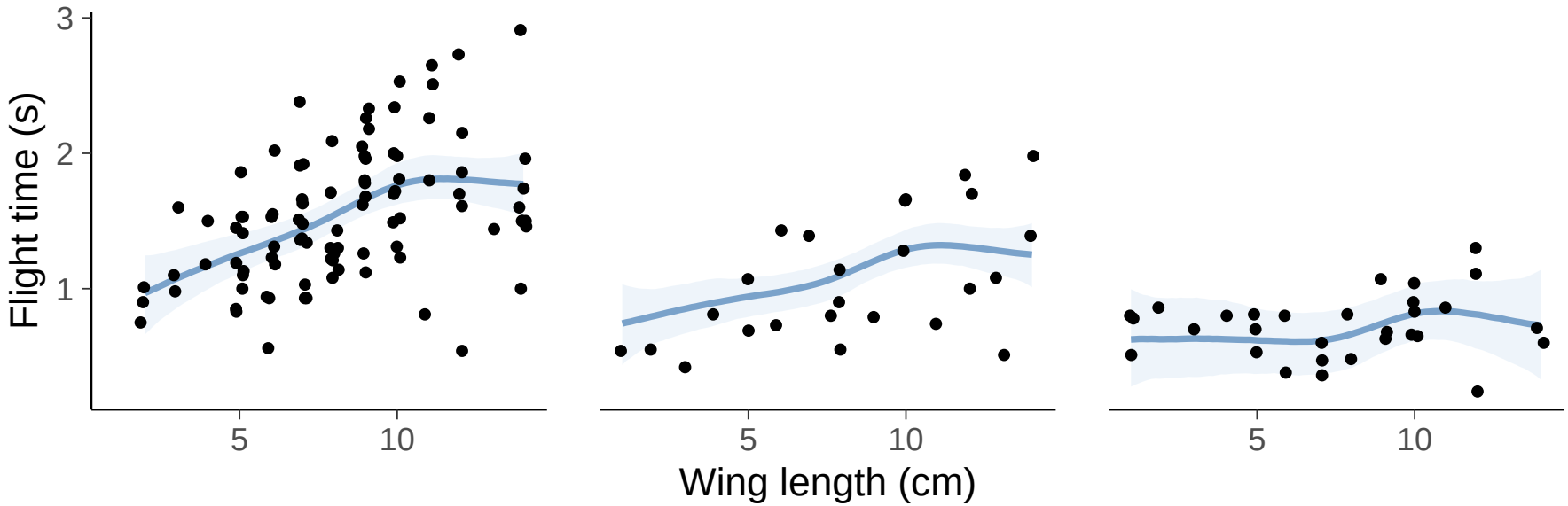
Separate model vs. hierarchical model



Paper helicopters: brms

Flight time

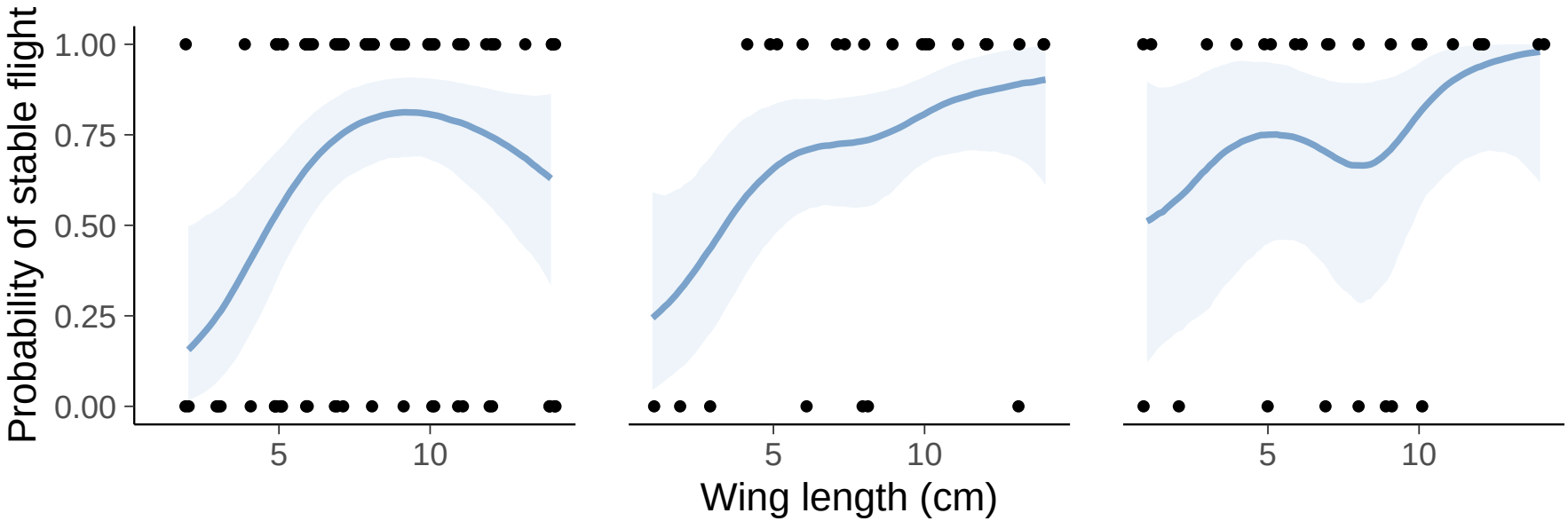
```
flight_time ~ s(wing_length) + s(wing_length, by = nclips)
```



Paper helicopters: brms

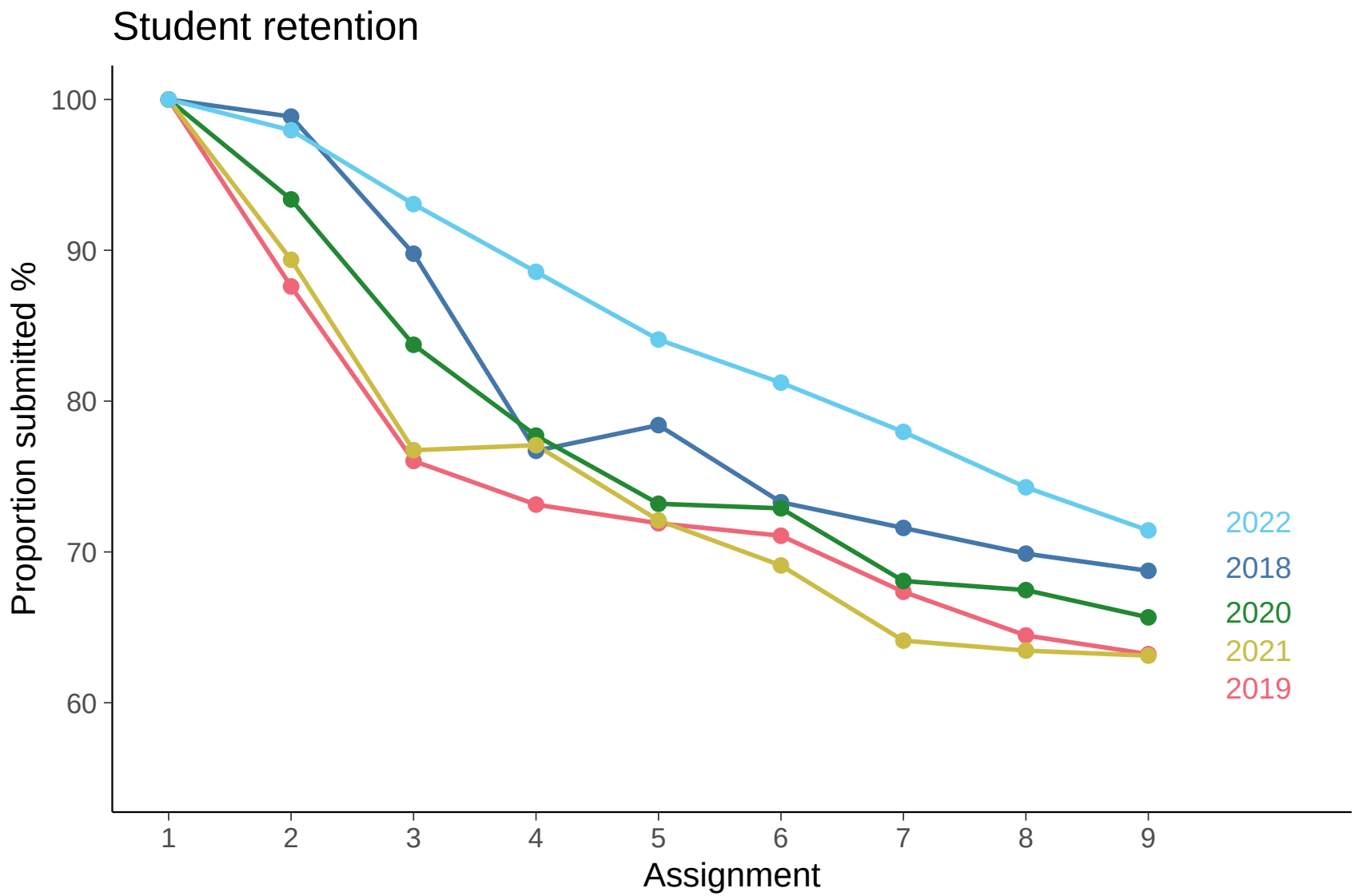
Stability

```
stable_flight ~ s(wing_length) + s(wing_length, by = nclips),
family = bernoulli()
```

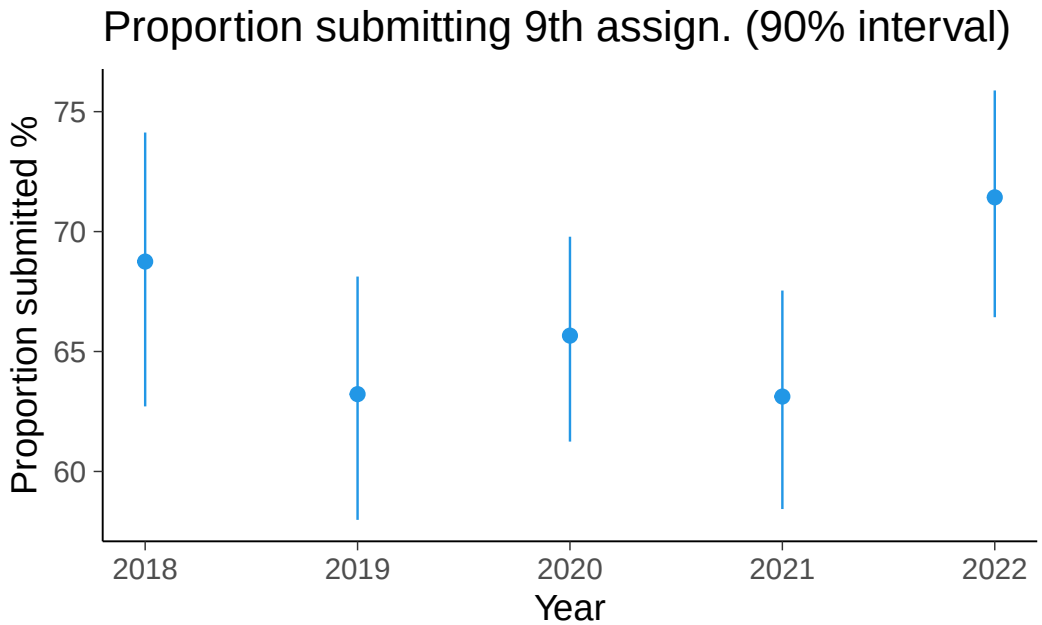


Student retention

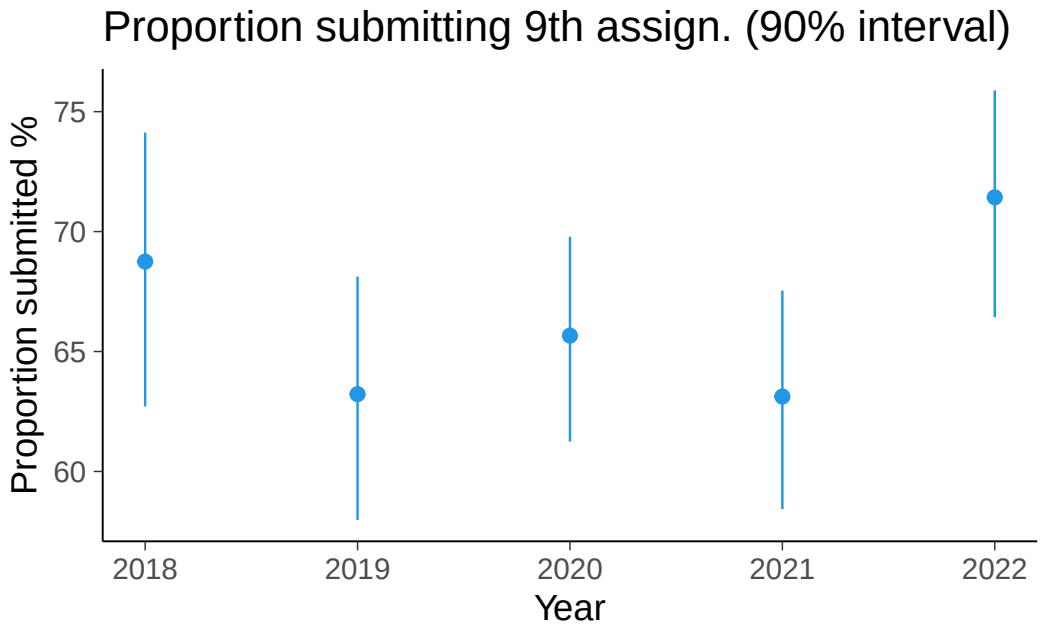
Was year 2022 better than earlier year?

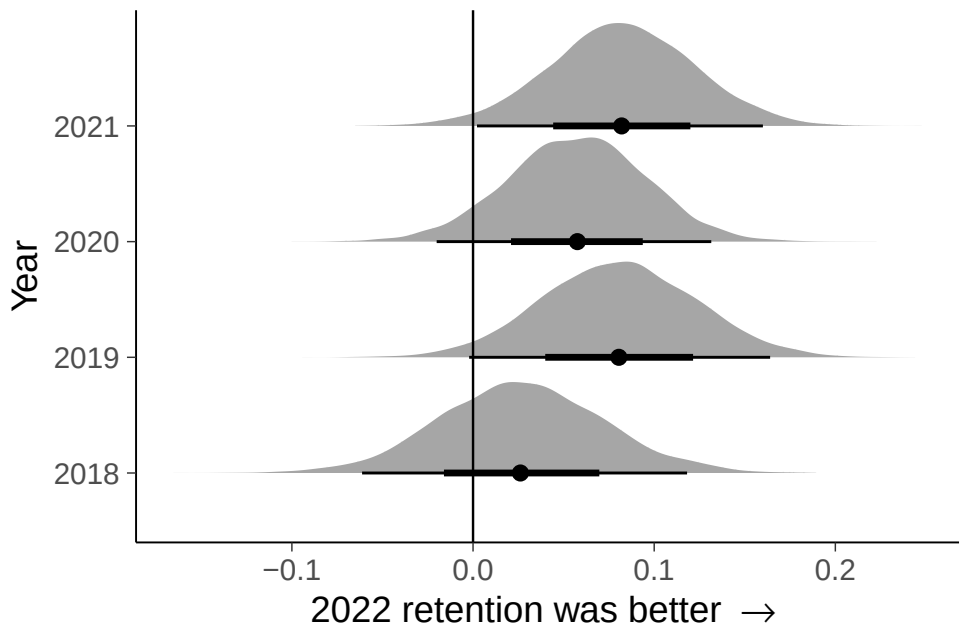


Student retention separate model

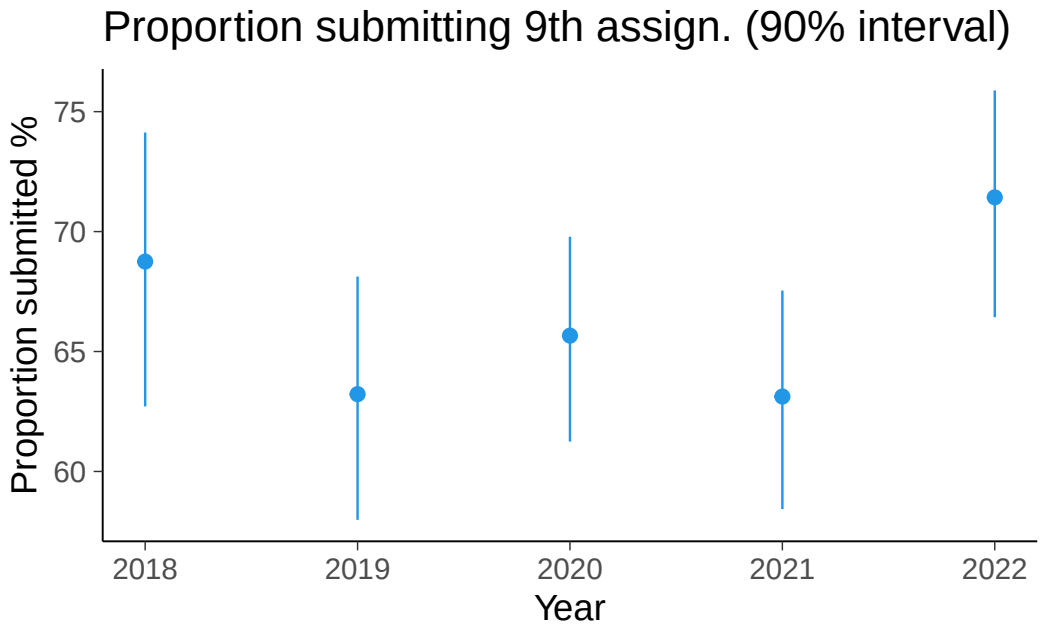


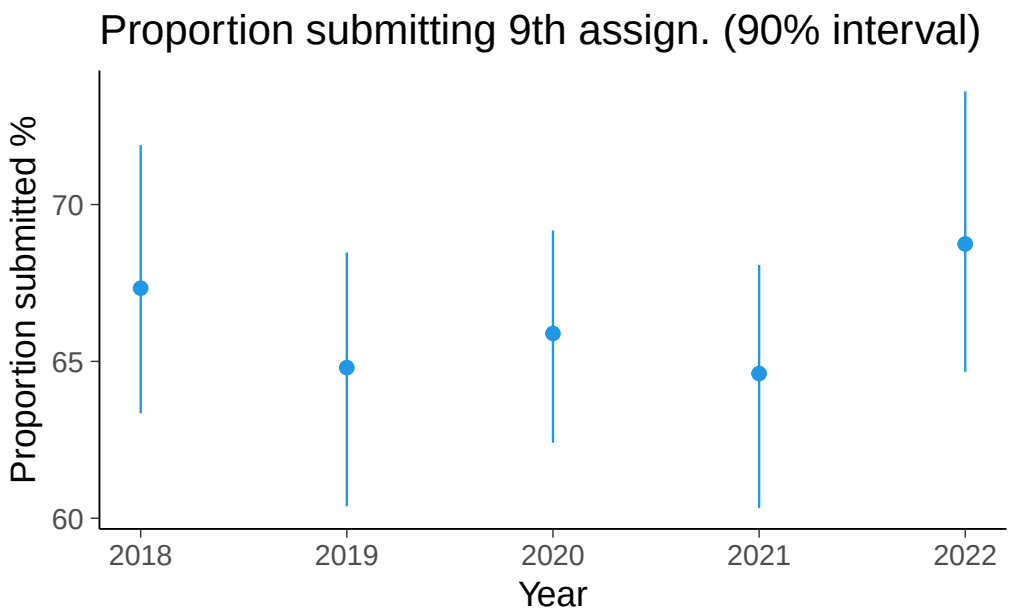
Student retention separate model



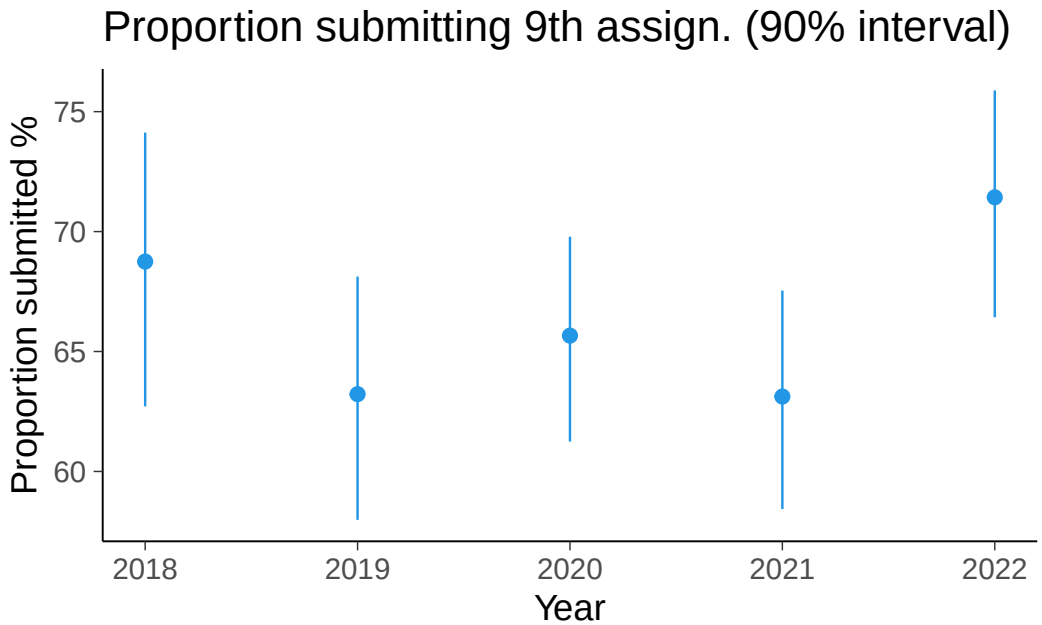


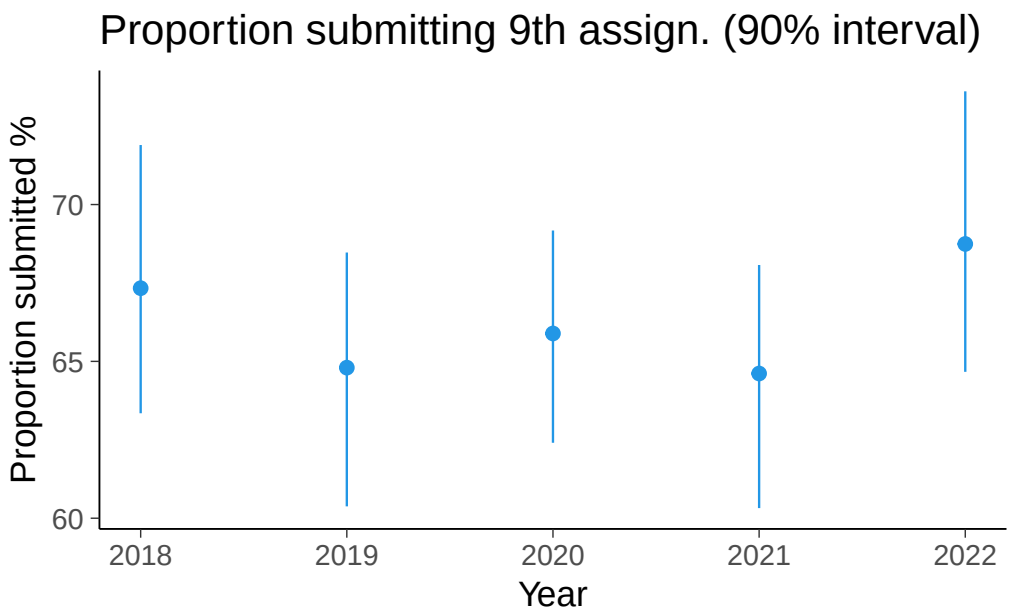
Student retention separate vs hierarchical model





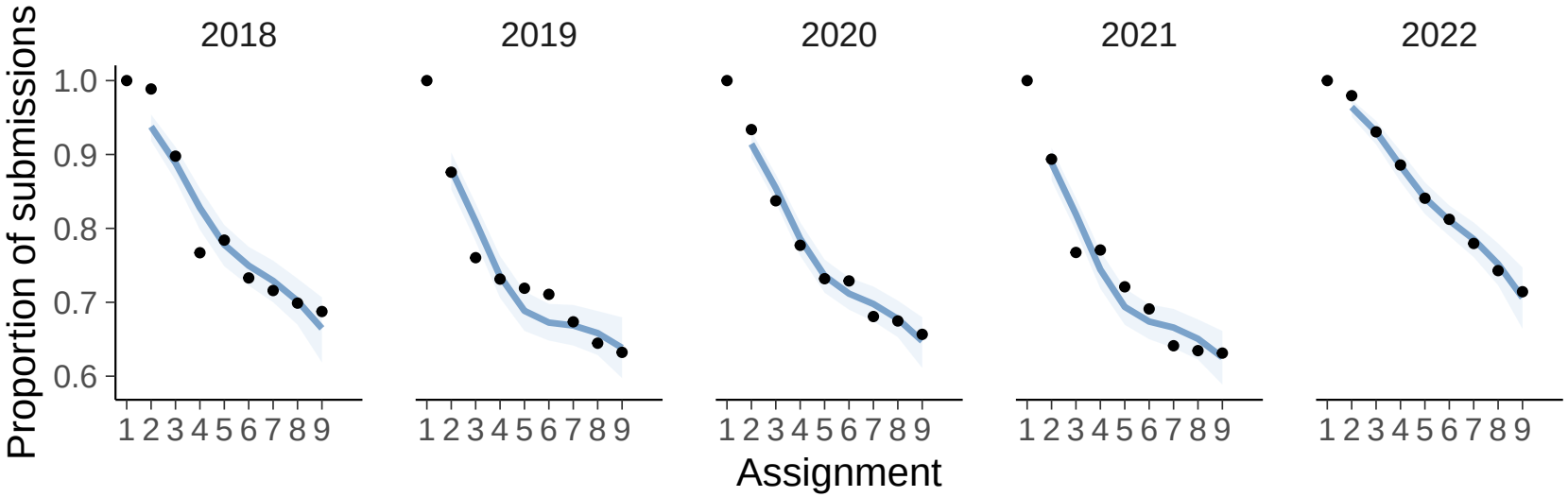
Student retention separate vs hierarchical model





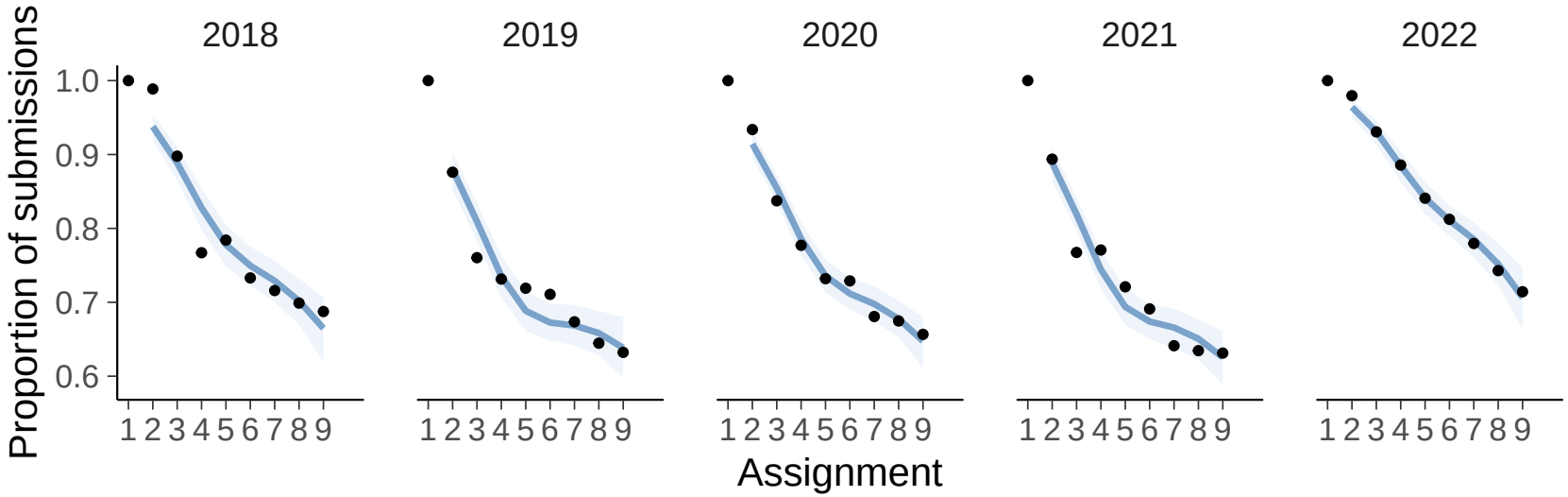

```
nstudents | trials(nstudents1) ~ 1 + (1 | year), family=binomial()
```

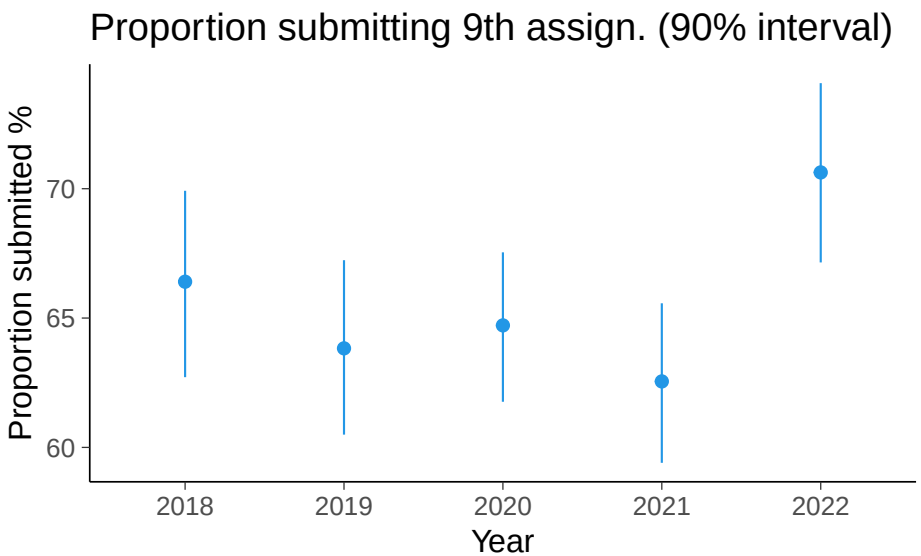
Student retention latent spline model



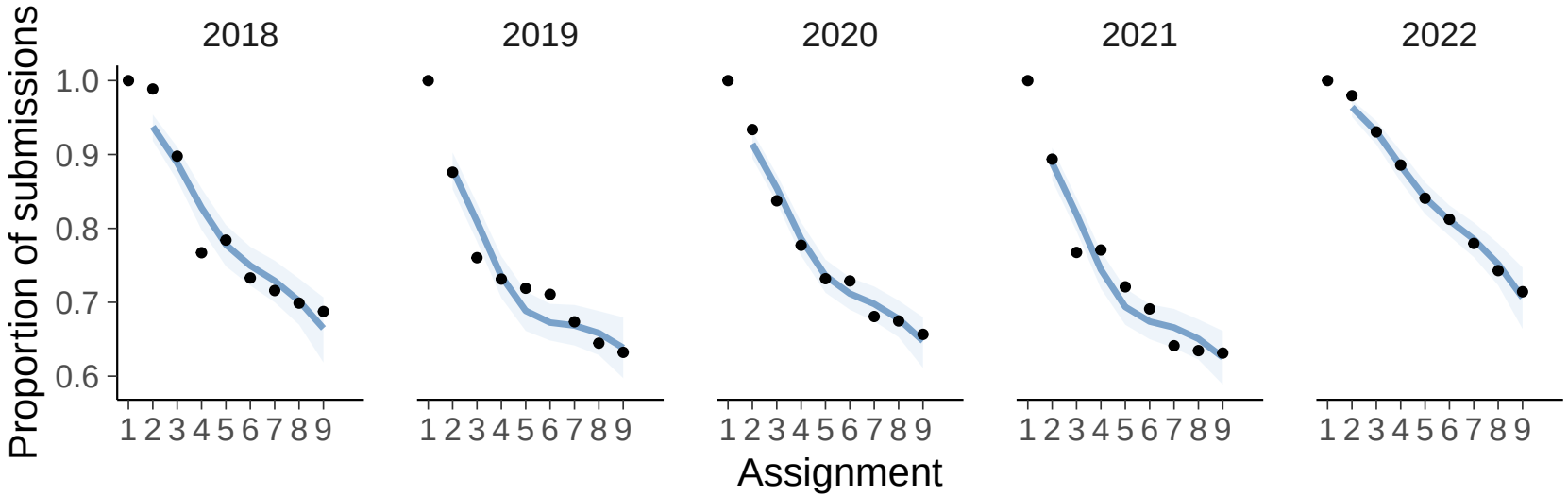

```
nstudents | trials(nstudents1) ~ s(assignment, k=4) + (assignment | year),
family=binomial()
```

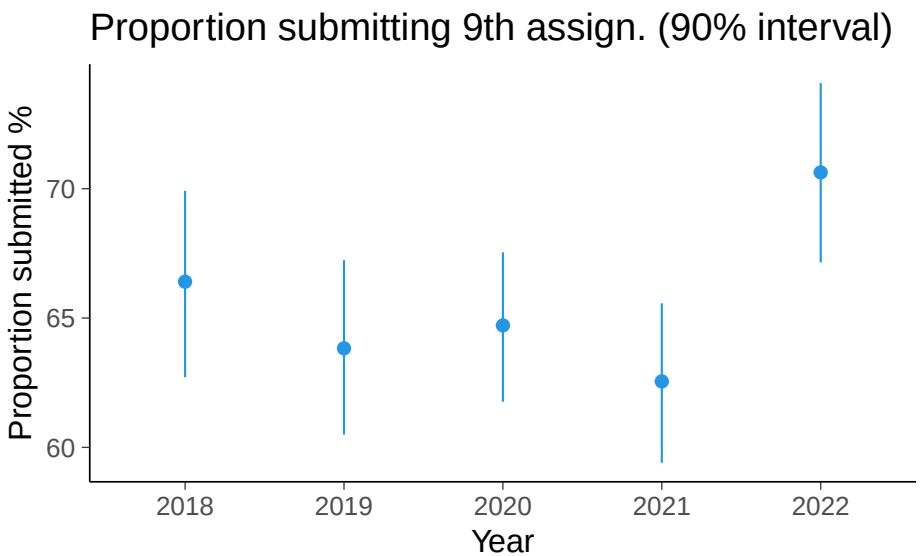
Student retention latent spline model

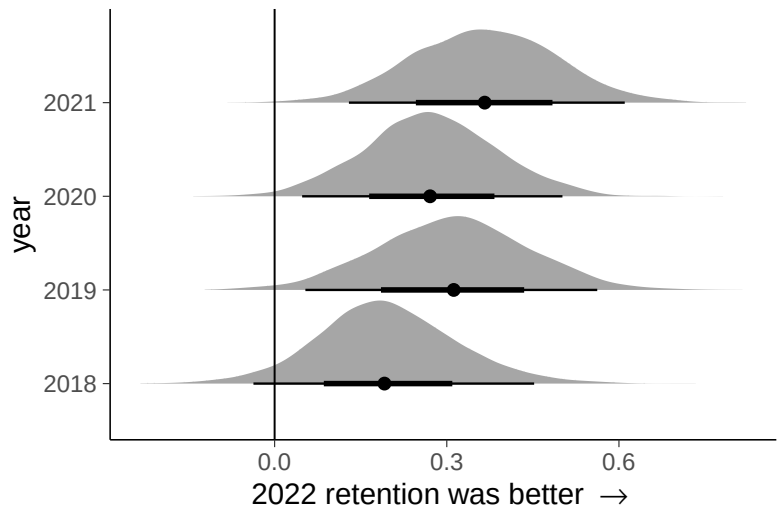




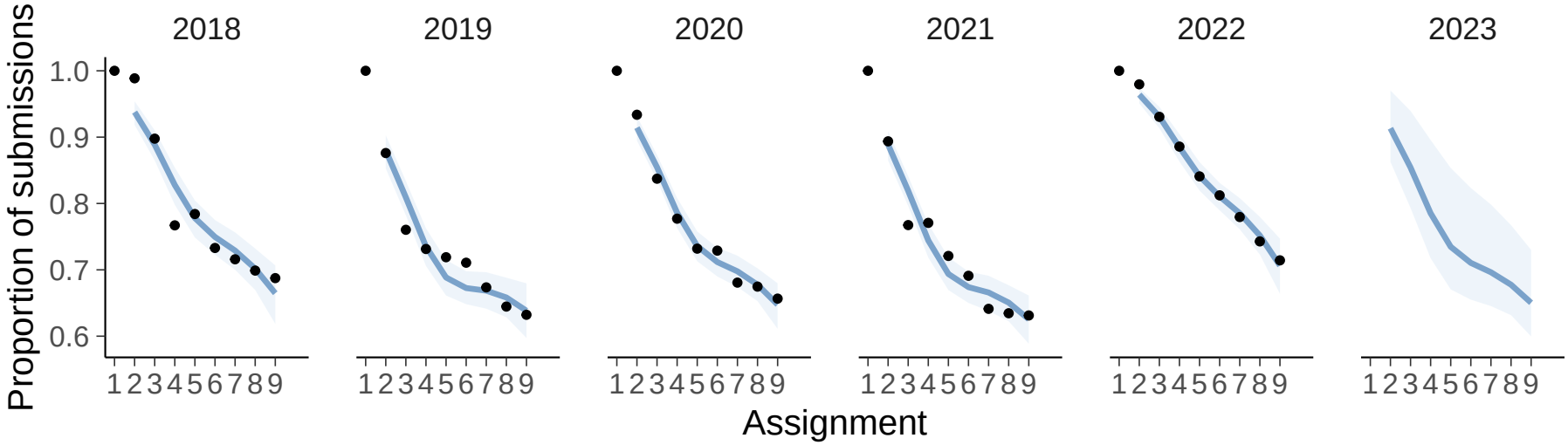
Student retention latent spline model



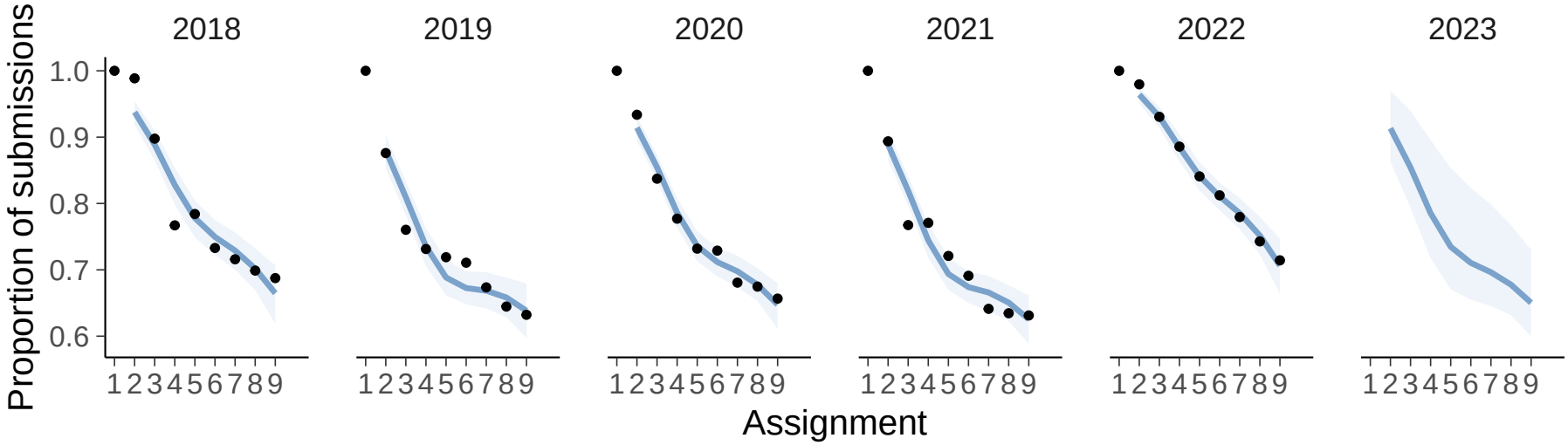


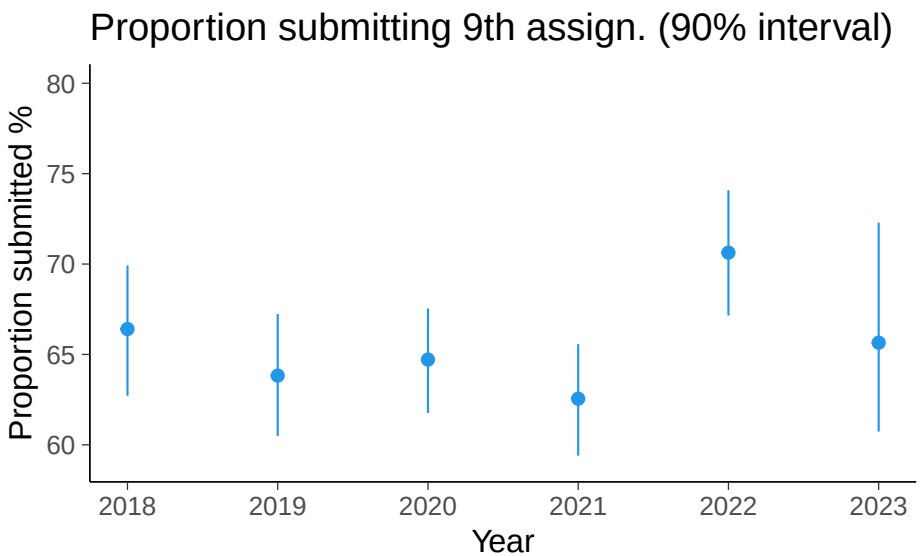


Student retention latent spline model, year 2023?

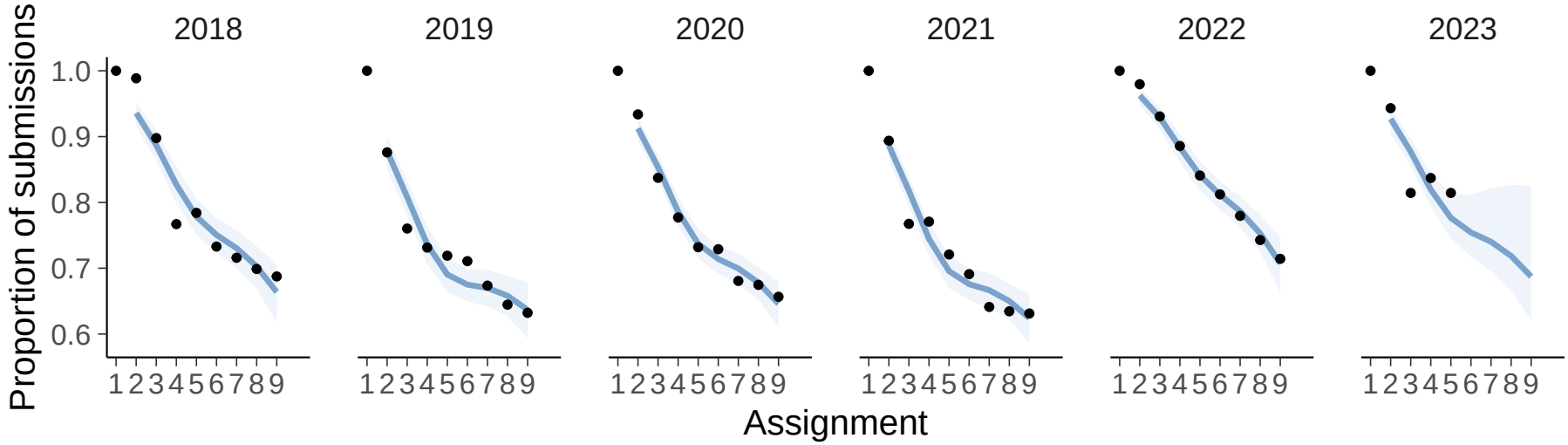


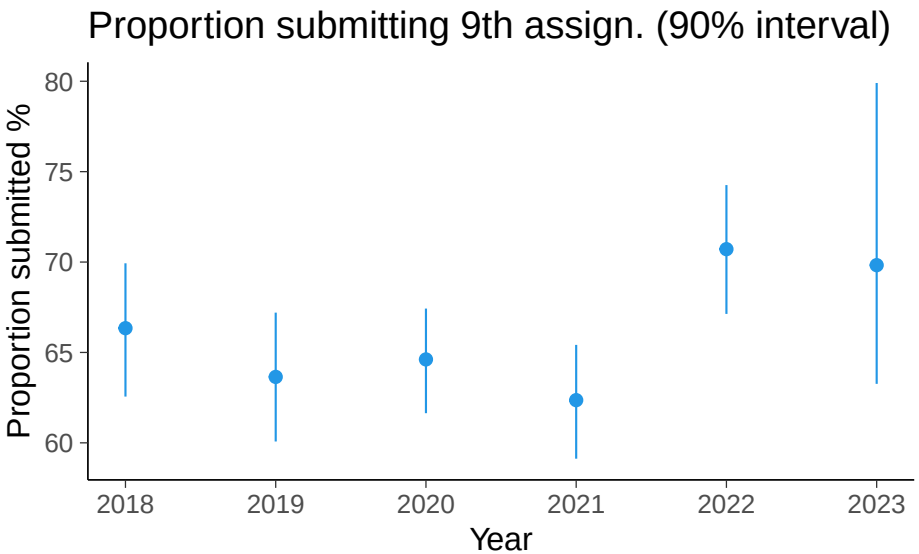
Student retention latent spline model, year 2023?



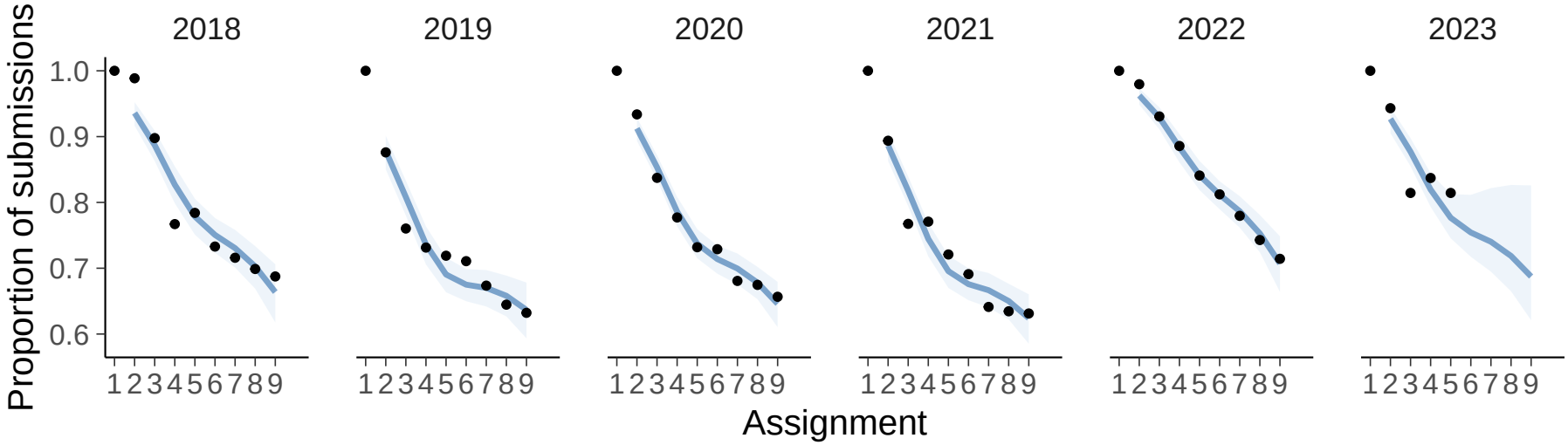


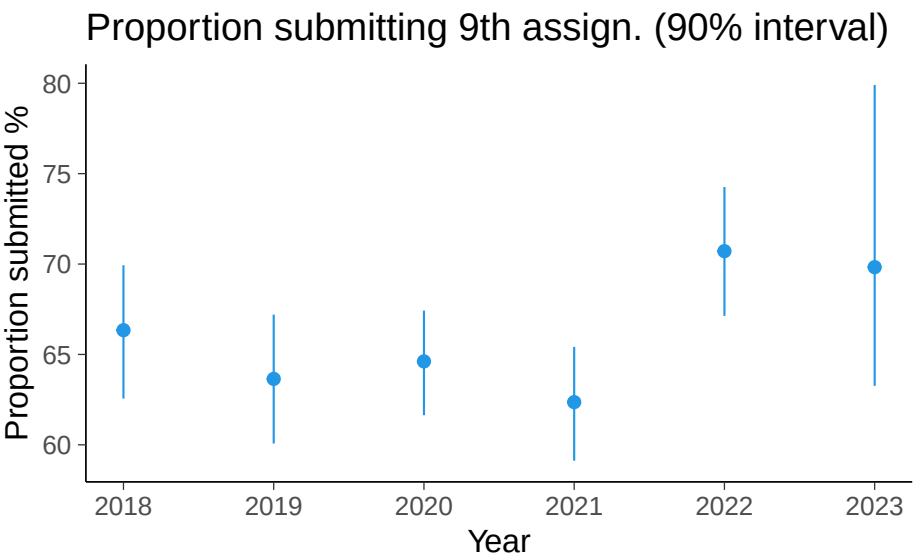
Student retention latent spline model, year 2023?

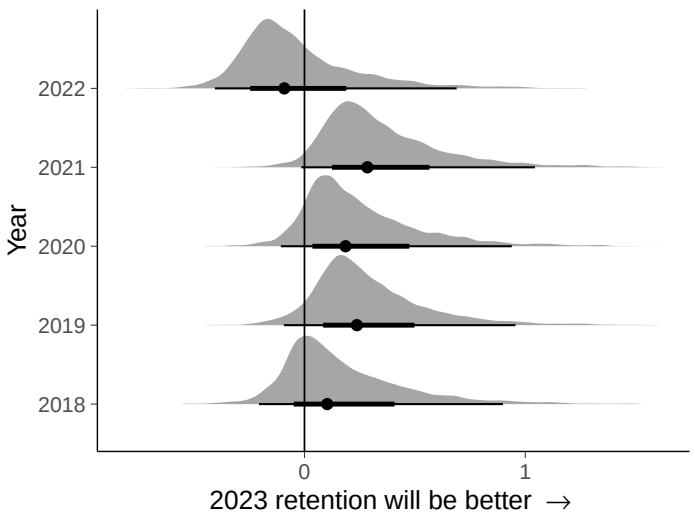




Student retention latent spline model, year 2023?







brms summary: one varying coefficient

```
nstudents | trials(nstudents1) ~ 1 + (1 | year), family=binomial()
```

```
 Family: binomial
  Links: mu = logit
Formula: nstudents | trials(nstudents1) ~ 1 + (1 | year)
   Data: filter(tb, assignment == 9) (Number of observations: 5)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Multilevel Hyperparameters:
~year (Number of levels: 5)
              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)     0.16      0.13     0.01     0.50 1.00      667      878

Regression Coefficients:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept     0.68      0.11     0.45     0.92 1.01      390      242
```

brms summary: two varying coefficients, nocor

```
   nstudents | trials(nstudents1) ~ assignment + (assignment || year), ...
```

```
 Family: binomial
  Links: mu = logit
Formula: nstudents | trials(nstudents1) ~ assignment + (assignment || year)
   Data: filter(tb, assignment > 1) (Number of observations: 40)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000
```

```
Multilevel Hyperparameters:
~year (Number of levels: 5)
               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)      0.79      0.45     0.30     1.95 1.00      972     1470
sd(assignment)     0.07      0.06     0.01     0.23 1.01      592      720
```

```
Regression Coefficients:
           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept      2.37      0.38     1.58     3.14 1.01     1114     1471
assignment    -0.21      0.04    -0.30    -0.14 1.00     1056      803
```

brms summary: two varying coefficients

```
   nstudents | trials(nstudents1) ~ assignment + (assignment | year), ...
```

```
 Family: binomial
  Links: mu = logit
Formula: nstudents | trials(nstudents1) ~ assignment + (assignment | year)
   Data: filter(tb, assignment > 1) (Number of observations: 40)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Multilevel Hyperparameters:
~year (Number of levels: 5)
                        Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)               0.76      0.35     0.34     1.69 1.00     1079     1487
sd(assignment)              0.06      0.03     0.02     0.15 1.00     1211     1979
cor(Intercept,assignment)  -0.85      0.23    -1.00    -0.20 1.00     1730     2265

Regression Coefficients:
           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept      2.36      0.37     1.60     3.10 1.00      860     1179
assignment    -0.21      0.03    -0.27    -0.14 1.00     1073     1464
```

brms summary: two varying coefficients

```
   nstudents | trials(nstudents1) ~ assignment + (assignment | year), ...
```

```
 Family: binomial
  Links: mu = logit
Formula: nstudents | trials(nstudents1) ~ assignment + (assignment | year)
   Data: filter(tb, assignment > 1) (Number of observations: 40)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Multilevel Hyperparameters:
~year (Number of levels: 5)
                        Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)               0.76      0.35     0.34     1.69 1.00     1079     1487
sd(assignment)              0.06      0.03     0.02     0.15 1.00     1211     1979
cor(Intercept,assignment)  -0.85      0.23    -1.00    -0.20 1.00     1730     2265

Regression Coefficients:
           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept      2.36      0.37     1.60     3.10 1.00      860     1179
assignment    -0.21      0.03    -0.27    -0.14 1.00     1073     1464
```

brms uses by default multivariate normal population prior for multiple varying coefficients, with LKJ prior on the correlation matrix

Centered vs non-centered parameterization

HMC divergences are more likely when using hierarchical models

Centered parameterization

Hierarchical model code from the course demos

```
data {
  int<lower=0> N;                     // number of observations
  int<lower=0> K;                     // number of groups
  array[N] int<lower=1, upper=K> x; // discrete group indicators
  vector[N] y;                        // real valued observations
}
```

Centered parameterization

Hierarchical model code from the course demos

```
data {
  int<lower=0> N;                     // number of observations
  int<lower=0> K;                     // number of groups
  array[N] int<lower=1, upper=K> x; // discrete group indicators
  vector[N] y;                        // real valued observations
}
parameters {
  real mu0;                           // prior mean
  real<lower=0> sigma0;               // prior std constrained to be pos.
  vector[K] mu;                       // group means
  real<lower=0> sigma;                // common std constrained to be pos.
}
```

Centered parameterization

Hierarchical model code from the course demos

```
data {
  int<lower=0> N;                       // number of observations
  int<lower=0> K;                       // number of groups
  array[N] int<lower=1, upper=K> x; // discrete group indicators
  vector[N] y;                          // real valued observations
}
parameters {
  real mu0;                             // prior mean
  real<lower=0> sigma0;                 // prior std constrained to be pos.
  vector[K] mu;                         // group means
  real<lower=0> sigma;                  // common std constrained to be pos.
}
model {
  mu0 ~ normal(10, 10);                 // weakly informative prior
  sigma0 ~ normal(0, 10);               // weakly informative prior
  mu ~ normal(mu0, sigma0);             // population prior with unknown param.
  sigma ~ lognormal(0, .5);             // weakly informative prior
  y ~ normal(mu[x], sigma);             // observation model
}
```

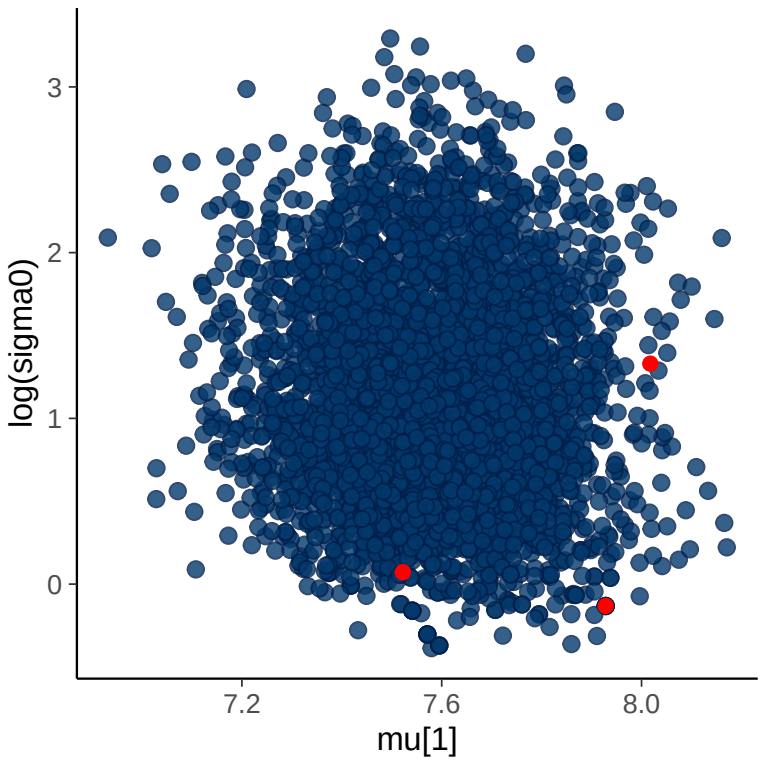
Centered parameterization

First data with many observations per group: 3 summer months with each having 71 observations.

Centered parameterization

First data with many observations per group: 3 summer months with each having 71 observations.

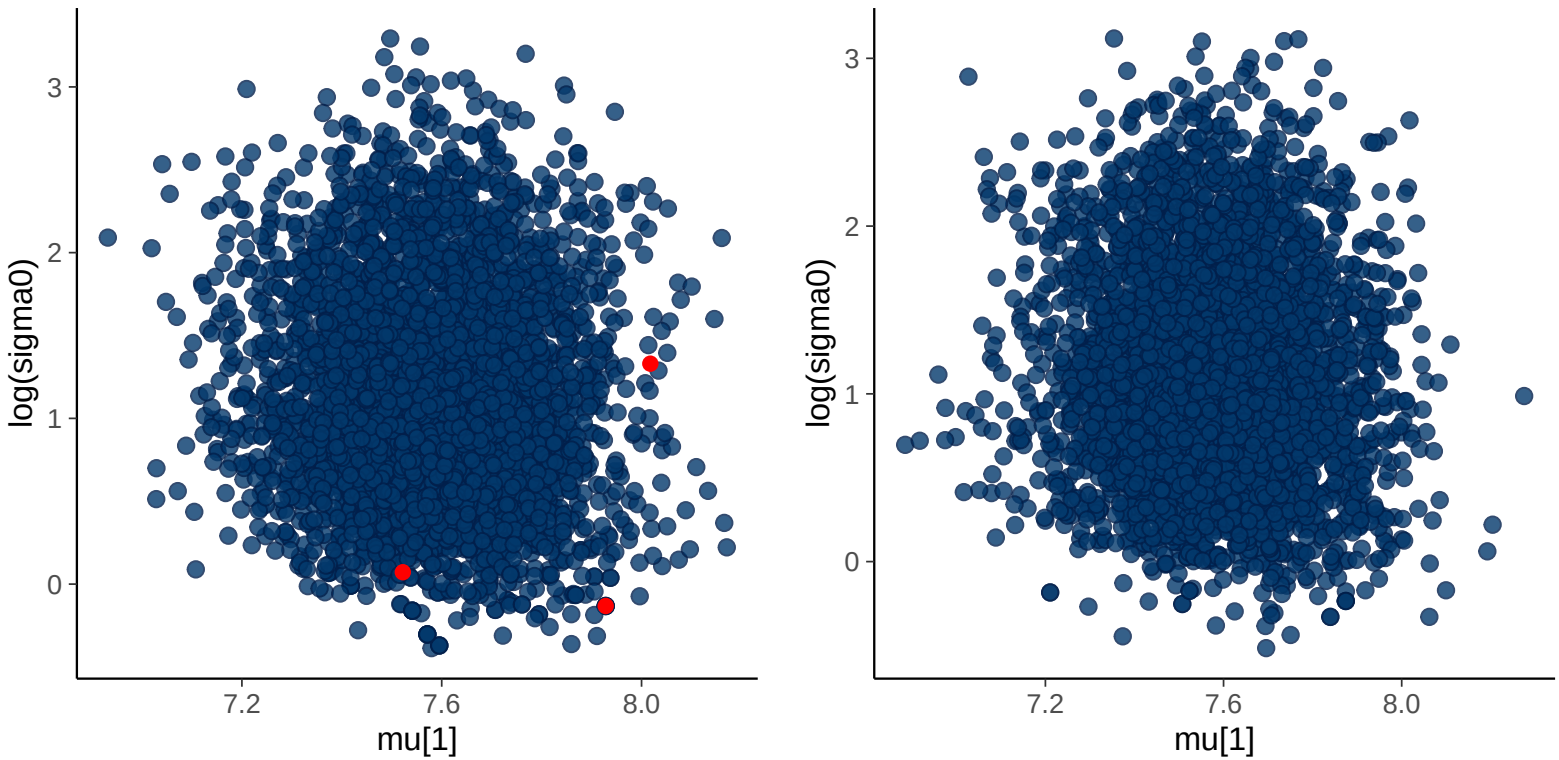
A few divergences that are not clustered.



Centered parameterization

First data with many observations per group: 3 summer months with each having 71 observations.

And decreasing step size a little helps.



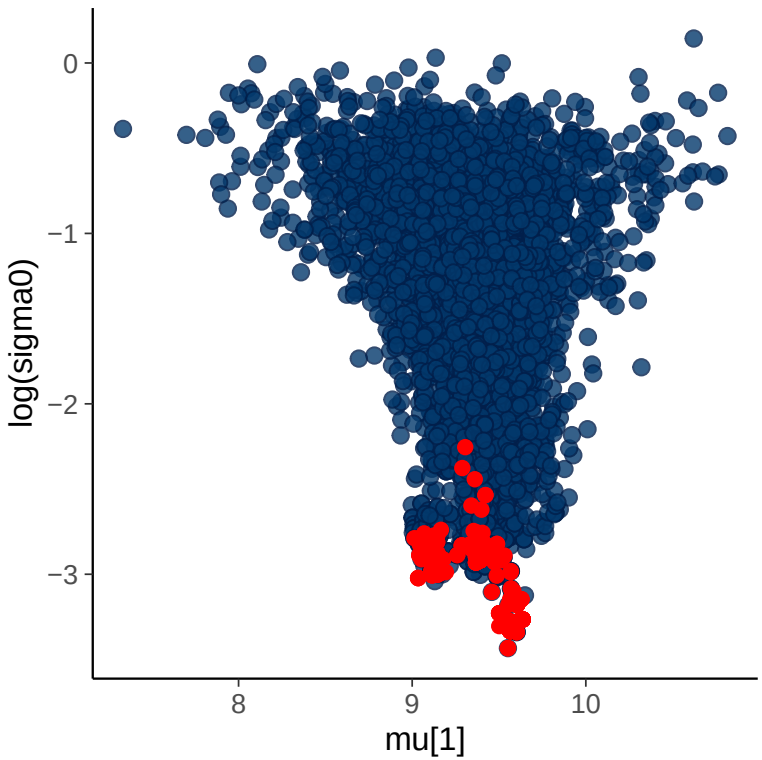
Centered parameterization

Second data with a few observations per group: 71 years with each having 3 observations.

Centered parameterization

Second data with a few observations per group: 71 years with each having 3 observations.

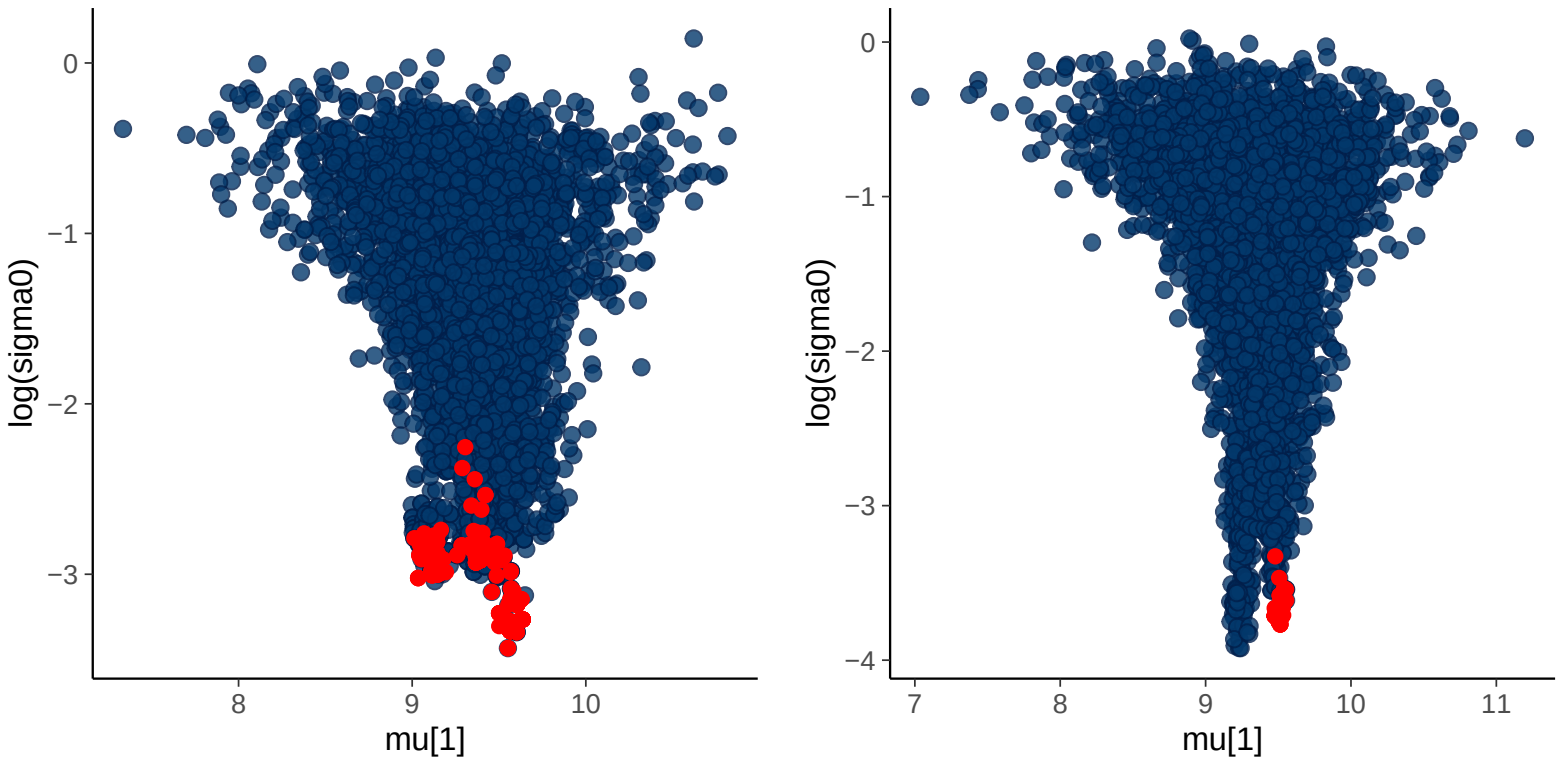
Many divergences that are clustered.



Centered parameterization

Second data with a few observations per group: 71 years with each having 3 observations.

And decreasing step size doesn't remove the problem.



Non-centered parameterization

Transformation

```
parameters {
  real mu0;                          // prior mean
  real<lower=0> sigma0;              // prior std constrained to be pos.
  vector[K] z;                       // latent variable
  real<lower=0> sigma;               // common std constrained to be pos.
}
transformed parameters {
  vector[K] mu = mu0 + sigma0 * z;  // group means
}
model {
  mu0 ~ normal(10, 10);      // weakly informative prior
  sigma0 ~ normal(0, 10);    // weakly informative prior
  z ~ normal(0, 1);          // unit normal
  sigma ~ lognormal(0, .5); // weakly informative prior
  y ~ normal(mu[x], sigma); // observation model
}
```

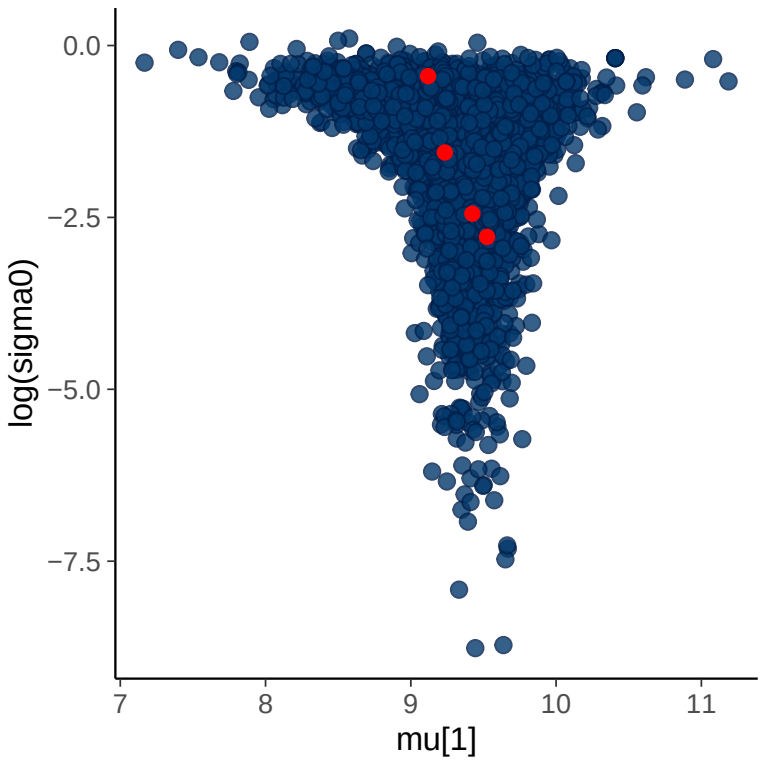
Non-centered parameterization

Second data with a few observations per group: 71 years with each having 3 observations.

Non-centered parameterization

Second data with a few observations per group: 71 years with each having 3 observations.

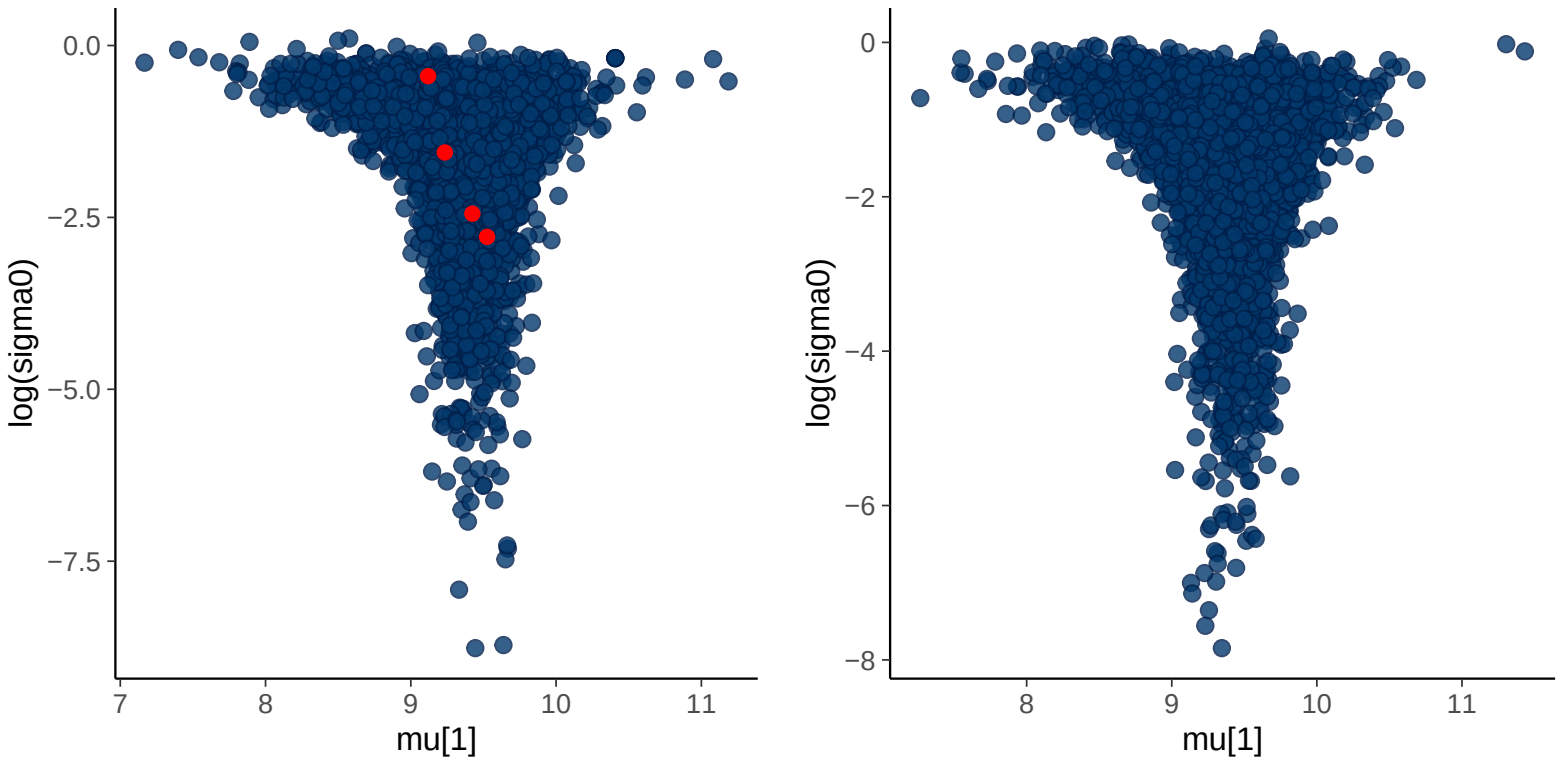
A few divergences that are not clustered.



Non-centered parameterization

Second data with a few observations per group: 71 years with each having 3 observations.

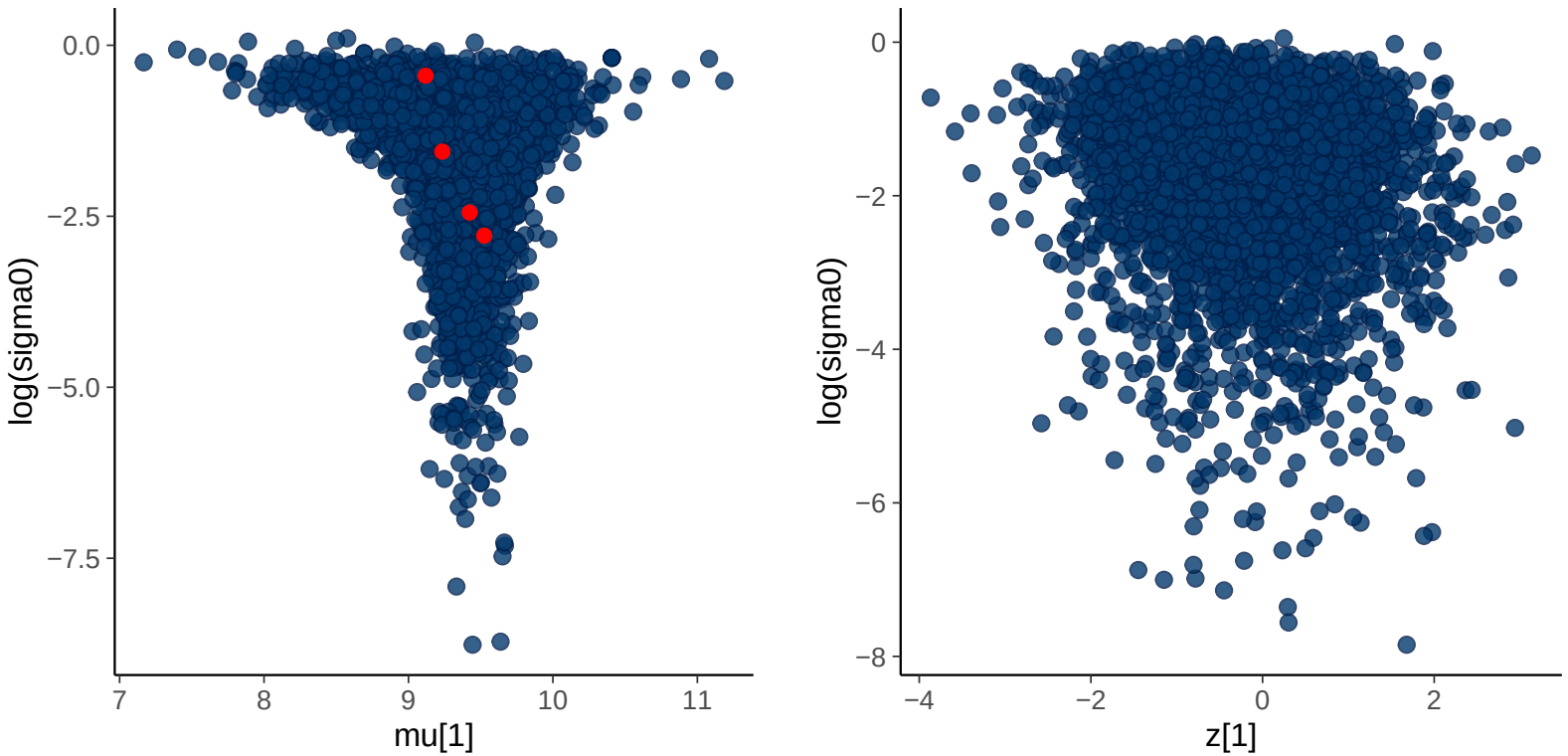
And decreasing step size a little helps.



Non-centered parameterization

Second data with a few observations per group: 71 years with each having 3 observations.

Because we're actually sampling `z` and not `mu`



Non-centered parameterization

No free lunch

- non-centered parameterization is good when likelihood is weak
- non-centered parameterization is bad when likelihood is strong

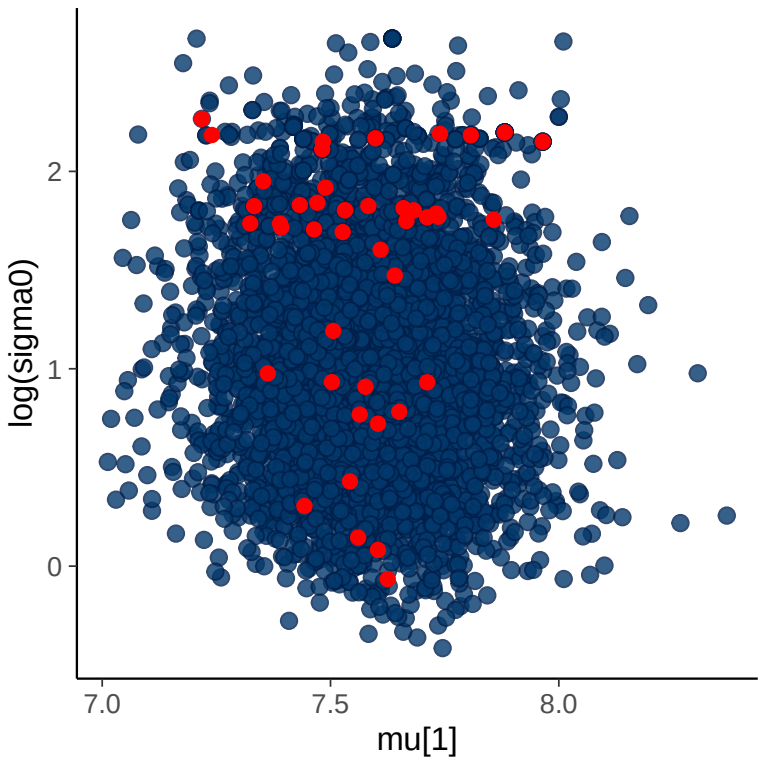
Non-centered parameterization

First data with many observations per group: 3 summer months with each having 71 observations.

Non-centered parameterization

First data with many observations per group: 3 summer months with each having 71 observations.

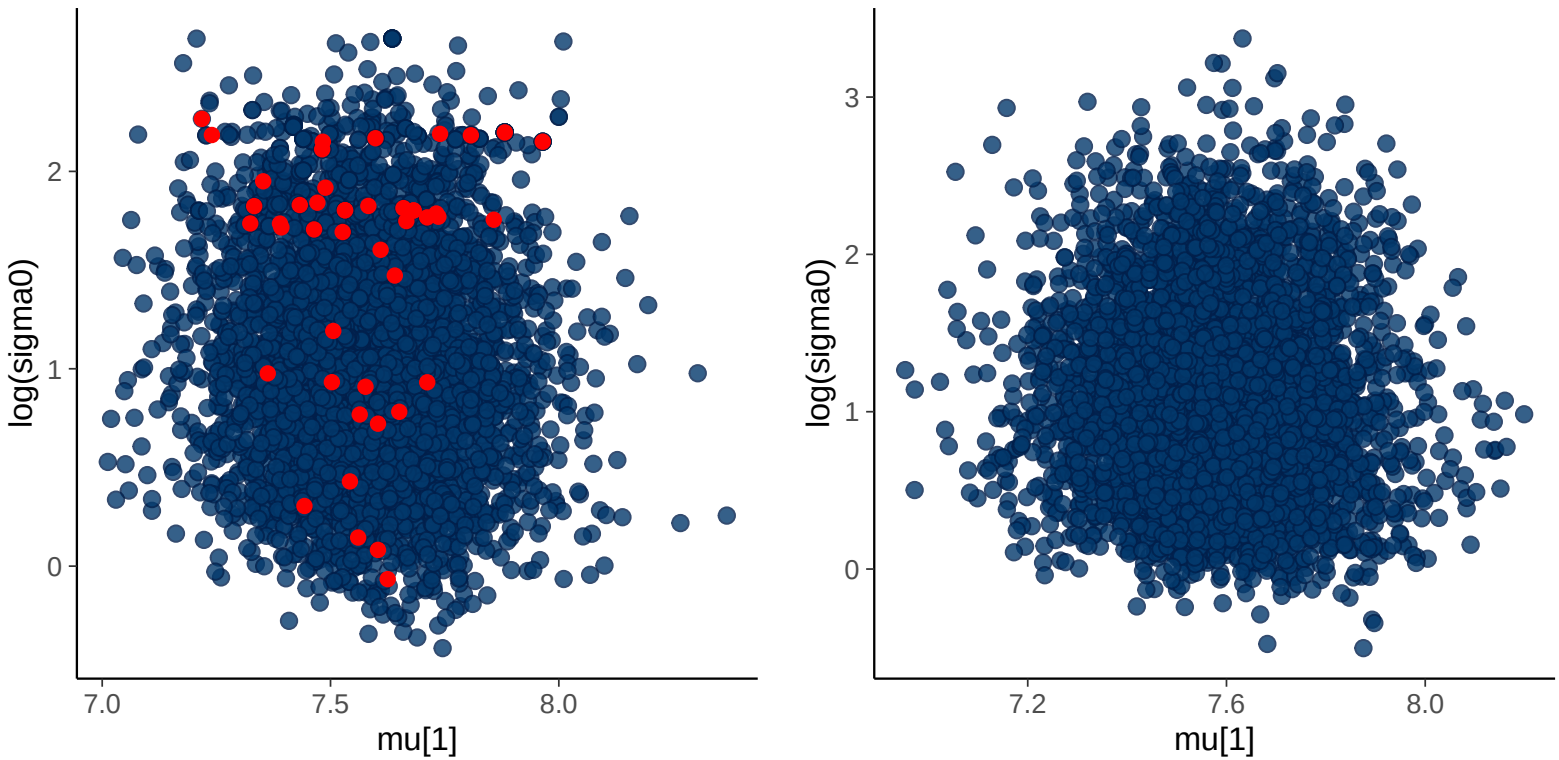
Many divergences that are not clustered.



Non-centered parameterization

First data with many observations per group: 3 summer months with each having 71 observations.

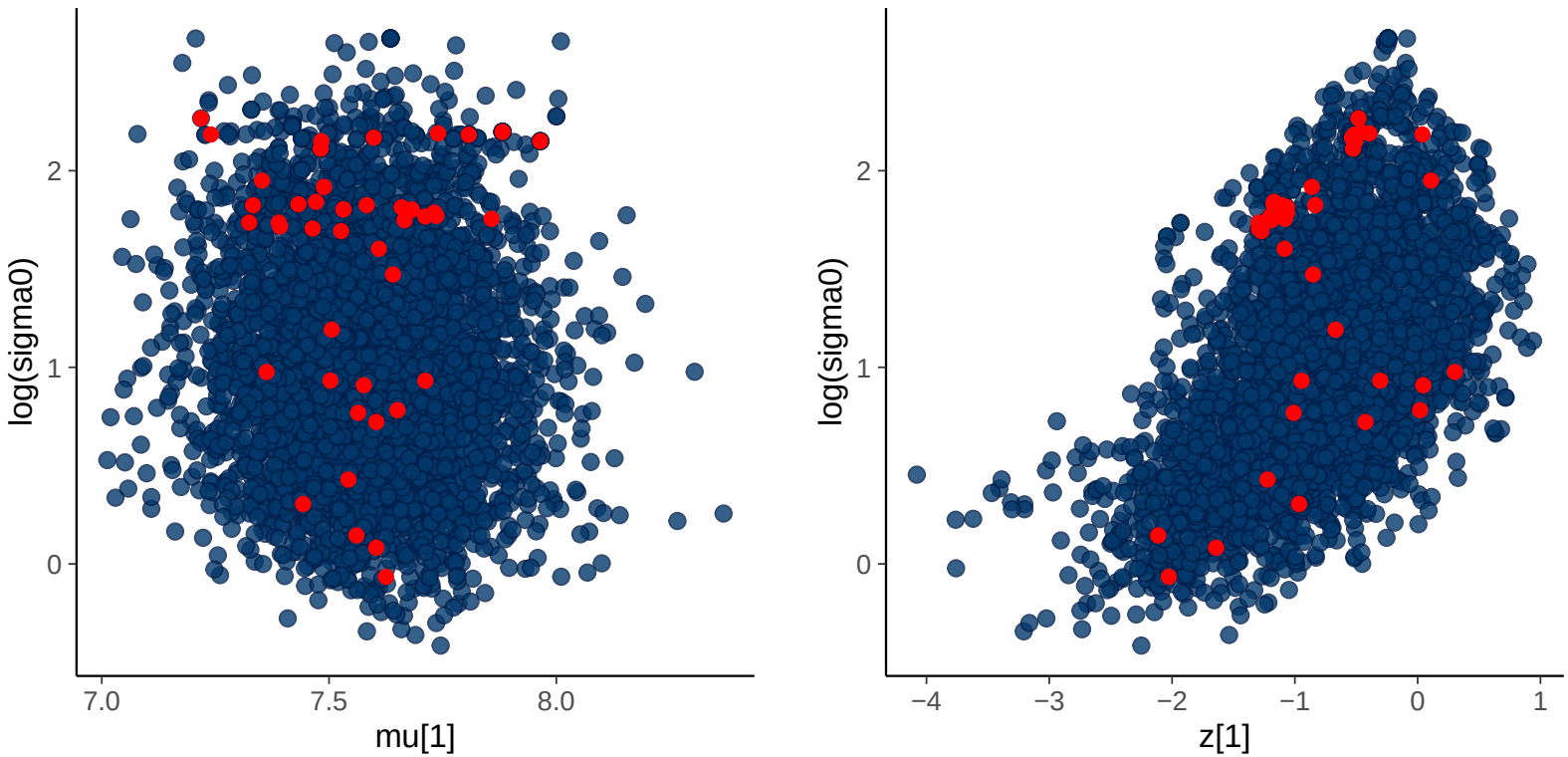
But decreasing step size a lot helps.



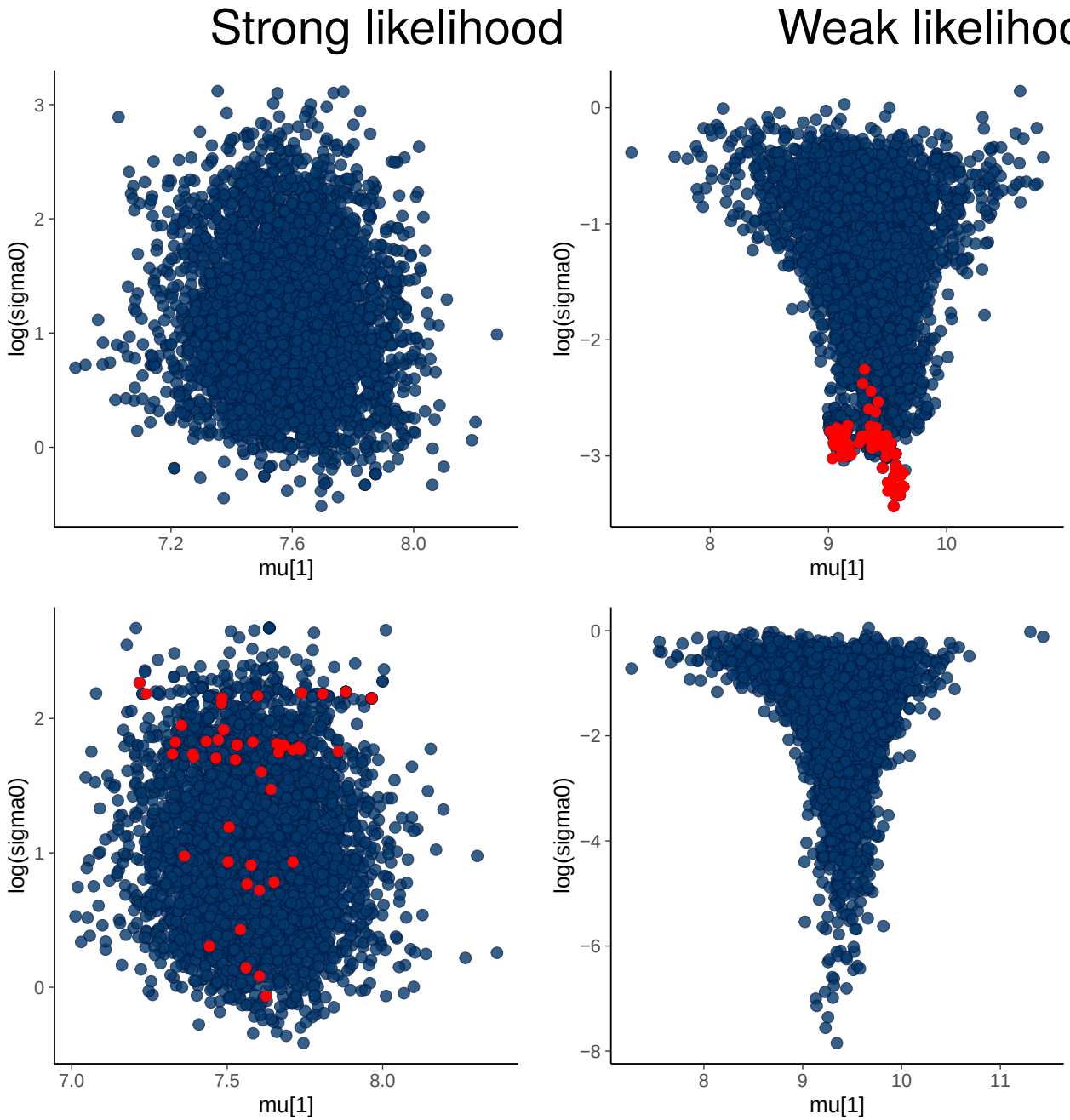
Non-centered parameterization

First data with many observations per group: 3 summer months with each having 71 observations.

Now the posterior for z is problematic.



Centered vs. non-centered parameterization



brms and rstanarm

- brms and rstanarm use non-centered parameterization
  - as hierarchical models and Bayesian inference is most useful when likelihood is weak

brms and rstanarm

- brms and rstanarm use non-centered parameterization
  - as hierarchical models and Bayesian inference is most useful when likelihood is weak
- If using brms and the likelihood very every group is strong and the non-centered parameterization is causing divergences
  - use non-hierarchical model as the hierarchical part is not that important with strong likelihood, that is, instead of

    ```
    y ~ 1 + (1 | group)
    ```

    use

    ```
    y ~ 1 + group
    ```

brms and rstanarm

- brms and rstanarm use non-centered parameterization
  - as hierarchical models and Bayesian inference is most useful when likelihood is weak
- If using brms and the likelihood very every group is strong and the non-centered parameterization is causing divergences
  - use non-hierarchical model as the hierarchical part is not that important with strong likelihood, that is, instead of

    ```
    y ~ 1 + (1 | group)
    ```

    use

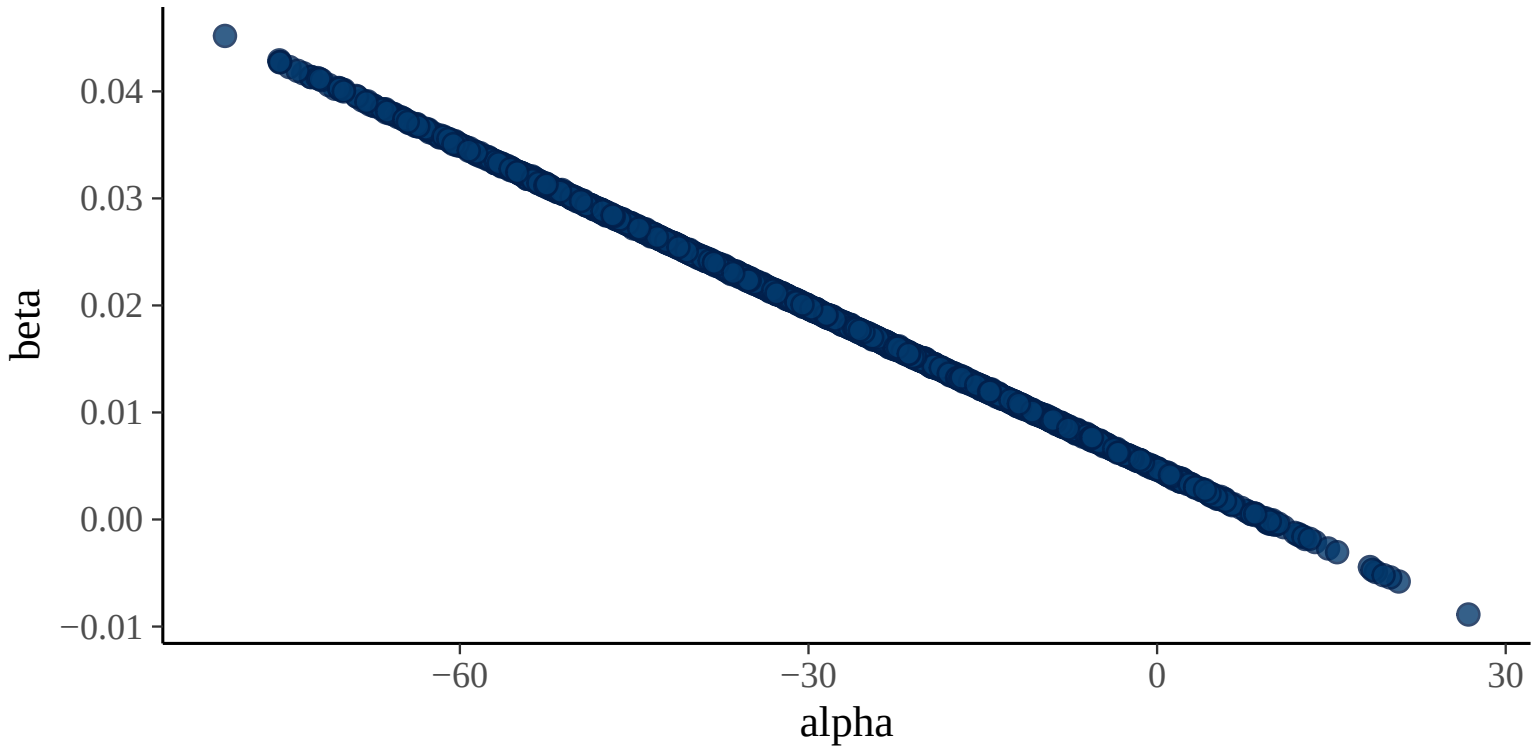
    ```
    y ~ 1 + group
    ```
- There can be need for both centered and non-centered parameterization in the same model
  - automation not easy, but research goes on

brms non-centered parameterization

```
parameters {
  real Intercept;  // temporary intercept for centered predictors
  real<lower=0> sigma;  // dispersion parameter
  vector<lower=0>[M_1] sd_1;  // group-level standard deviations
  array[M_1] vector[N_1] z_1;  // standardized group-level effects
}
transformed parameters {
  vector[N_1] r_1_1;  // actual group-level effects
  r_1_1 = (sd_1[1] * (z_1[1]));
//...
}
model {
//...
    for (n in 1:N) {
      // add more terms to the linear predictor
      mu[n] += r_1_1[J_1[n]] * Z_1_1[n];
    }
//...
  target += std_normal_lpdf(z_1[1]);
}
```

Kilpisjärvi summer temperature

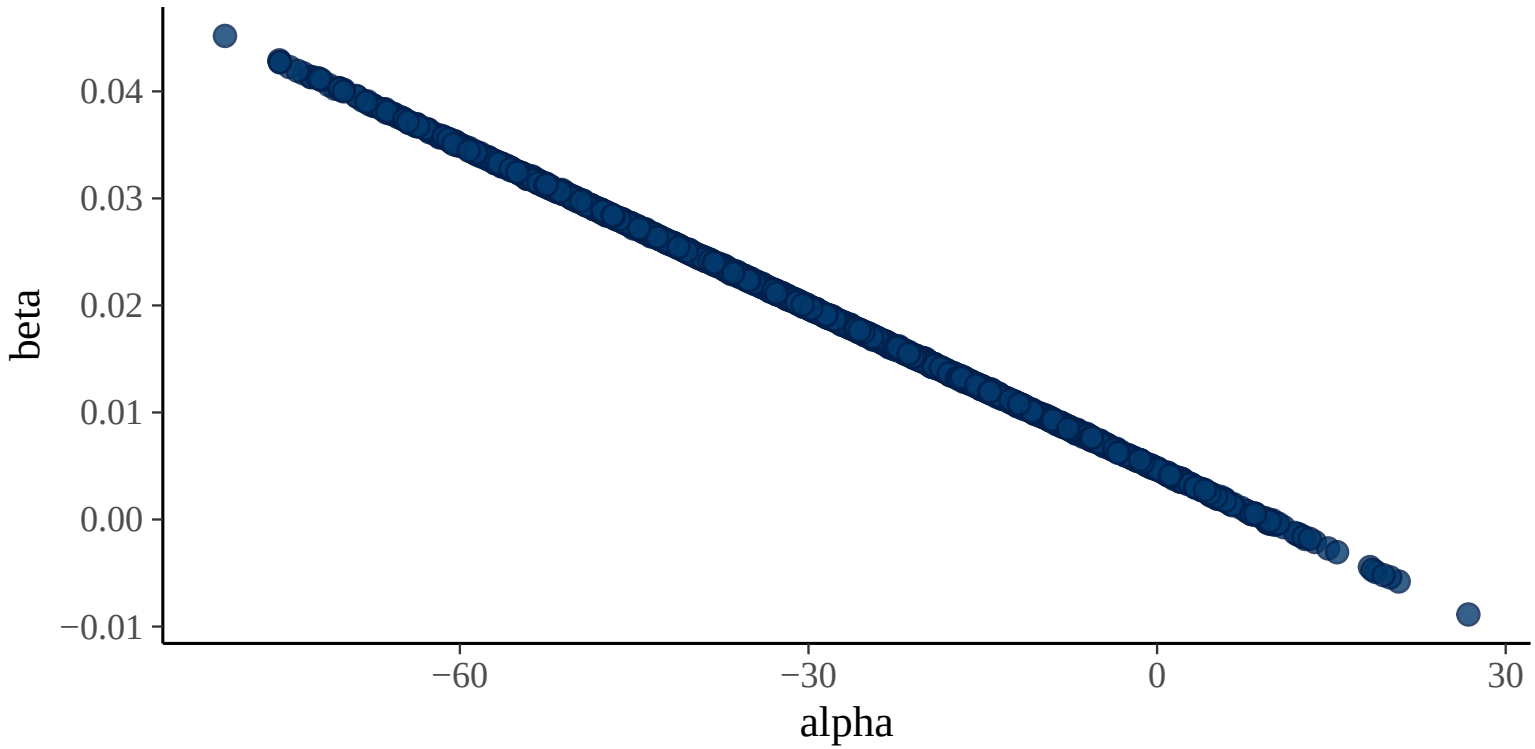
Posterior draws of alpha and beta




```
Warning: 1 of 4000 (0.0%) transitions hit the maximum treedepth limit of 10.
See https://mc-stan.org/misc/warnings for details.
```

Kilpisjärvi summer temperature

Posterior draws of alpha and beta




```
Warning: 1 of 4000 (0.0%) transitions hit the maximum treedepth limit of 10.
See https://mc-stan.org/misc/warnings for details.
```

Solution was to center the covariate time to have mean 0, so that the intercept is the expected temperature in the middle of the range

brms covariate centering by default

```
fit_lin <- brm(temp ~ year, data = data_lin, family = gaussian())
```

brms covariate centering by default

```
fit_lin <- brm(temp ~ year, data = data_lin, family = gaussian())
```

```
transformed data {
  matrix[N, Kc] Xc;  // centered version of X without an intercept
  vector[Kc] means_X;  // column means of X before centering
  for (i in 2:K) {
    means_X[i - 1] = mean(X[, i]);
    Xc[, i - 1] = X[, i] - means_X[i - 1];
  }
}
parameters {
  vector[Kc] b;  // regression coefficients
  real Intercept;  // temporary intercept for centered predictors
  real<lower=0> sigma;  // dispersion parameter
}
model {
//...
  target += normal_id_glm_lpdf(Y | Xc, Intercept, b, sigma);
//...
}
generated quantities {
  // actual population-level intercept
  real b_Intercept = Intercept - dot_product(means_X, b);
}
```

brms formulas

model	formula	alternative formula
intercept only $y \sim \mathrm{N}(\alpha, \sigma)$	y ~ 1	

brms formulas

model	formula	alternative formula
intercept only		
$y \sim \mathrm{N}(\alpha, \sigma)$	y ~ 1	
linear models		
$y \sim \mathrm{N}(\alpha + \beta x, \sigma)$	y ~ x	y ~ 1 + x

brms formulas

model	formula	alternative formula
intercept only		
$y \sim \mathrm{N}(\alpha, \sigma)$	`y ~ 1`	
linear models		
$y \sim \mathrm{N}(\alpha + \beta x, \sigma)$	`y ~ x`	`y ~ 1 + x`
$y \sim \mathrm{N}(\beta x, \sigma)$	`y ~ 0 + x`	`y ~ -1 + x`

brms formulas

model	formula	alternative formula
intercept only		
$y \sim \mathrm{N}(\alpha, \sigma)$	`y ~ 1`	
linear models		
$y \sim \mathrm{N}(\alpha + \beta x, \sigma)$	`y ~ x`	`y ~ 1 + x`
$y \sim \mathrm{N}(\beta x, \sigma)$	`y ~ 0 + x`	`y ~ -1 + x`
hierarchical models		
$y \sim \mathrm{N}(\alpha_0 + \alpha_g + \beta x, \sigma)$	`y ~ x + (1 | g)`	

brms formulas

model	formula	alternative formula
intercept only		
$y \sim \mathrm{N}(\alpha, \sigma)$	`y ~ 1`	
linear models		
$y \sim \mathrm{N}(\alpha + \beta x, \sigma)$	`y ~ x`	`y ~ 1 + x`
$y \sim \mathrm{N}(\beta x, \sigma)$	`y ~ 0 + x`	`y ~ -1 + x`
hierarchical models		
$y \sim \mathrm{N}(\alpha_0 + \alpha_g + \beta x, \sigma)$	`y ~ x + (1 | g)`	
$y \sim \mathrm{N}(\alpha_0 + \alpha_g + \beta_0 x + \beta_g x, \sigma)$	`y ~ x + (x | g)`	(see above)

brms formulas

model	formula
heteroskedastic	
$y \sim \mathrm{N}\left(\alpha_\mu + \beta_\mu x,\, \exp(\alpha_\sigma + \beta_\sigma x)\right)$	`bf(y ~ x, sigma ~ x)`

brms formulas

model	formula
heteroskedastic	
$y \sim \mathrm{N}\left(\alpha_\mu + \beta_\mu x,\, \exp(\alpha_\sigma + \beta_\sigma x)\right)$	`bf(y ~ x, sigma ~ x)`

brms families

`family` argument determines the observation model family

model	brms
$y \sim t_\nu(\alpha + \beta x, \sigma)$	`y ~ x, family = student()`
$y \sim \text{Bin}(\text{logit}^{-1}(\alpha + \beta x), N)$	`y | trials(N) ~ x, family = binomial()`
$y \sim \text{Neg-bin}(\exp(\alpha + \beta x), \phi)$	`y ~ x, family = negbinomial()`

BDA course demo for brms

Link in the course web site or directly

https://avehtari.github.io/BDA_R_demos/demos_rstan/brms_demo.html

Exchangeability

- Justifies why we can use
  - a joint model for data
  - a joint prior for a set of parameters
- Less strict than independence

Exchangeability

- *Exchangeability*: Parameters $\theta_1, \ldots, \theta_J$ (or observations $y_1, \ldots, y_J$) are exchangeable if the joint distribution p is invariant to the permutation of indices $(1, \ldots, J)$
- e.g.

$$p(\theta_1, \theta_2, \theta_3) = p(\theta_2, \theta_3, \theta_1)$$

- Exchangeability implies symmetry: If there is no information which can be used *a priori* to separate θ_j form each other, we can assume exchangeability. ("Ignorance implies exchangeability")

Exchangeability

- Exchangeability does not mean that the results of the experiments could not be different
  - e.g. if we know that the experiments have been in two different laboratories, and we know that the other laboratory has better conditions for the rats, but we do not know which experiments have been made in which laboratory
  - a priori experiments are exchangeable
  - model could have unknown parameter for the laboratory with a conditional prior for rats assumed to come form the same place (clustering model)

Exchangeability and additional information

- Example: bioassay
  - y_i number of dead animals are not exchangeable alone

Exchangeability and additional information

- Example: bioassay
  - y_i number of dead animals are not exchangeable alone
  - x_i dose is additional information

Exchangeability and additional information

- Example: bioassay
  - y_i number of dead animals are not exchangeable alone
  - x_i dose is additional information
  - (x_i, y_i) exchangeable and logistic regression was used

$$p(\alpha, \beta \mid y, n, x) \propto \prod_{i=1}^{n} p(y_i \mid \alpha, \beta, n_i, x_i) p(\alpha, \beta)$$

Hierarchical exchangeability

- Example: hierarchical rats example
  - all rats not exchangeable

Hierarchical exchangeability

- Example: hierarchical rats example
  - all rats not exchangeable
  - in a single laboratory rats exchangeable

Hierarchical exchangeability

- Example: hierarchical rats example
  - all rats not exchangeable
  - in a single laboratory rats exchangeable
  - laboratories exchangeable

Hierarchical exchangeability

- Example: hierarchical rats example
  - all rats not exchangeable
  - in a single laboratory rats exchangeable
  - laboratories exchangeable
  - $\longrightarrow$ hierarchical model

Partial or conditional exchangeability

- Conditional exchangeability
  - if y_i is connected to an additional information x_i, so that y_i are not exchangeable, but (y_i, x_i) exchangeable use joint model or conditional model $(y_i \mid x_i)$.

Partial or conditional exchangeability

- Conditional exchangeability
  - if y_i is connected to an additional information x_i, so that y_i are not exchangeable, but (y_i, x_i) exchangeable use joint model or conditional model $(y_i \mid x_i)$.
- Partial exchangeability
  - if the observations can be grouped (a priori), then use hierarchical model

Exchangeability

- The simplest form of the exchangeability (but not the only one) for the parameters θ conditional independence

$$p(x_1, \ldots, x_J \mid \theta) = \prod_{j=1}^{J} p(x_j \mid \theta)$$

Exchangeability - Counter example

- A six sided die with probabilities $\theta_1, \ldots, \theta_6$
  - without additional knowledge $\theta_1, \ldots, \theta_6$ exchangeable
  - due to the constraint $\sum_{j=1}^{6} \theta_j$, parameters are not independent and thus joint distribution can not be presented as iid

Exchangeability

- See more examples in the BDA3 notes - Exchangeability vs. independence