Practical 5: Generalized linear models

Benjamin Rosenbaum

Table of contents

We learn how to use some classic GLMs, and a distributional model. We focus on prior specifications and appropriate posterior predictive checks. Sometimes we have to apply ggplot tricks for good conditional effects plots.

```r
rm(list=ls())
library("brms")
library("ggplot2")
library("arm")
library("emmeans")
library("performance")
library("Data4Ecologists")
library("cowplot")
try(dev.off())
```

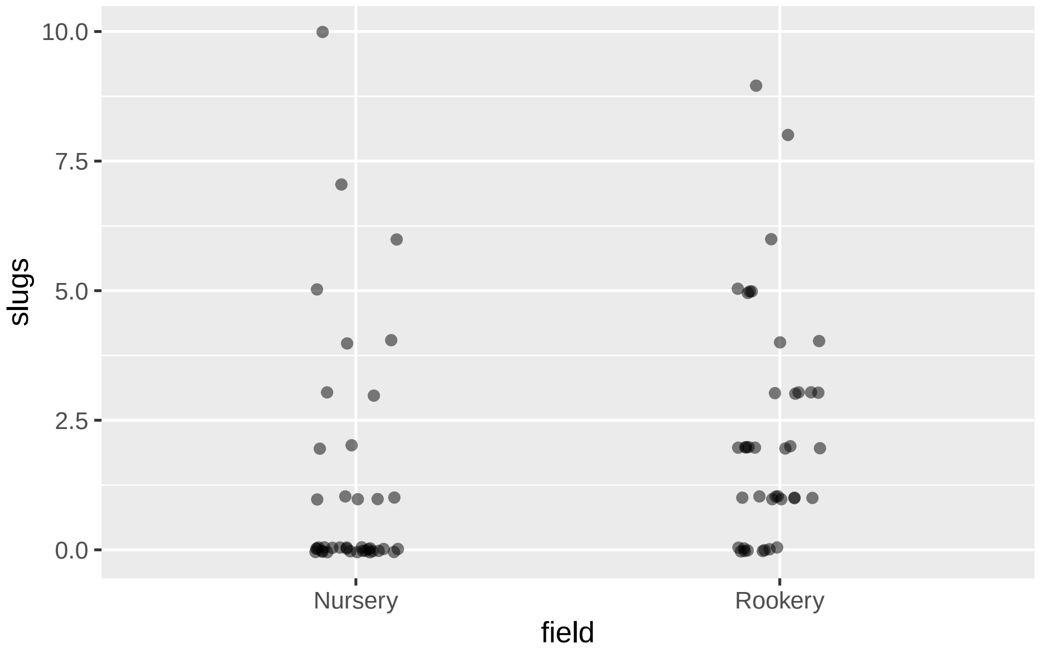
Poisson regression

We start with a simple comparison of 2 group means (frequentists would use a t-test), but the response are counts (number of slugs found per plot, 2 field types, from Dta4Ecologists package).

Deterministic part: $\mu = \mu(field)$ $->$ μ_1 or μ_2
Stochastic part: $slugs \sim \text{Poisson}(\mu)$

No log-link is used for this simple model, just "identity link".

```r
data(slugs)
ggplot(slugs, aes(field, slugs)) +
  geom_jitter(alpha=0.5, width=0.1, height=0.05)
```



It's always good to check brms default priors in a GLM:

```r
default_prior(slugs ~ field,
              family = poisson(link=identity),
              data = slugs)
```

```
               prior     class         coef group resp dpar nlpar lb ub       source
              (flat)         b                                                default
              (flat)         b fieldRookery                                (vectorized)
 student_t(3, 1, 2.5) Intercept                                                default
```

We provide a vague prior for effect (difference in means, dummy-coded) and ensure that the intercept is positive with lower boundary (`lb=0`)

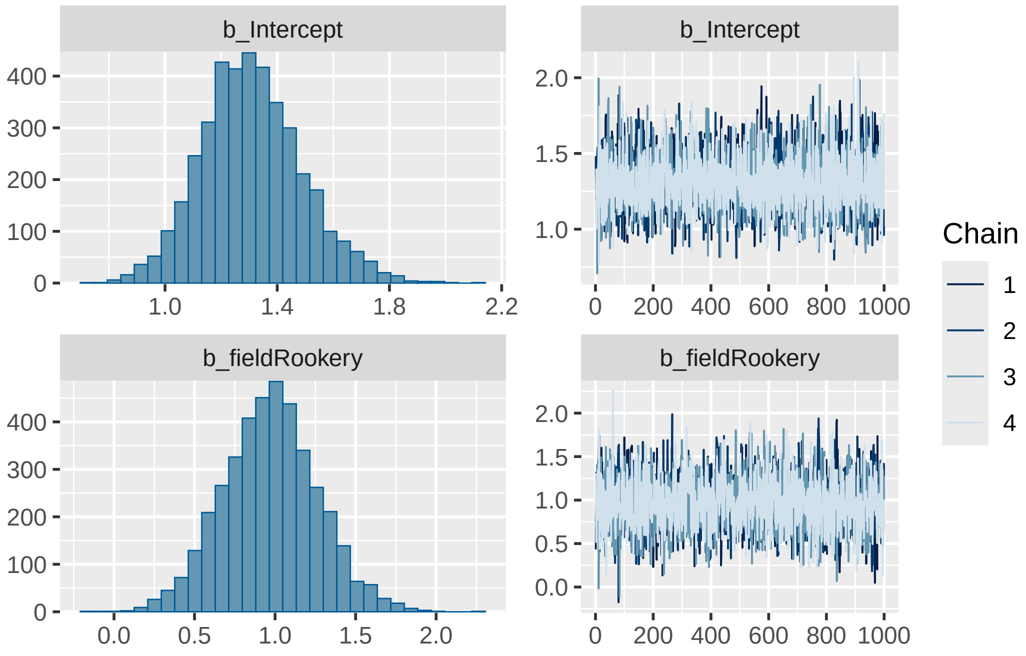
```
fit.slugs = brm(slugs ~ field,
                family = poisson(link=identity),
                prior =
                  prior(normal(0,2), class=b) +
                  prior(student_t(3, 1, 2.5), class=Intercept, lb=0),
                data = slugs)
```

Check convergence

```
summary(fit.slugs, prior=TRUE)
```

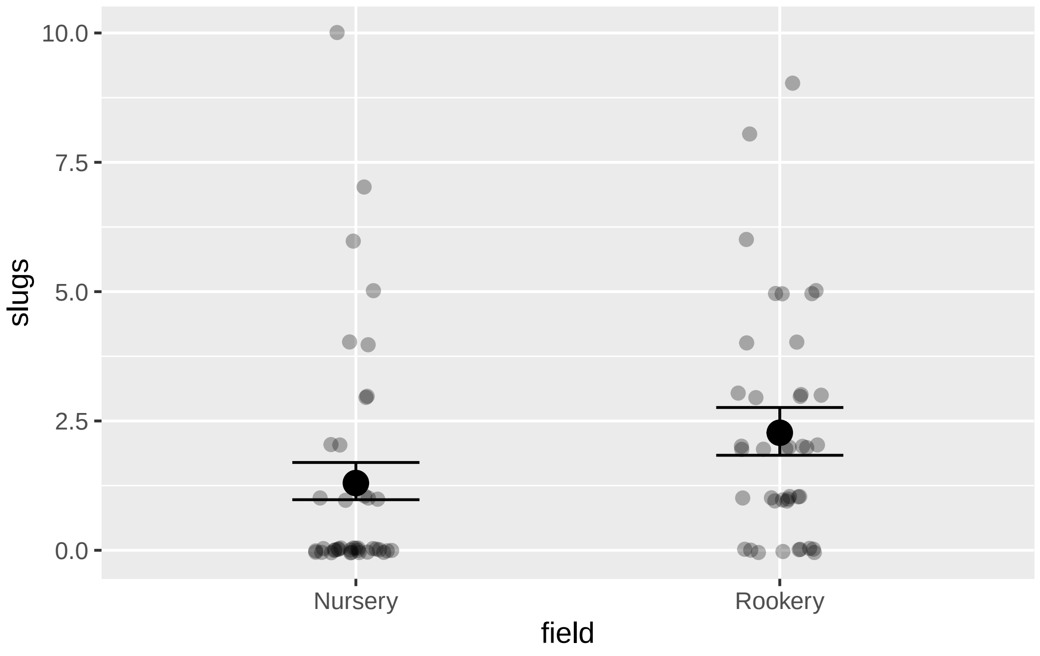
```
 Family: poisson
  Links: mu = identity
Formula: slugs ~ field
   Data: slugs (Number of observations: 80)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Priors:
b ~ normal(0, 2)
<lower=0> Intercept ~ student_t(3, 1, 2.5)

Regression Coefficients:
             Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept        1.31      0.18     0.98     1.70 1.00     3355     2765
fieldRookery     0.97      0.29     0.40     1.58 1.00     2634     2123

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
plot(fit.slugs)
```



Posterior predictions

```
plot(conditional_effects(fit.slugs),
     points=T,
     point_args=list(alpha=0.3, width=0.1, height=0.05)
)
```



In this simple example of comparing two group-means, the summary table already tells us that `field=Rookery` has on average +0.97 more slugs (95% CI [0.40,1.58]).

Importantly, we get quite a different effect size if we had just fitted an LM (effect=0.94, but 95% CI covering zero: [-0.05,1.90]). The Poisson model for count data is the correct model here, especially when observations are close or equal to zero!

```
fit.slugs.lm = brm(slugs ~ field,
                   prior =
                     prior(normal(0,2), class=b) +
                     prior(student_t(3, 1, 2.5), class=Intercept, lb=0),
                   data = slugs)
```

```
fixef(fit.slugs.lm) |> round(3)
```

```
             Estimate Est.Error   Q2.5 Q97.5
Intercept       1.303     0.352  0.612 1.990
fieldRookery    0.935     0.494 -0.054 1.904
```

Binomial regression

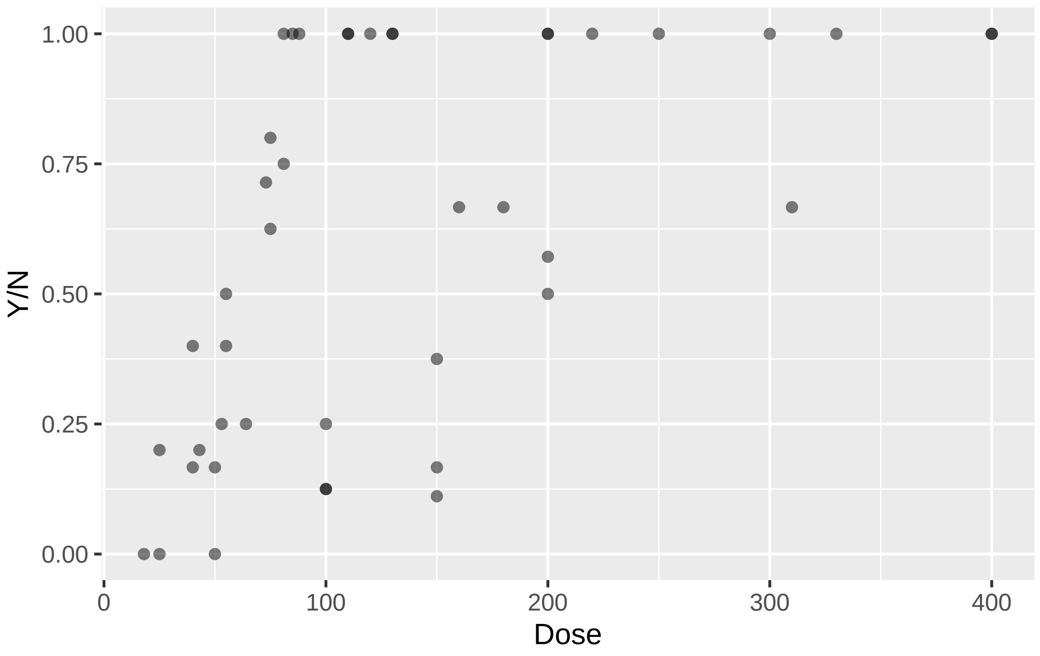
This experiment is about mice infected with parasites of the genus Cryptosporidium. Each replicate has a certain number of mice inoculated by some Dose of parasite oocysts. Then the number of infected mice is counted. (from Quian (2016) Environmental and Ecological Statistics with R)

Fit a dose-response model.

Question: What parasite dose is required to infect 75% of a mouse population?

Deterministic part: $\text{logit}(\mu) = a + b \cdot Dose$
Stochastic part: $Y_{infected} \sim \text{Binomial}(Y_{inoculated}, \mu)$

```
df = read.csv("https://raw.githubusercontent.com/songsqian/eesR/refs/heads/master/R/Data/cryp
              sep=" ")
df.sub = subset(df, Source=="Finch")
df.sub$Y = as.integer(df.sub$Y)
ggplot(df.sub, aes(Dose, Y/N)) + geom_point(alpha=0.5)
```



Again, we want to scale the predictor `Dose`. Here, we directly write `scale(Dose)` in the model formula. The advantage is that priors and parameters on z-scale, but plots / conditional_effects are on original scale.

```r
default_prior(Y | trials(N) ~ scale(Dose),
              family = binomial(link=logit),
              data = df.sub )
```

```
                prior     class      coef group resp dpar nlpar lb ub       source
               (flat)         b                                            default
               (flat)         b scaleDose                             (vectorized)
 student_t(3, 0, 2.5) Intercept                                            default
```

We add a vaguely informative prior on the (linear scale) slope. The effect is expected to be positive.

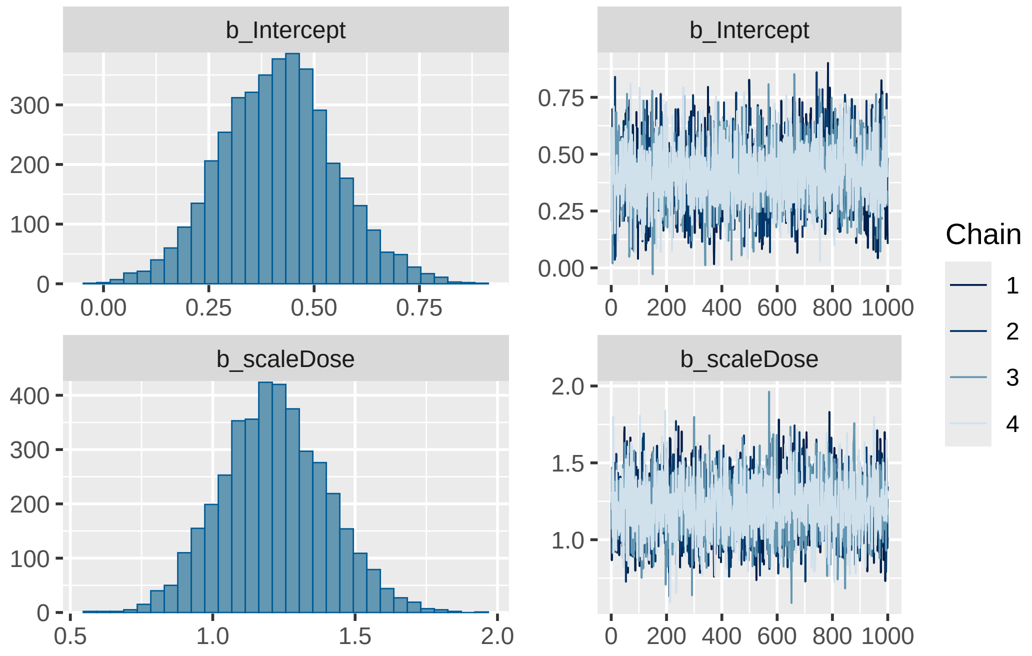
```r
fit.crypto = brm(Y | trials(N) ~ scale(Dose),
                 family = binomial(link=logit),
                 prior = prior(normal(1,1), class=b),
                 data = df.sub )
```

Check convergence

```r
summary(fit.crypto, prior=T)
```

```
 Family: binomial
  Links: mu = logit
Formula: Y | trials(N) ~ scale(Dose)
   Data: df.sub (Number of observations: 43)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Priors:
b ~ normal(1, 1)
Intercept ~ student_t(3, 0, 2.5)

Regression Coefficients:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept     0.41      0.14     0.15     0.69 1.00     3019     2729
scaleDose     1.22      0.19     0.86     1.59 1.00     2677     2494

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

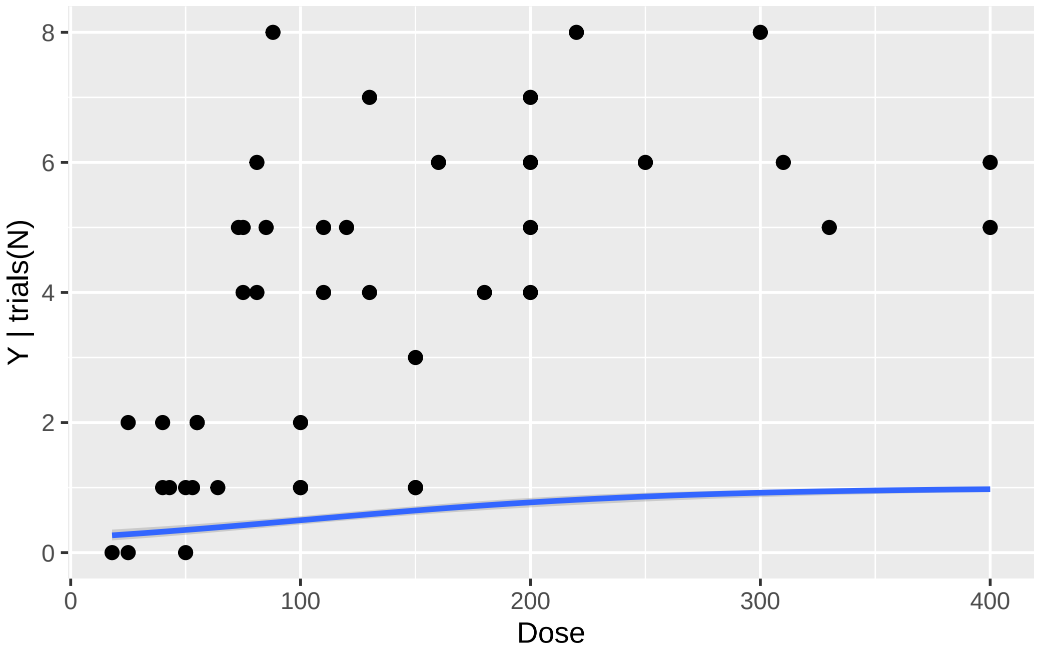
```r
plot(fit.crypto)
```



`conditional_effects()` plots predictions for N=1 trials only. `points=TRUE` just adds Y values, but we need Y/N values!

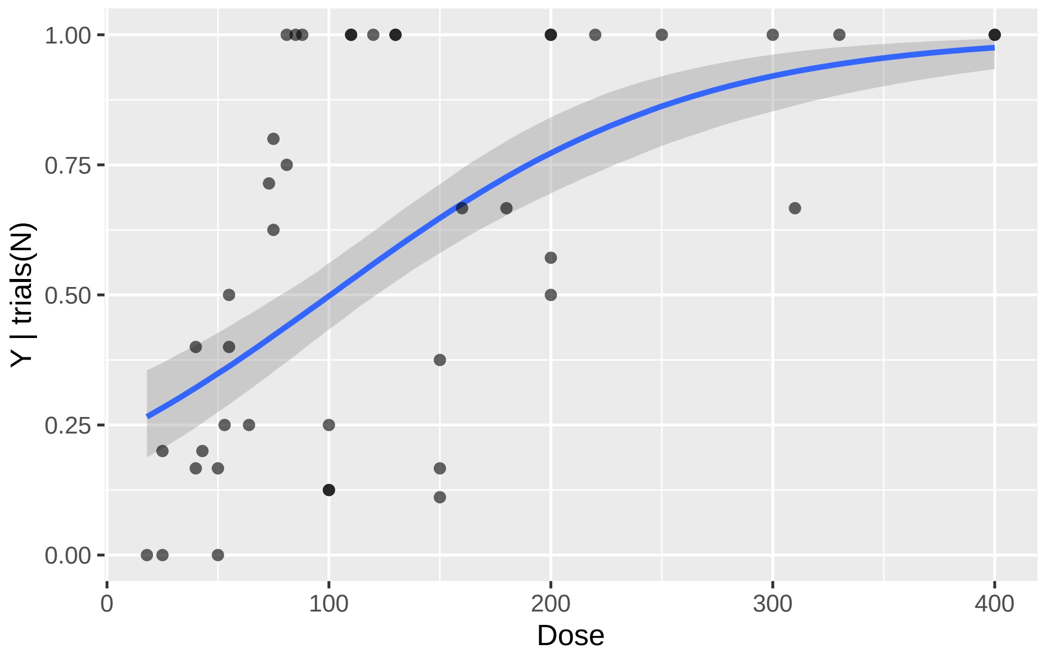
```
plot(conditional_effects(fit.crypto, effects="Dose"), points=TRUE)
```

```
Setting all 'trials' variables to 1 by default if not specified otherwise.
```



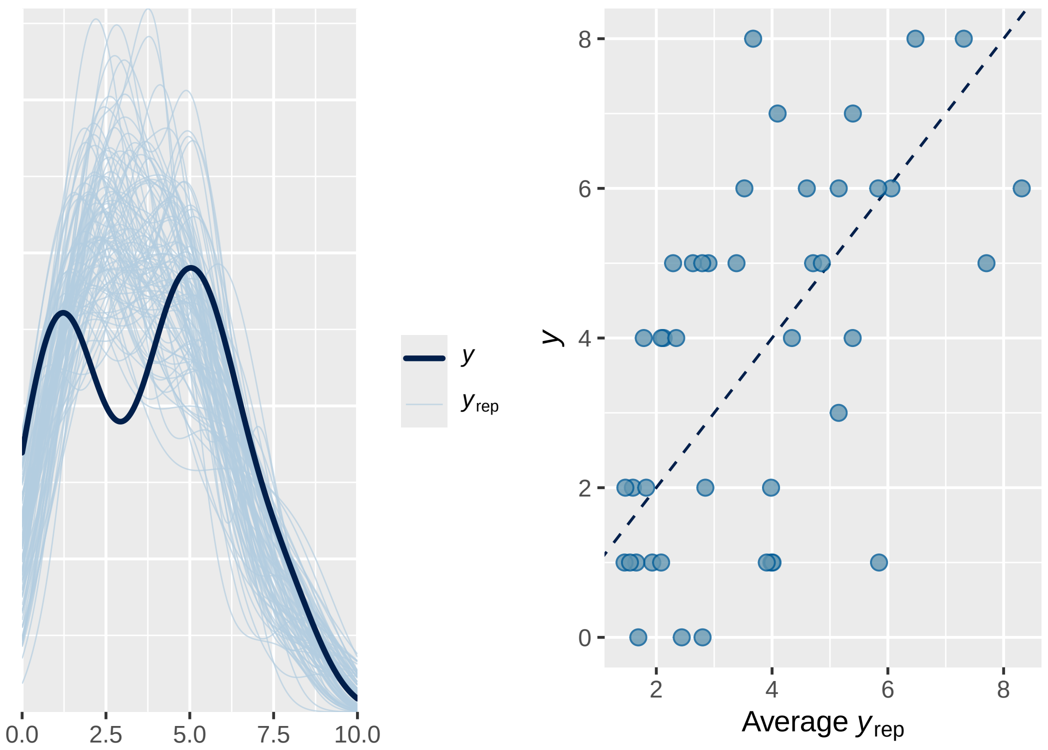
Some ggplot-tricks to the rescue!! Note that `conditional_effects()` now plots everything on the predictor's original scale, not on the z-scale.

```
p1 = plot(conditional_effects(fit.crypto, effects="Dose"),
          points=FALSE, plot=FALSE)
p1[[1]] + geom_point(data=df.sub,
                     aes(Dose, Y/N),
                     alpha=0.6, size=1.5,
                     inherit.aes=FALSE)
```

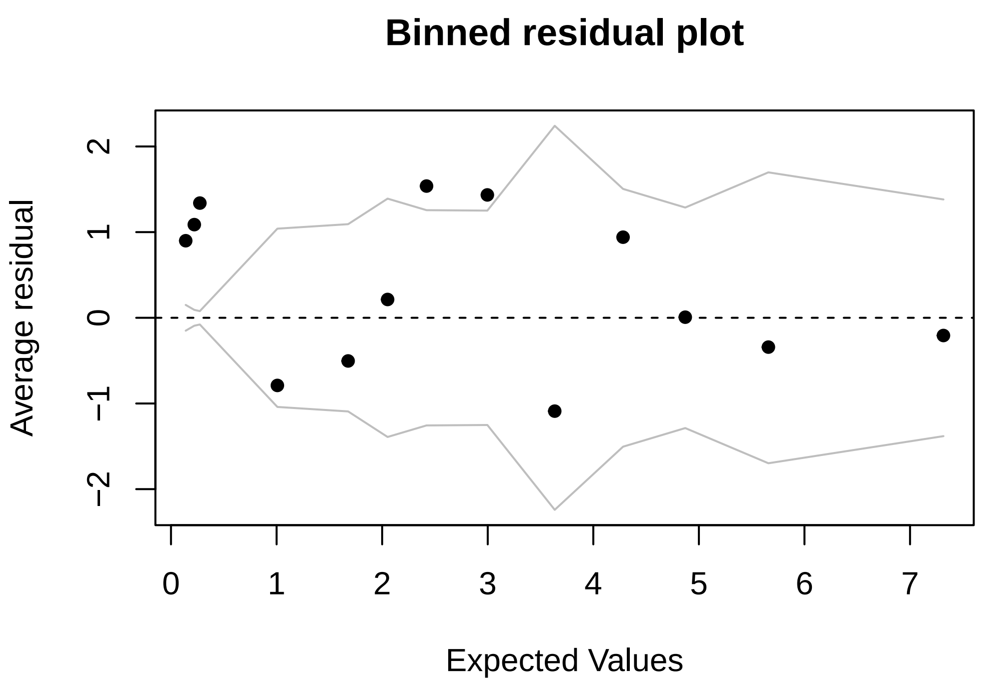


Posterior predictive checks are not very pretty, but we didn't measure any other predictors to improve it.

```
p1 = pp_check(fit.crypto, ndraws=100)
p2 = pp_check(fit.crypto, type="scatter_avg")
plot_grid(p1,p2)
```

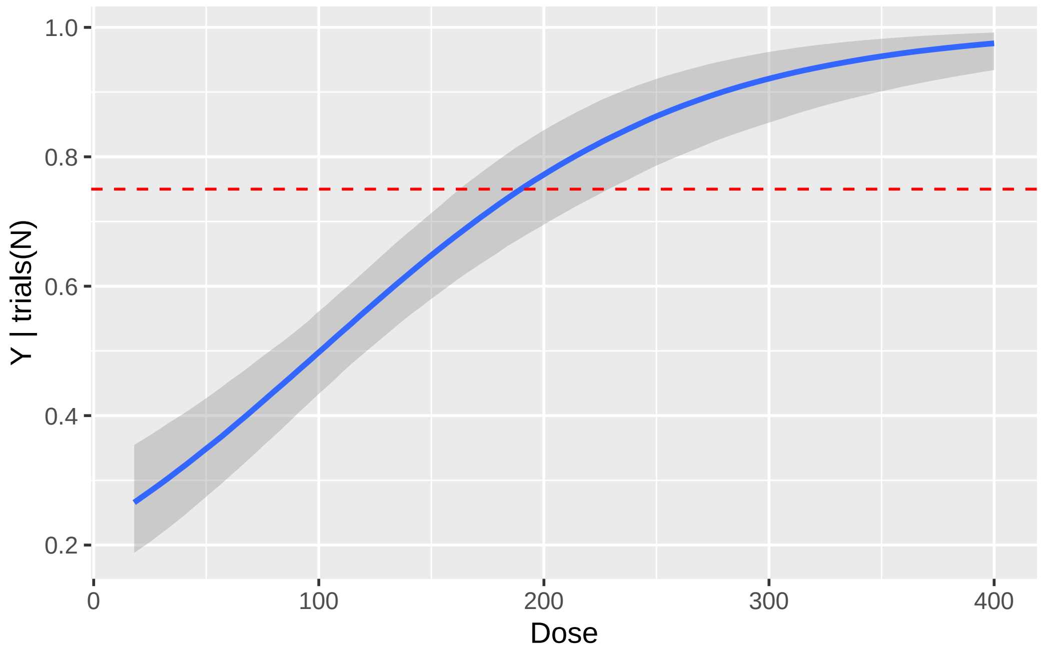


```
fitted = fitted(fit.crypto)
residuals = residuals(fit.crypto)
binnedplot(fitted, residuals)
```



What parasite dose is needed to get 75% of mice infected?

```r
p1 = plot(conditional_effects(fit.crypto, effects="Dose"),
          points=FALSE, plot=FALSE)
p1[[1]] + geom_hline(yintercept=0.75, linetype=2, col="red")
```

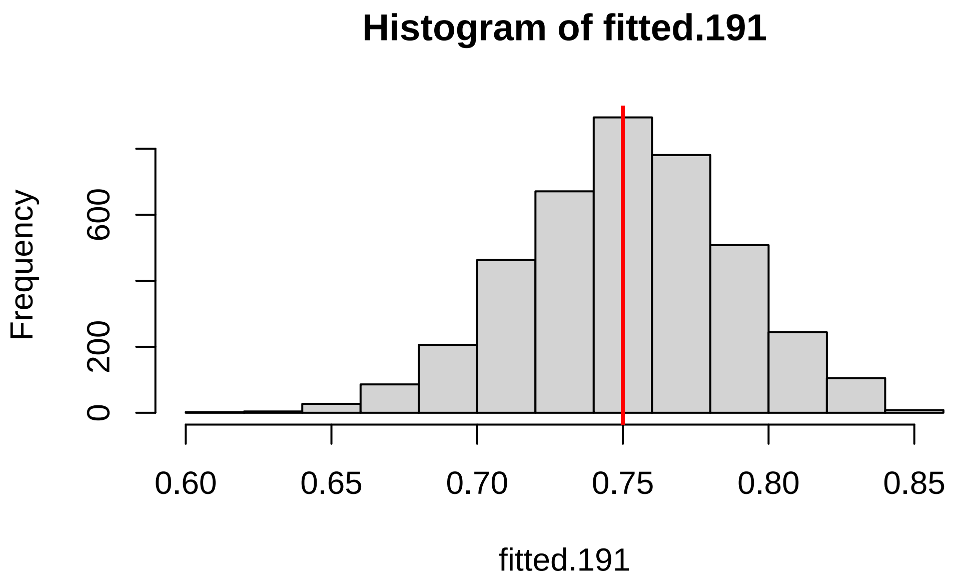


For Dose=191, the mean regression curve crosses the 75%-infected line, **on average** 75% are infected. But this means that we are **only 50% certain** that at least 75% are infected, $P(\mu > 0.75) \approx 0.5$.

```
fitted(fit.crypto,
       newdata = data.frame(Dose=191, N=1))
```

```
      Estimate  Est.Error      Q2.5     Q97.5
[1,] 0.7513291 0.03655965 0.6768719 0.8215464
```

```
fitted.191 = fitted(fit.crypto,
                    newdata = data.frame(Dose=191, N=1),
                    summary=FALSE)
hist(fitted.191)
abline(v=0.75, col="red", lwd=2)
```

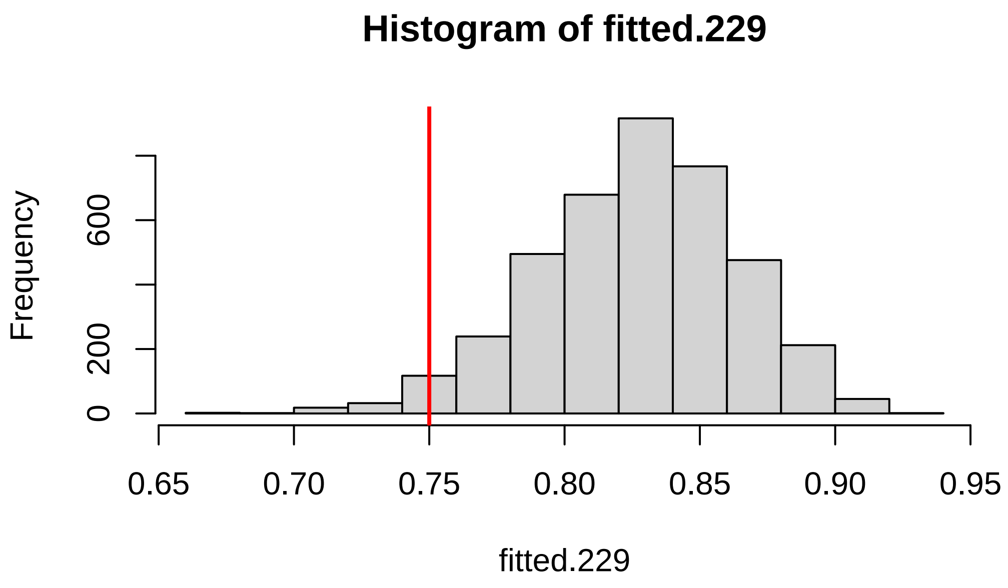


Rather, we'd have to supply a dose where the 75%-line crosses the fitted lower quantile (Q2.5), which happens around Dose=229. Then we are 97.5% certain that at least 75% of mice are infected, $P(\mu > 0.75) = 0.975$.

```
fitted(fit.crypto,
       newdata=data.frame(Dose=229, N=1),
       probs=0.025)
```

```
      Estimate  Est.Error      Q2.5
[1,] 0.8269148 0.03625958 0.7504071
```

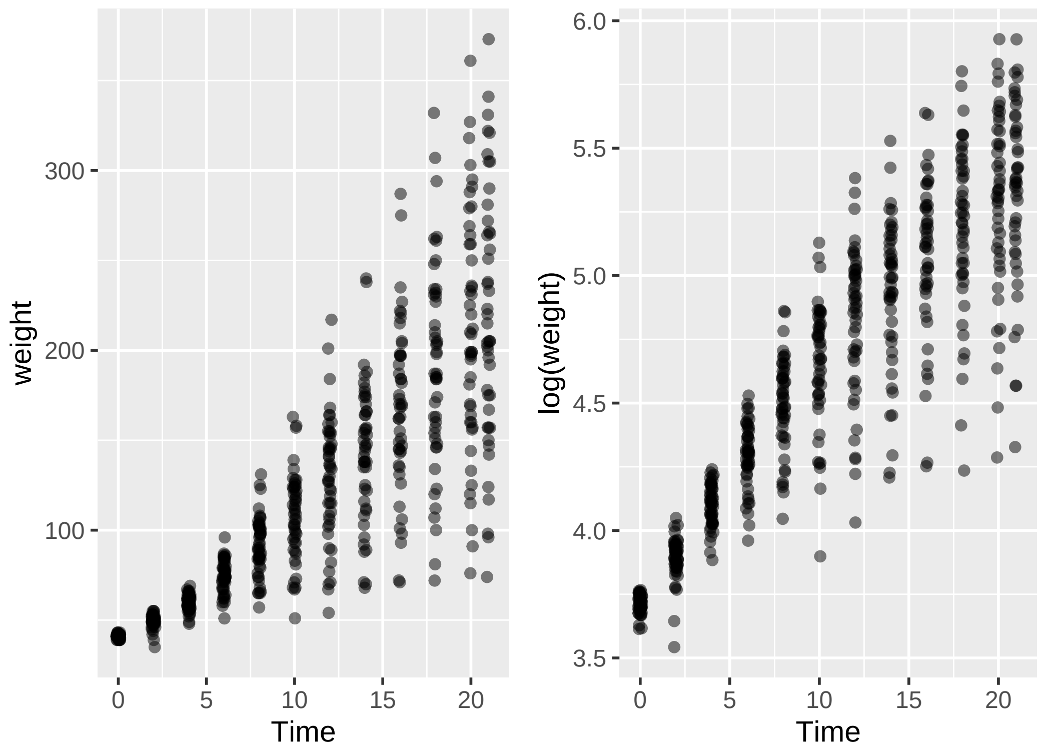
```
fitted.229 = fitted(fit.crypto,
                    newdata = data.frame(Dose=229, N=1),
                    summary=FALSE)
hist(fitted.229)
abline(v=0.75, col="red", lwd=2)
```



Overdispersion

We model the daily growth of young chicken (from datasets package). We use only the early exponential growth phase (looks linear on the logscale). Later, growth slows down and a the nonlinear von Bertalanffy growth function is more appropriate.

```
data(ChickWeight)
p1 = ggplot(ChickWeight, aes(Time, weight)) +
  geom_jitter(alpha=0.5, width=0.1, height=0.05)
p2 = ggplot(ChickWeight, aes(Time, log(weight))) +
  geom_jitter(alpha=0.5, width=0.1, height=0.05)
plot_grid(p1,p2)
```



```r
ChickWeight = subset(ChickWeight, Time<15)
```

Poisson

The response weight is measured in integers, so we could use Poisson here. Using a log-link, we fit a linear model.

Deterministic part: $\log(\mu) = a + b \cdot time$
Stochastic part: $weight \sim \text{Poisson}(\mu)$

```r
default_prior(weight ~ Time,
              family = poisson(link=log),
              data = ChickWeight)
```

```
                prior     class coef group resp dpar nlpar lb ub       source
               (flat)         b                                       default
               (flat)         b Time                             (vectorized)
 student_t(3, 4.3, 2.5) Intercept                                     default
```

We add a weak prior for the slope, but growth must be positive (chickens don't shrink).

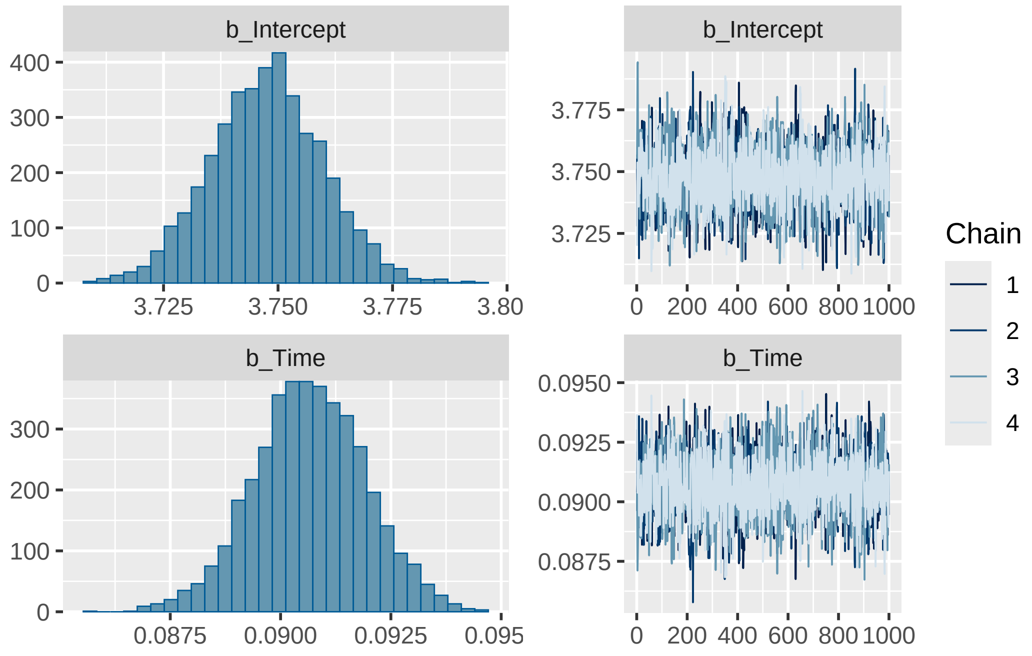
```
fit.growth.1 = brm(weight ~ Time,
                   family = poisson(link=log),
                   prior = prior(normal(0,1), class=b, lb=0),
                   data = ChickWeight)
```

Check convergence

```
summary(fit.growth.1)
```

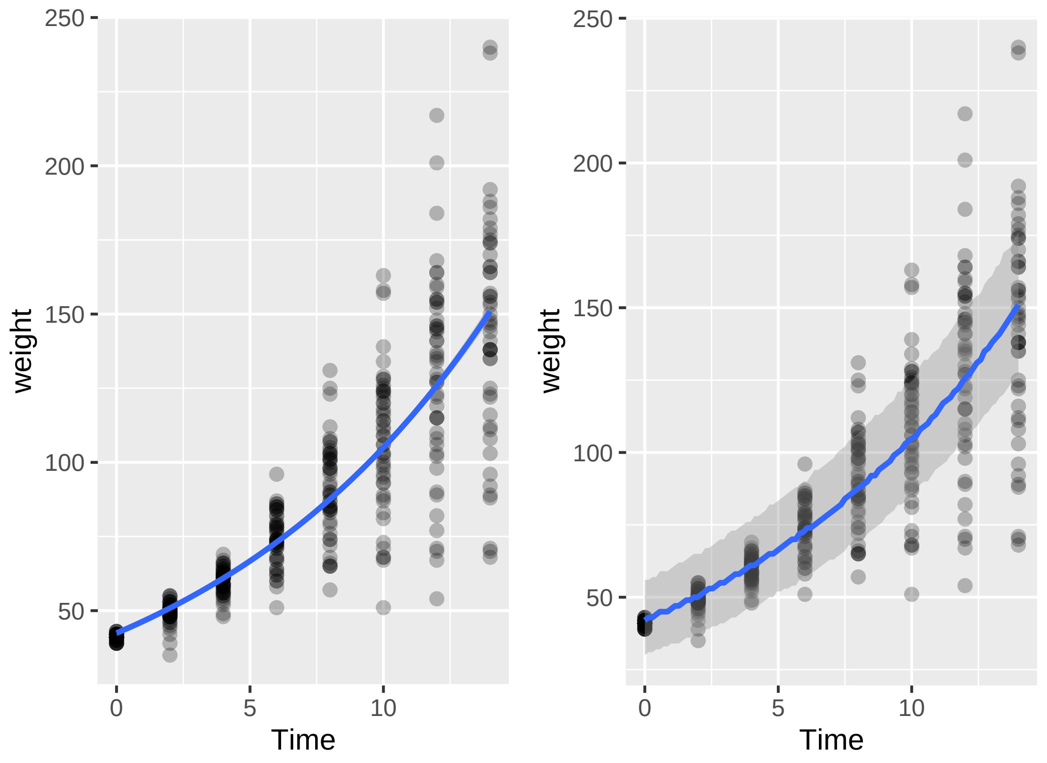
```
 Family: poisson
  Links: mu = log
Formula: weight ~ Time
   Data: ChickWeight (Number of observations: 393)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Regression Coefficients:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept     3.75      0.01     3.72     3.77 1.00     1994     1903
Time          0.09      0.00     0.09     0.09 1.00     2180     1779

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
plot(fit.growth.1)
```

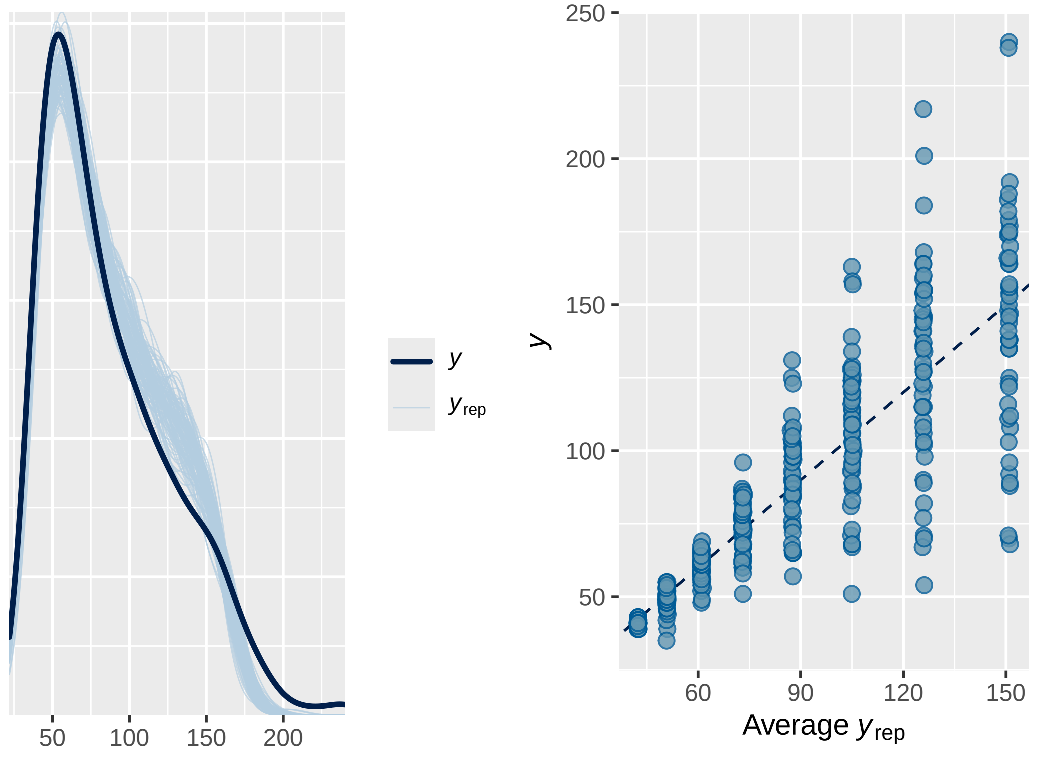


Posterior predictions for fitted values (deterministic part) look OK, but the predicted values don't correspond to that variance in the data (95% prediction intervals should contain ~95% of the data)

```
p1 = plot(conditional_effects(fit.growth.1),
          points=T,
          point_args=list(alpha=0.25),
          plot=F)
p2 = plot(conditional_effects(fit.growth.1, method="posterior_predict"),
          points=T,
          point_args=list(alpha=0.25),
          plot=F)
plot_grid(p1[[1]],p2[[1]])
```



```
p1 = pp_check(fit.growth.1, ndraws=100)
p2 = pp_check(fit.growth.1, type="scatter_avg")
plot_grid(p1,p2)
```



Negative Binomial

Deterministic part: $\log(\mu) = a + b \cdot time$
Stochastic part: $weight \sim \text{NegativeBinomial}(\mu, \phi)$

ϕ is an additional shape parameter modeling overdispersion, meaning the variance of the response does not scale linearly with the response itself (as it does in Poisson).

```
default_prior(weight ~ Time,
              family = negbinomial(link=log),
              data = ChickWeight)
```

```
                 prior     class coef group resp dpar nlpar lb ub       source
                (flat)         b                                       default
                (flat)         b Time                             (vectorized)
 student_t(3, 4.3, 2.5) Intercept                                      default
    inv_gamma(0.4, 0.3)     shape                             0         default
```

Again, we add a weak by strictly positive prior on the slope, brms provides default values also for ϕ

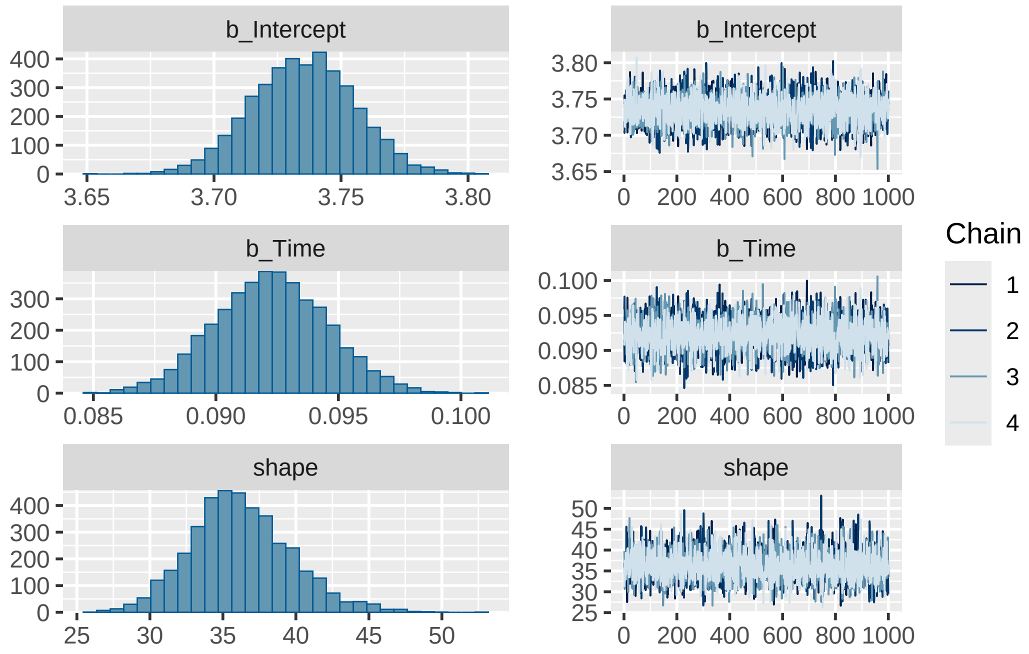
```r
fit.growth.2 = brm(weight ~ Time,
                   family = negbinomial(link=log),
                   prior = prior(normal(0,1), class=b, lb=0),
                   data = ChickWeight)
```

Check convergence

```r
summary(fit.growth.2)
```

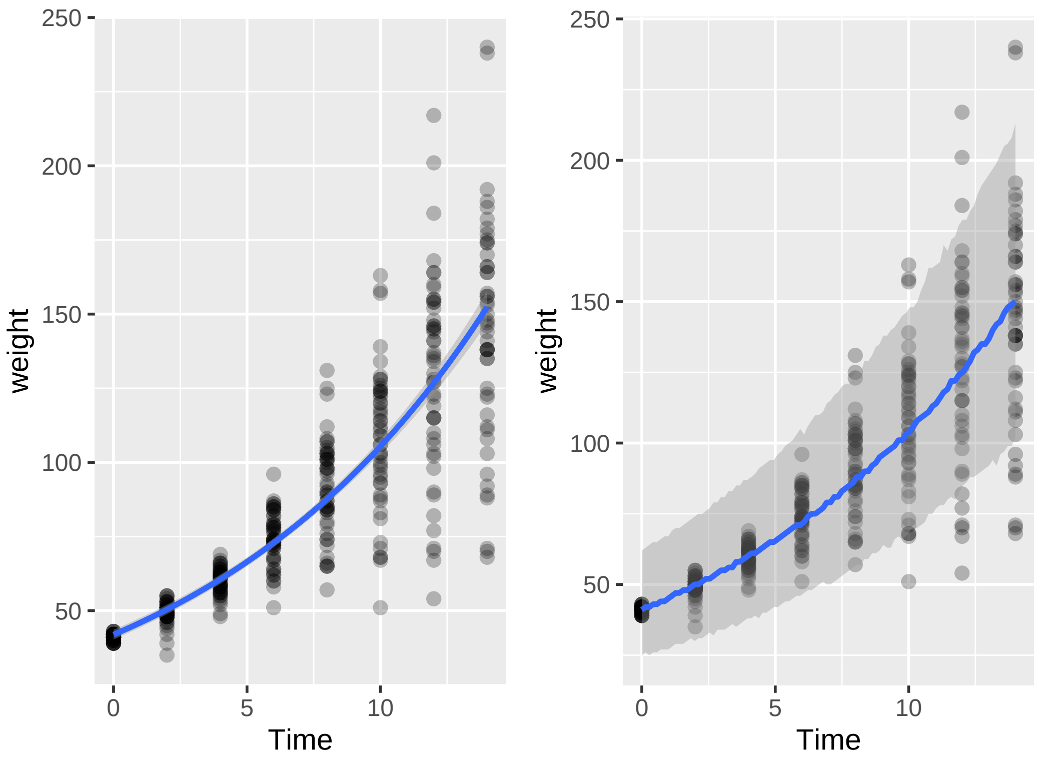
```
 Family: negbinomial
  Links: mu = log; shape = identity
Formula: weight ~ Time
   Data: ChickWeight (Number of observations: 393)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Regression Coefficients:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept     3.74      0.02     3.70     3.77 1.00     4080     2769
Time          0.09      0.00     0.09     0.10 1.00     4537     3190

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
shape    36.25      3.46    30.03    43.87 1.00     4312     2861

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

```r
plot(fit.growth.2)
```

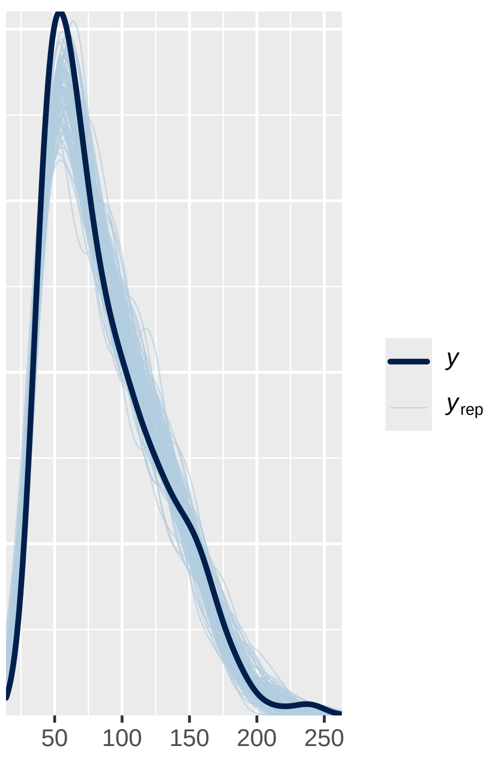


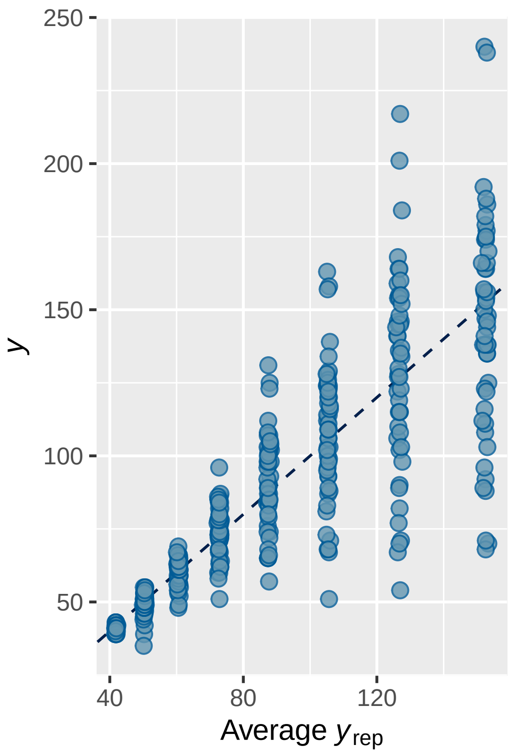
Looking at the same plot as before, the negative binomial model is more appropriate and replicates the variance of the observations much better.

```
p1 = plot(conditional_effects(fit.growth.2),
          points=T,
          point_args=list(alpha=0.25),
          plot=F)
p2 = plot(conditional_effects(fit.growth.2, method="posterior_predict"),
          points=T,
          point_args=list(alpha=0.25),
          plot=F)
plot_grid(p1[[1]],p2[[1]])
```



```
p1 = pp_check(fit.growth.2, ndraws=100)
p2 = pp_check(fit.growth.2, type="scatter_avg")
plot_grid(p1,p2)
```





Finally, model comparison reveals that the negative binomial model fits the data better.

```
LOO(fit.growth.1, fit.growth.2)
```

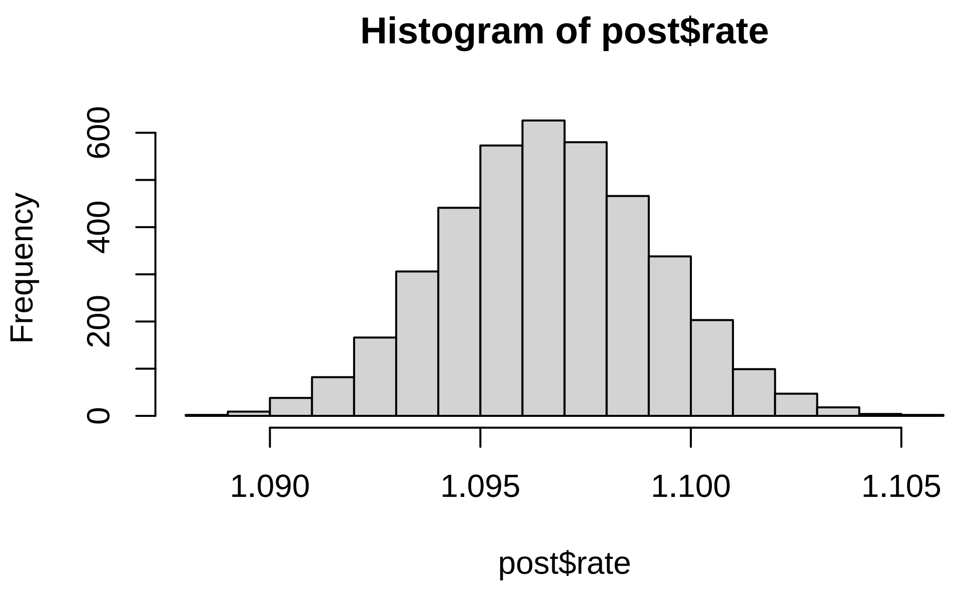
```
             elpd_diff se_diff
fit.growth.2    0.0       0.0
fit.growth.1 -327.9      66.2
```

Question: What is the average daily growth rate?

$\log(\mu) = a + b \cdot time$ $\quad$ −> $\quad$ $\mu = \exp(a) \cdot \exp(b)^{time}$

$\exp(b)$ is average daily growth rate, but we must use the whole posterior distribution! Don't just use mean $\bar{b}$ and compute $\exp(\bar{b})$. Jensen's inequality says you might get a biased value with nonlinear parameter transformations!

```
post = as_draws_df(fit.growth.2)
post$rate = exp(post$b_Time)
hist(post$rate)
```



```r
mean(post$rate)
```

```
[1] 1.096641
```

```r
quantile(post$rate, prob=c(0.05, 0.95))
```

```
      5%      95%
1.092519 1.100806
```

The daily growth rate is 1.096, this means an 9.6% increase every day.

Distributional model

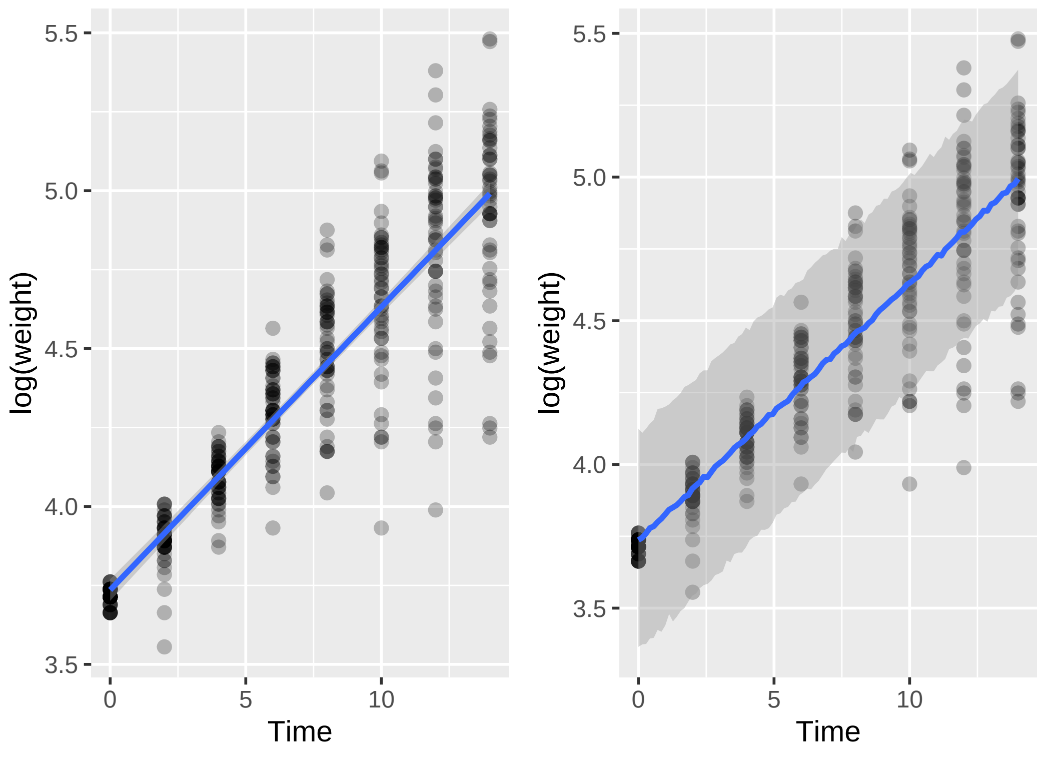
We use the same data, but treat the response as a continuous variable. We want to perform a simple linear regression on log(weight) first.

Deterministic part: $\mu = a + b \cdot time$
Stochastic part: $\log(weight) \sim \text{Normal}(\mu, \sigma)$

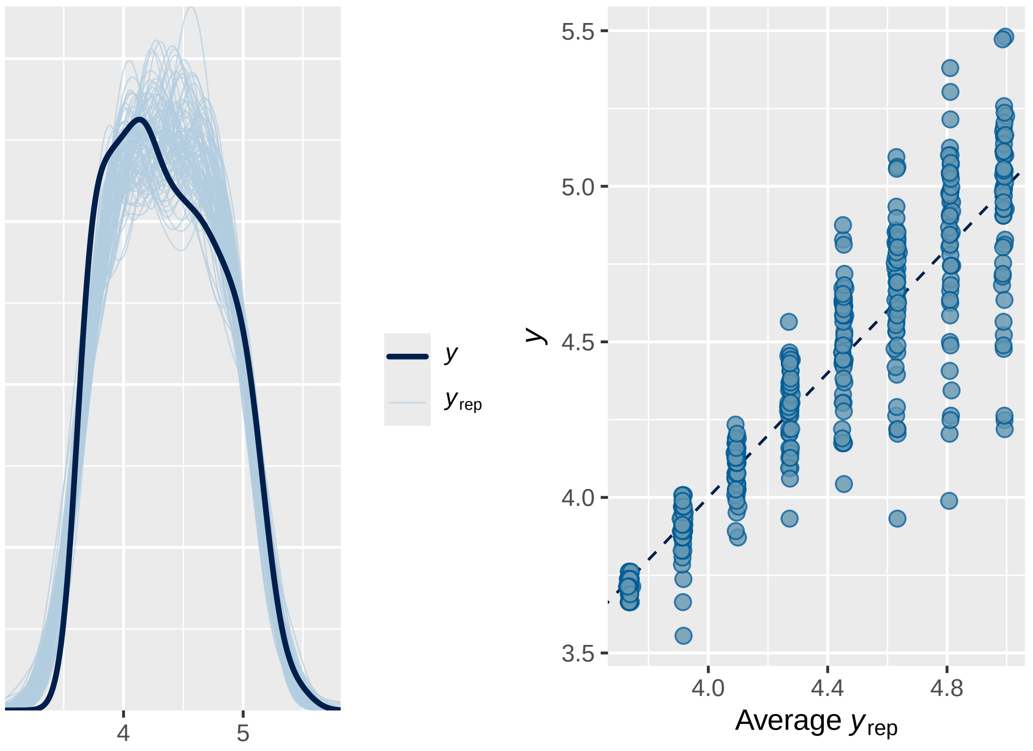
```r
fit.growth.3 = brm(log(weight) ~ Time,
                   prior = prior(normal(0,1), class=b),
                   data = ChickWeight)
```

Again, we see that prediction intervals don't represent the variance of data well.

```
p1 = plot(conditional_effects(fit.growth.3),
          points=T,
          point_args=list(alpha=0.25),
          plot=F)
p2 = plot(conditional_effects(fit.growth.3, method="posterior_predict"),
          points=T,
          point_args=list(alpha=0.25),
          plot=F)
plot_grid(p1[[1]],p2[[1]])
```

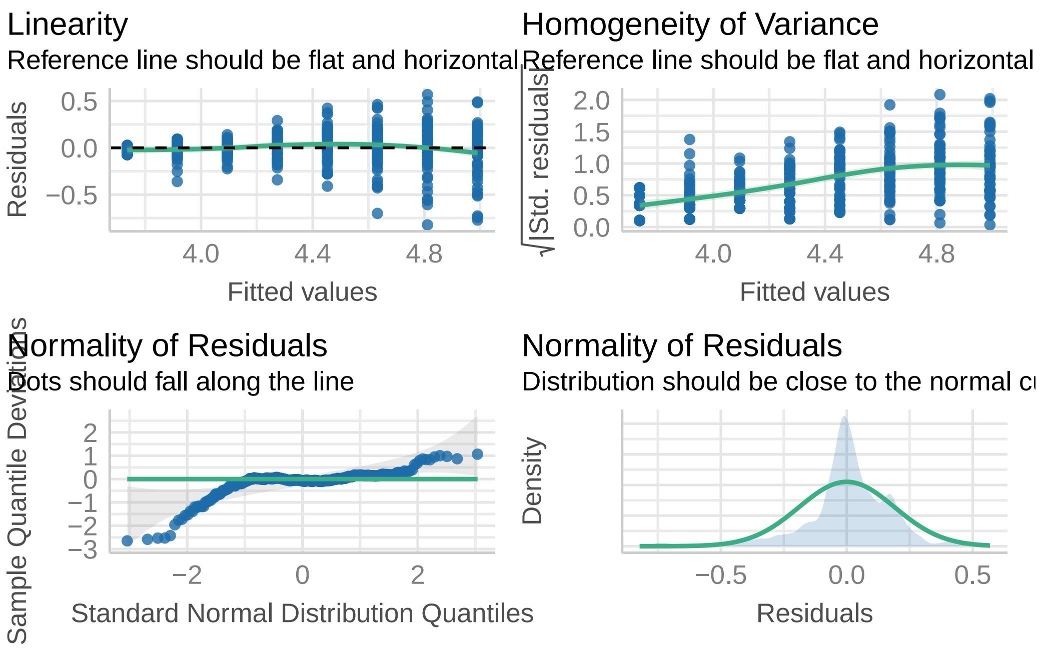


```
p1 = pp_check(fit.growth.3, ndraws=100)
p2 = pp_check(fit.growth.3, type="scatter_avg")
plot_grid(p1,p2)
```



Linear model assumptions are not satisfied

```r
check_model(fit.growth.3, check=c("linearity","homogeneity","qq","normality"))
```



We introduce a **distributional model**, where not only μ, but also σ is a function of the predictor *time*.

Deterministic part: $\mu = a + b \cdot time$

$\log(\sigma) = c + d \cdot time$ (log-link to keep $\sigma > 0$)

Stochastic part: $\log(weight) \sim \text{Normal}(\mu, \sigma)$

Both model parts for μ and σ are combined with `bf()` (short for brmsformula). For sigma, the log-link is provided automatically

```
default_prior(bf(log(weight) ~ Time,
                 sigma ~ Time),
              data = ChickWeight)
```

```
                prior     class coef group resp  dpar nlpar lb ub       source
               (flat)         b                                        default
               (flat)         b Time                              (vectorized)
 student_t(3, 4.3, 2.5) Intercept                                      default
               (flat)         b                 sigma                  default
               (flat)         b Time            sigma             (vectorized)
   student_t(3, 0, 2.5) Intercept                 sigma                  default
```

We provide weak priors for both effects (slope for μ and σ with time)

```
fit.growth.4 = brm(bf(log(weight) ~ Time,
                      sigma ~ Time),
                   prior =
                     prior(normal(0,1), class=b, coef=Time) +
                     prior(normal(0,1), class=b, dpar=sigma),
                   data = ChickWeight)
```

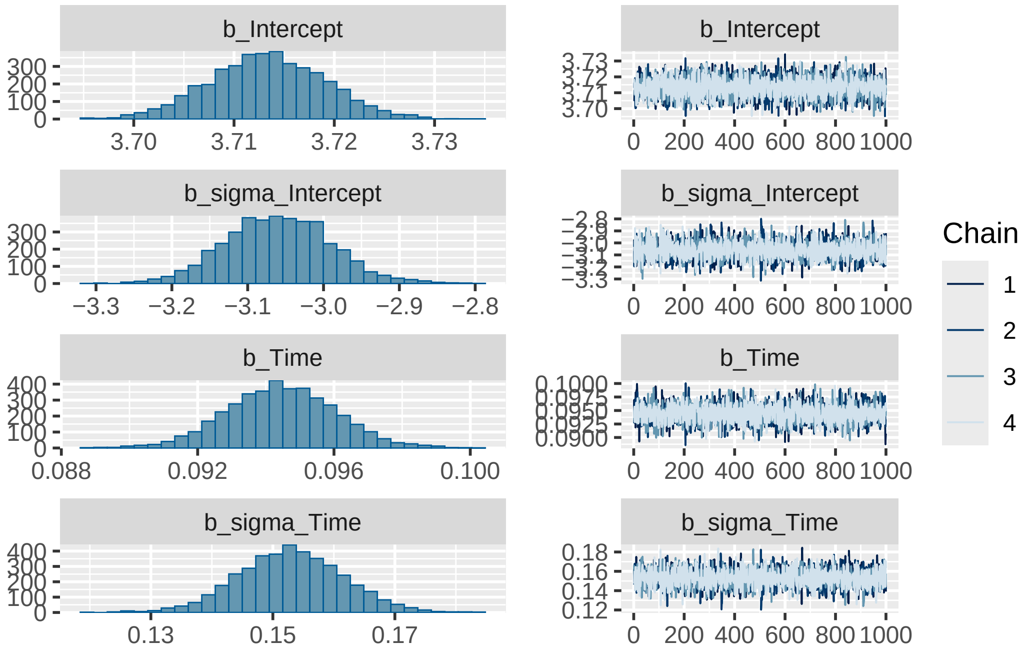
Check convergence

```
summary(fit.growth.4, prior=T)
```

```
 Family: gaussian
  Links: mu = identity; sigma = log
Formula: log(weight) ~ Time
         sigma ~ Time
   Data: ChickWeight (Number of observations: 393)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
```

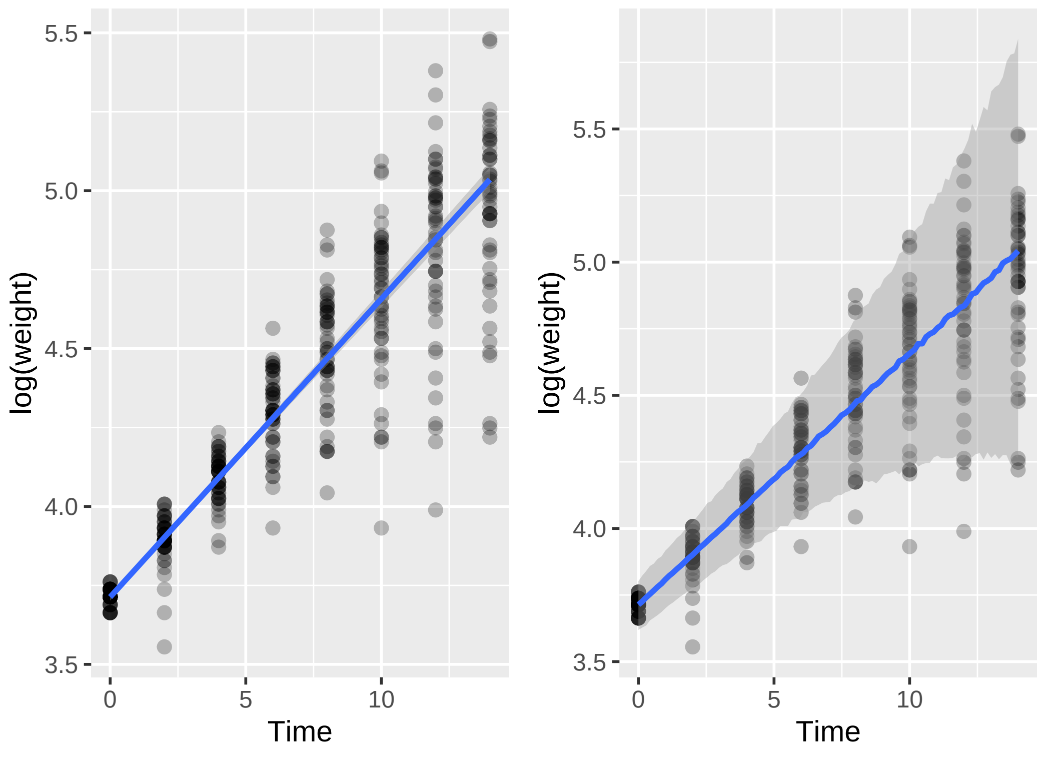
```
         total post-warmup draws = 4000

Priors:
b_Time ~ normal(0, 1)
b_sigma ~ normal(0, 1)
Intercept ~ student_t(3, 4.3, 2.5)
Intercept_sigma ~ student_t(3, 0, 2.5)

Regression Coefficients:
                Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept           3.71      0.01     3.70     3.72 1.00     4494     3323
sigma_Intercept    -3.06      0.07    -3.19    -2.92 1.00     2997     2371
Time                0.09      0.00     0.09     0.10 1.00     3013     2094
sigma_Time          0.15      0.01     0.14     0.17 1.00     3155     2368

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
plot(fit.growth.4)
```

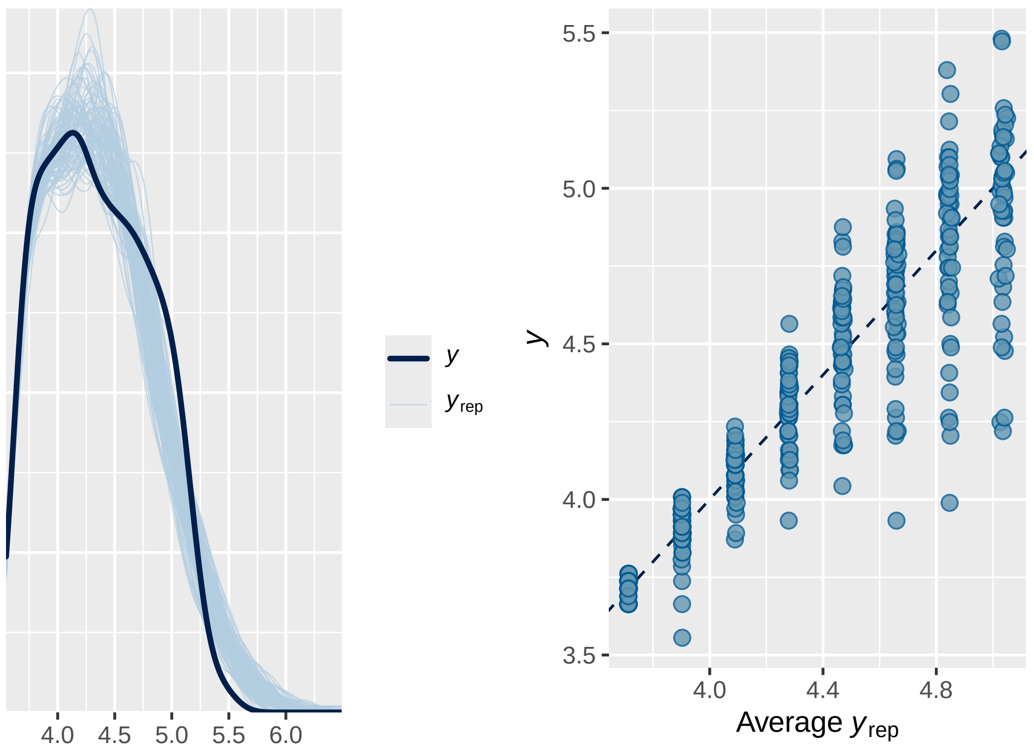


Posterior predictions are looking much better now. We don't use the `check_model()` here, because it's not a linear model anymore.

```
p1 = plot(conditional_effects(fit.growth.4),
          points=T,
          point_args=list(alpha=0.25),
          plot=F)
p2 = plot(conditional_effects(fit.growth.4, method="posterior_predict"),
          points=T,
          point_args=list(alpha=0.25),
          plot=F)
plot_grid(p1[[1]],p2[[1]])
```



```
p1 = pp_check(fit.growth.4, ndraws=100)
p2 = pp_check(fit.growth.4, type="scatter_avg")
plot_grid(p1,p2)
```



Model comparison reveals that the distributional model is preferred.

```
LOO(fit.growth.3, fit.growth.4)
```

```
             elpd_diff se_diff
fit.growth.4    0.0       0.0 
fit.growth.3 -131.1      22.3 
```

Attention: Do not use LOOIC / WAIC / AIC / BIC / DIC for comparing models for continuous responses (Normal, Lognormal, Gamma, Beta, ...) to models for discrete responses (Binomial, Poisson, Negative Binomial, ...)

Continuous distributions are based on probability density functions, while discrete distributions have probability mass functions, which you can't compare.

Logistic regression

From Qian, S. (2016) Environmental and Ecological Statistics with R

This field experiment is about seed predation by rodents (response $Predation = \{0, 1\}$). Seeds were placed in gauze bags in 4 topographic locations (predictor *topo*). Sites were visited in 6 sampling campaigns (predictor *time*). Also control for continuous predictor *seed.weight*.

Deterministic part: $\text{logit}(\mu) = ...$
Stochastic part: $Predation \sim \text{Bernoulli}(\mu)$

Start with `Predation~log(seed.weight)`

Test if there is a difference between locations (`+topo`)

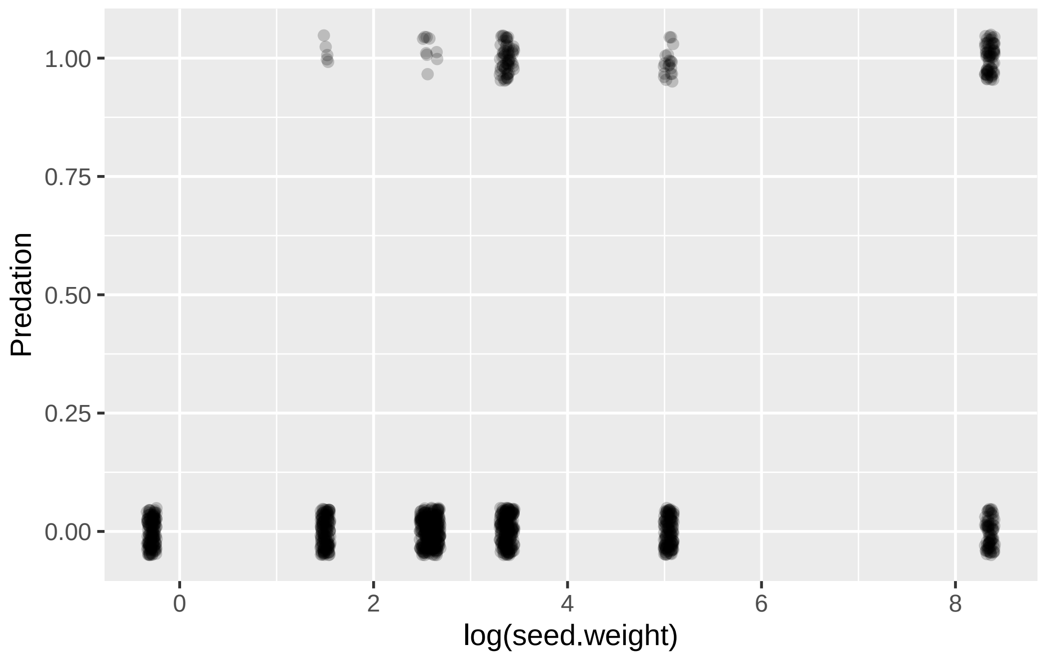
Also use time as factorial predictor (`+time`)

Question: Does predation rate reach a stable value in the last three time periods?

```
df = read.csv("https://raw.githubusercontent.com/songsqian/eesR/refs/heads/master/R/Data/seed
              sep=",")
df = subset(df, Predation %in% 0:1)
df$Predation = as.integer(df$Predation)
df$time = as.factor(df$time)
df$topo = as.factor(df$topo)
df$seed = scale(log(df$seed.weight))
head(df)
```

```
  species seed.weight time topo ground Predation        seed
1       8      4250.0    3    1      2         0  2.10606568
2       5        28.5    1    1      2         0  0.01142359
3       5        28.5    3    1      2         0  0.01142359
4       2         4.5    1    1      2         0 -0.76110864
5       2         4.5    2    1      2         0 -0.76110864
6       2         4.5    3    1      2         0 -0.76110864
```

```
ggplot(df, aes(log(seed.weight), Predation)) +
  geom_jitter(height=0.05, width=0.05, alpha=0.2)
```



Model 1

We start with a simple logistic regression, just with the continuous predictor `seed`. Note that priors for intercept and slope are on linear scale (η in the lecture), not on response scale (μ in the lecture).

```
df.seed.1 = brm(Predation ~ seed,
                family = bernoulli(link=logit),
                prior = prior(normal(1,1), class=b),
                data = df )
```

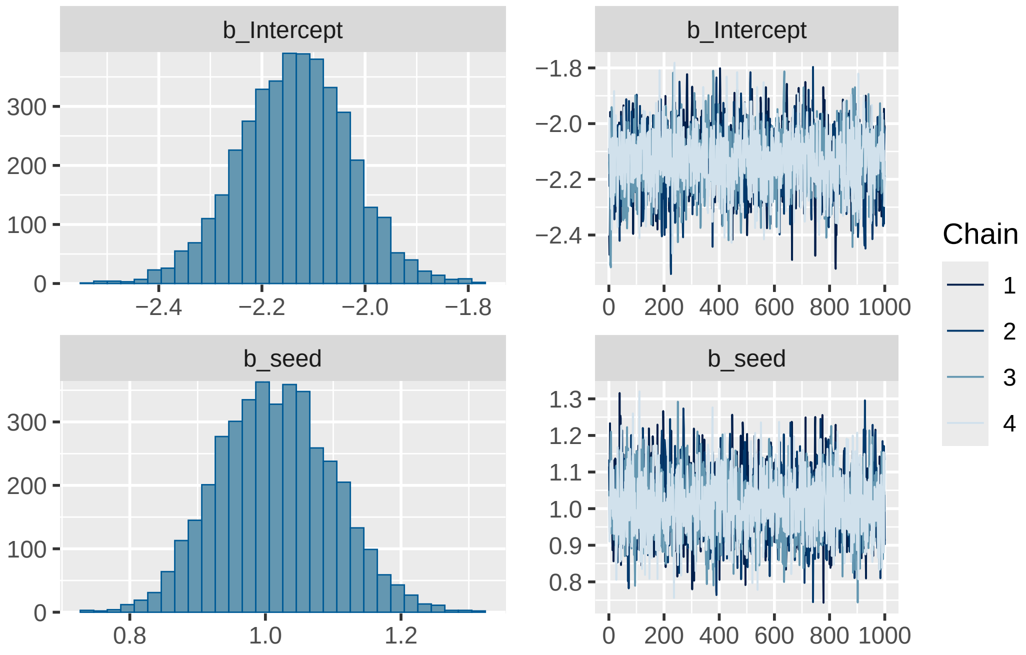
Check convergence

```
summary(df.seed.1, prior=T)
```

```
 Family: bernoulli
  Links: mu = logit
Formula: Predation ~ seed
   Data: df (Number of observations: 1142)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000
```

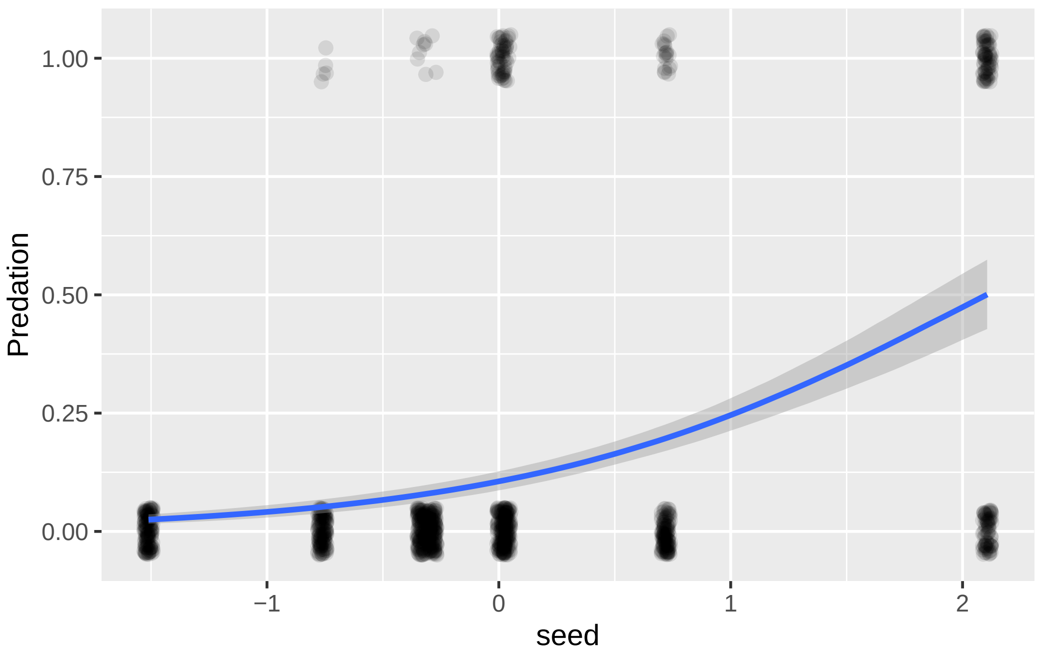
```
Priors:
b ~ normal(1, 1)
Intercept ~ student_t(3, 0, 2.5)

Regression Coefficients:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept    -2.14      0.11    -2.35    -1.93 1.00     1743     2201
seed          1.02      0.09     0.86     1.19 1.00     1876     2271

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

```r
plot(df.seed.1)
```



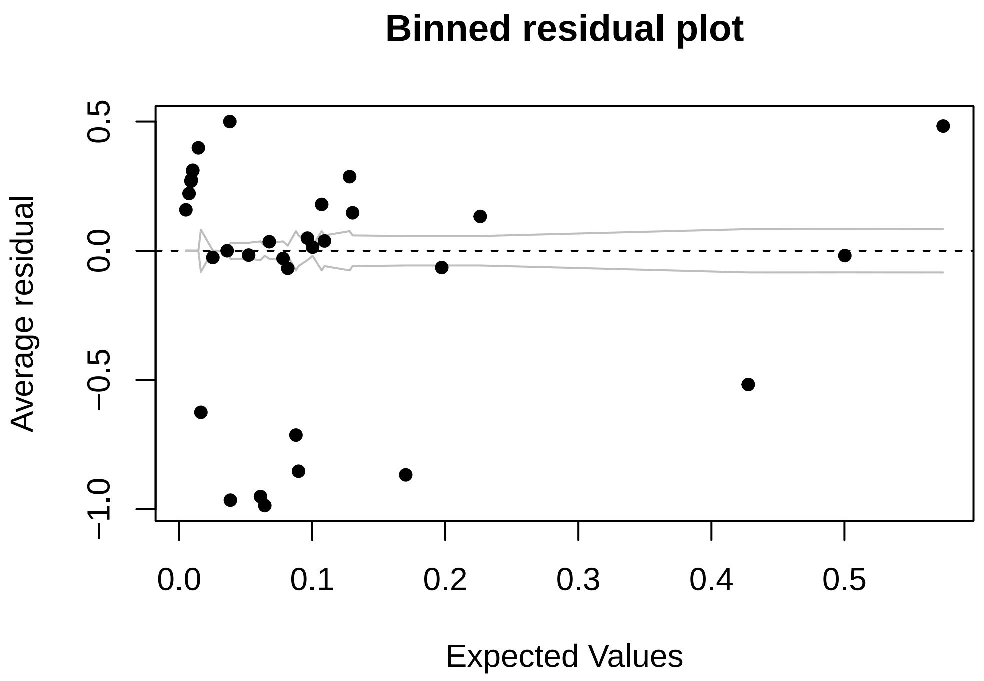
Posterior predictive checks

```r
plot(conditional_effects(df.seed.1),
     points = T,
     point_args = list(alpha=0.1, width=0.02, height=0.05)
)
```



We use the binned residuals plot, which is appropriate for 0/1 responses. It doesn't look great, but we will add more predictors

```
fitted = fitted(df.seed.1)
residuals = residuals(df.seed.1)
binnedplot(fitted, residuals)
```



Model 2

Now we add the categorical predictor `topo` to account for differences in location.

```
default_prior(Predation ~ seed + topo,
              family = bernoulli(link=logit),
              data = df )
```

```
                prior     class  coef group resp dpar nlpar lb ub        source
               (flat)         b                                         default
               (flat)         b  seed                              (vectorized)
               (flat)         b topo2                              (vectorized)
               (flat)         b topo3                              (vectorized)
               (flat)         b topo4                              (vectorized)
 student_t(3, 0, 2.5) Intercept                                         default
```

We assign priors for seed slope (continuous) and topo-differences (categorical)

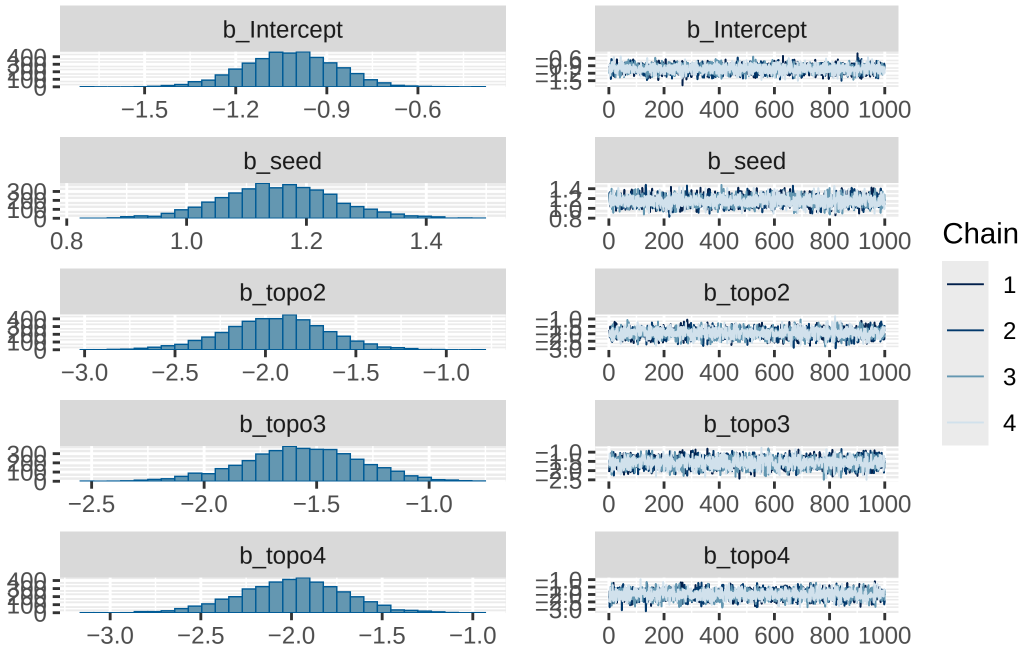
```
df.seed.2 = brm(Predation ~ seed + topo,
                family = bernoulli(link=logit),
                prior =
                  prior(normal(0,2), class=b) + # for all effects
                  prior(normal(1,1), class=b, coef=seed), # just for seed
                data = df )
```

Check convergence

```
summary(df.seed.2, prior=T)
```

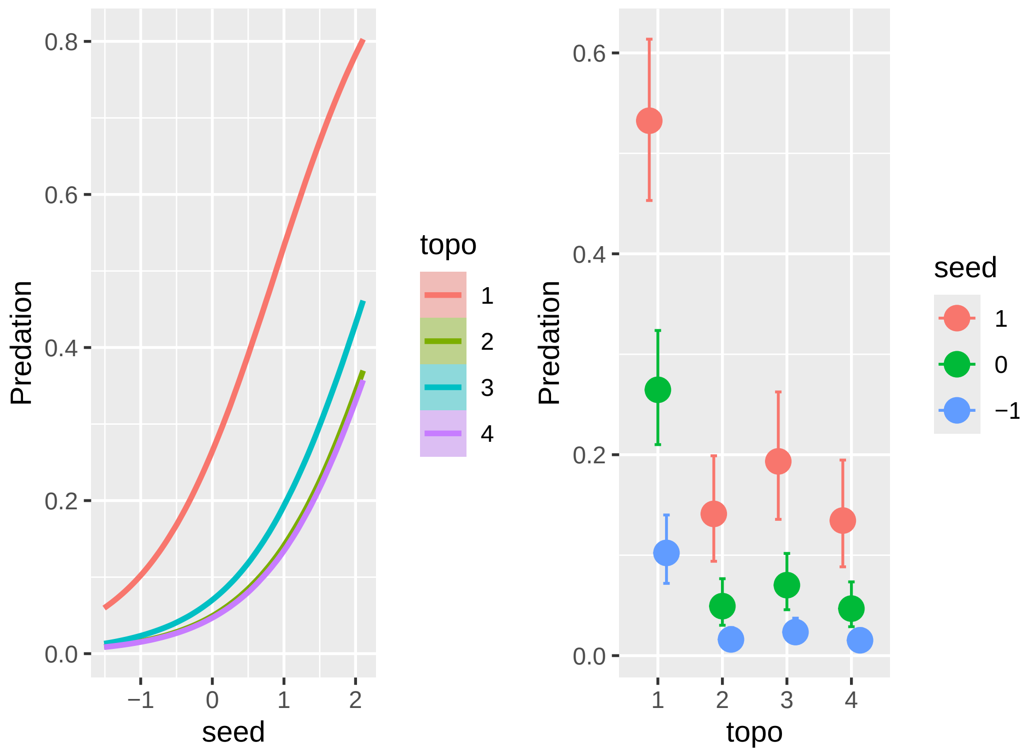
```
 Family: bernoulli
  Links: mu = logit
Formula: Predation ~ seed + topo
   Data: df (Number of observations: 1142)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Priors:
b ~ normal(0, 2)
b_seed ~ normal(1, 1)
Intercept ~ student_t(3, 0, 2.5)

Regression Coefficients:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept    -1.02      0.15    -1.32    -0.74 1.00     4301     3184
seed          1.15      0.10     0.97     1.35 1.00     2927     2778
topo2        -1.94      0.28    -2.52    -1.39 1.00     2927     2120
topo3        -1.57      0.26    -2.09    -1.07 1.00     3231     3086
topo4        -2.00      0.29    -2.58    -1.45 1.00     3392     2937

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
plot(df.seed.2)
```



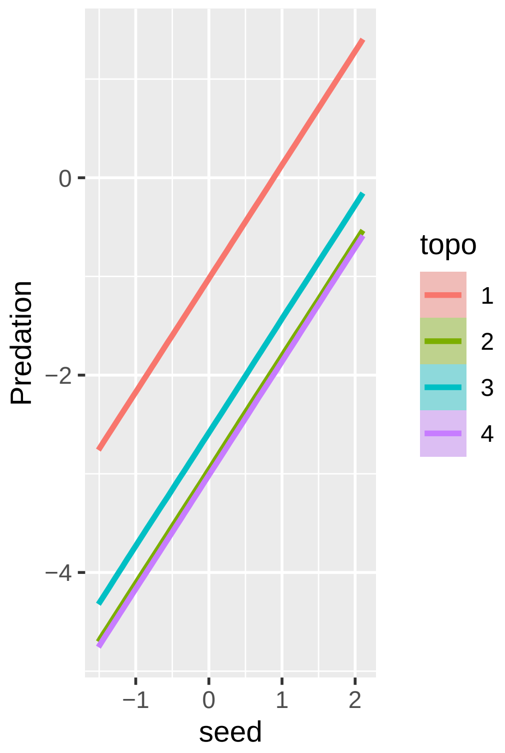
We see there is substantial variation between locations (`topo`)

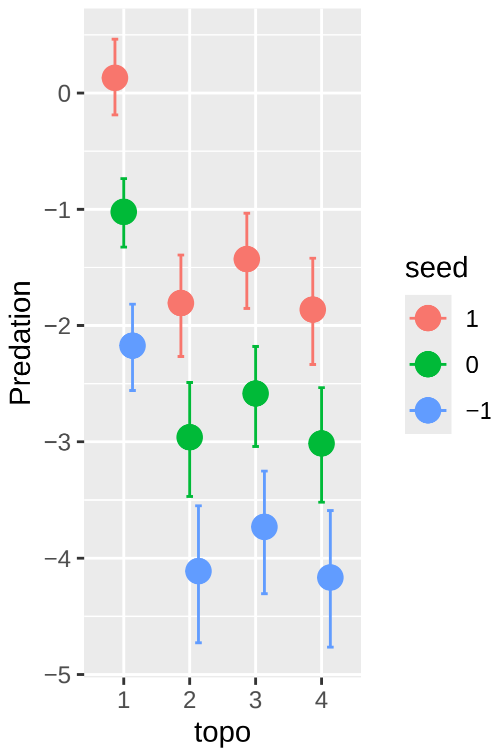
```
p1 = plot(conditional_effects(df.seed.2,
                              effect="seed:topo",
                              prob=0),
          plot=F)
p2 = plot(conditional_effects(df.seed.2,
                              effect="topo:seed"),
          plot=F)
plot_grid(p1[[1]],p2[[1]])
```



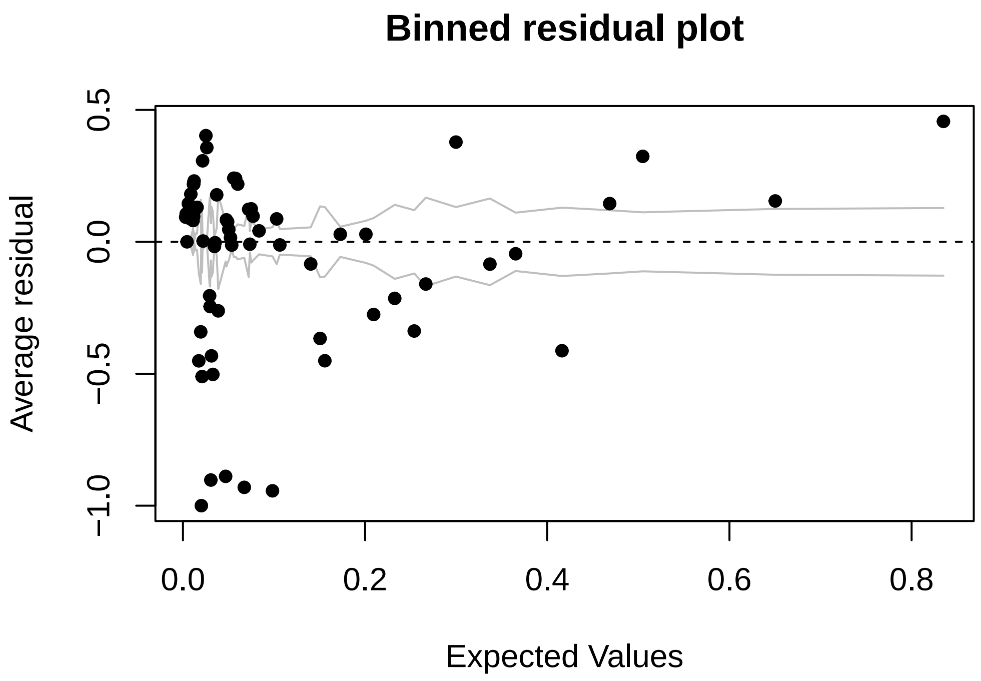
This `~seed+topo` (generalized) ANCOVA without interaction produces parallel fitted values (intercept varies with `topo`) on the linear scale (`method="posterior_linpred"`).

```
p1 = plot(conditional_effects(df.seed.2,
                              effect="seed:topo",
                              prob=0,
                              method="posterior_linpred"),
          plot=F)
p2 = plot(conditional_effects(df.seed.2,
                              effect="topo:seed",
                              method="posterior_linpred"),
          plot=F)
plot_grid(p1[[1]],p2[[1]])
```





```
fitted = fitted(df.seed.2)
residuals = residuals(df.seed.2)
binnedplot(fitted, residuals)
```



Residuals still not looking great, but model comparison suggests that including `topo` improves the model.

```
LOO(df.seed.1, df.seed.2)
```

```
          elpd_diff se_diff
df.seed.2   0.0       0.0
df.seed.1 -40.3       9.3
```

Model 3

Now we add sampling campaign (`time`) as a factorial predictor.

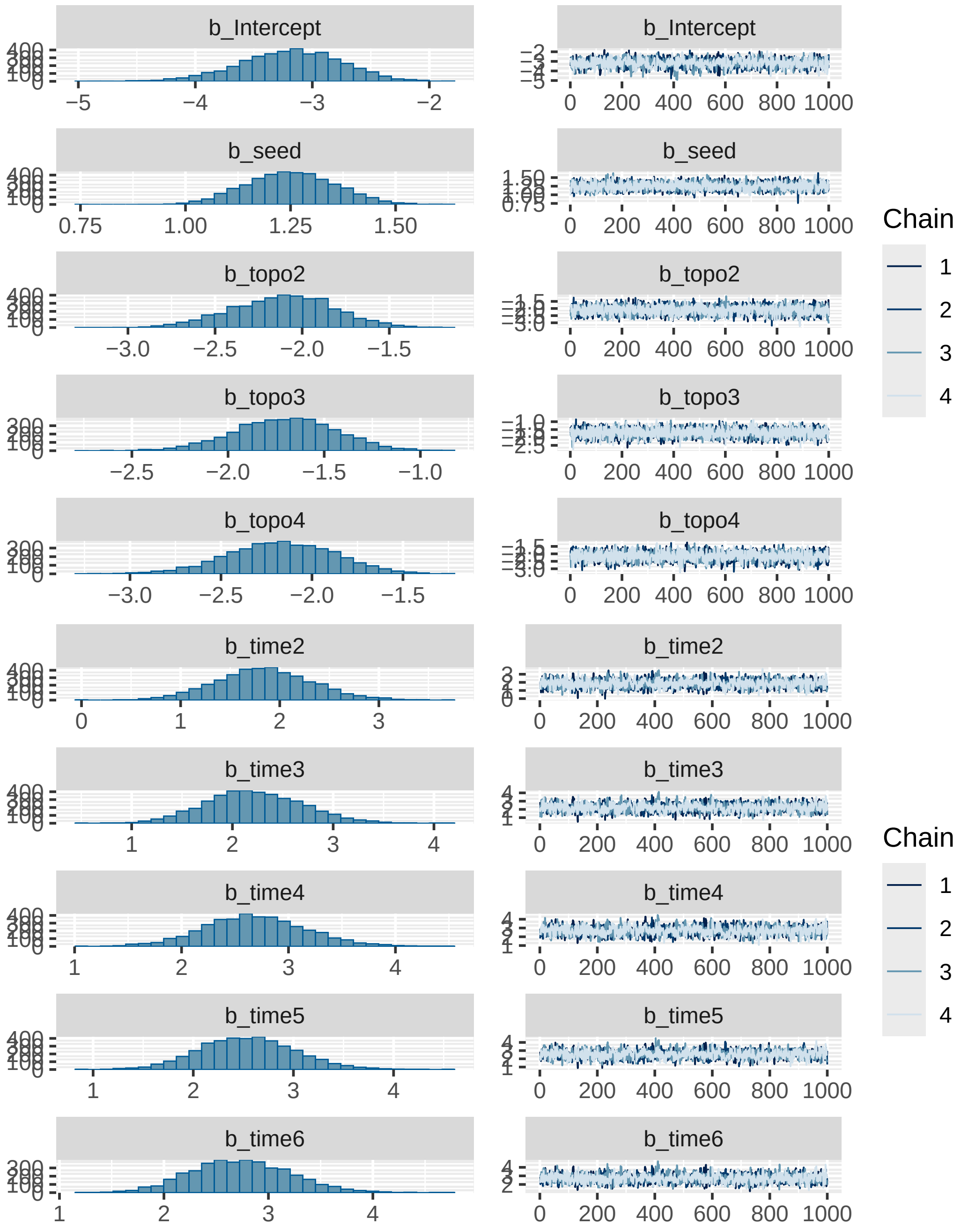
```
df.seed.3 = brm(Predation ~ seed + topo + time,
                family = bernoulli(link=logit),
                prior =
                  prior(normal(0,2), class=b) + # for all effects
                  prior(normal(1,1), class=b, coef=seed), # just for seed
                data = df )
```

Check convergence

```r
summary(df.seed.3, prior=T)
```

```
 Family: bernoulli
  Links: mu = logit
Formula: Predation ~ seed + topo + time
   Data: df (Number of observations: 1142)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Priors:
b ~ normal(0, 2)
b_seed ~ normal(1, 1)
Intercept ~ student_t(3, 0, 2.5)

Regression Coefficients:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept    -3.19      0.43    -4.07    -2.39 1.00     1418     1671
seed          1.25      0.10     1.05     1.45 1.00     2017     2646
topo2        -2.11      0.29    -2.69    -1.56 1.00     2774     2972
topo3        -1.70      0.26    -2.23    -1.20 1.00     3010     3333
topo4        -2.17      0.29    -2.75    -1.61 1.00     2671     2714
time2         1.85      0.49     0.92     2.84 1.00     1642     1785
time3         2.22      0.48     1.32     3.19 1.00     1594     1777
time4         2.69      0.48     1.78     3.66 1.00     1497     1952
time5         2.54      0.48     1.65     3.49 1.00     1550     1941
time6         2.74      0.48     1.85     3.70 1.00     1541     2034

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

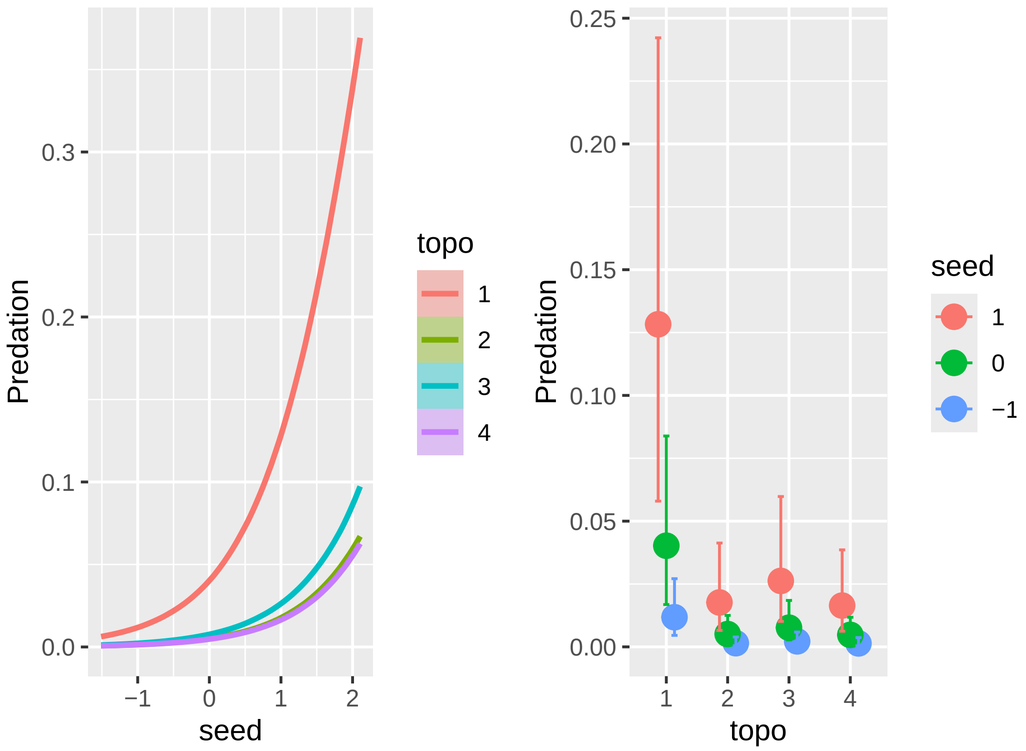
```r
plot(df.seed.3)
```



With 3 predictors, plotting conditional effects is getting more complicated, but the function can handle it.

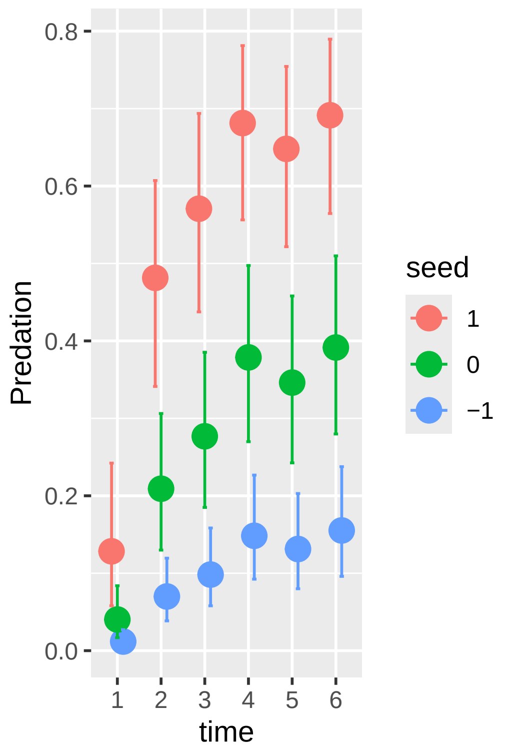
The same plot as in Model 2, but when we don't specify the 3rd predictor, fitted values are shown at its reference level (`time=1`)

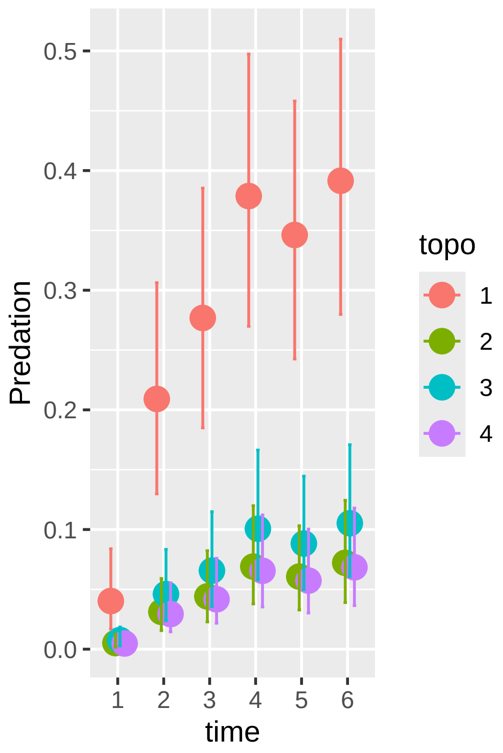
```
p1 = plot(conditional_effects(df.seed.3,
                              effect="seed:topo",  # time at reference level
                              prob=0),
          plot=F)
p2 = plot(conditional_effects(df.seed.3,
                              effect="topo:seed"), # time at reference level
          plot=F)
plot_grid(p1[[1]],p2[[1]])
```



We're actually interested in prediction for different `time`-levels. Plotting `time:seed` and `time:topo`, the omitted 3rd predictor is held at its reference level.

```
p1 = plot(conditional_effects(df.seed.3,
                              effect="time:seed"), # topo at reference level
          plot=F)

p2 = plot(conditional_effects(df.seed.3,
                              effect="time:topo"), # seed at mean=0
          plot=F)
plot_grid(p1[[1]],p2[[1]])
```

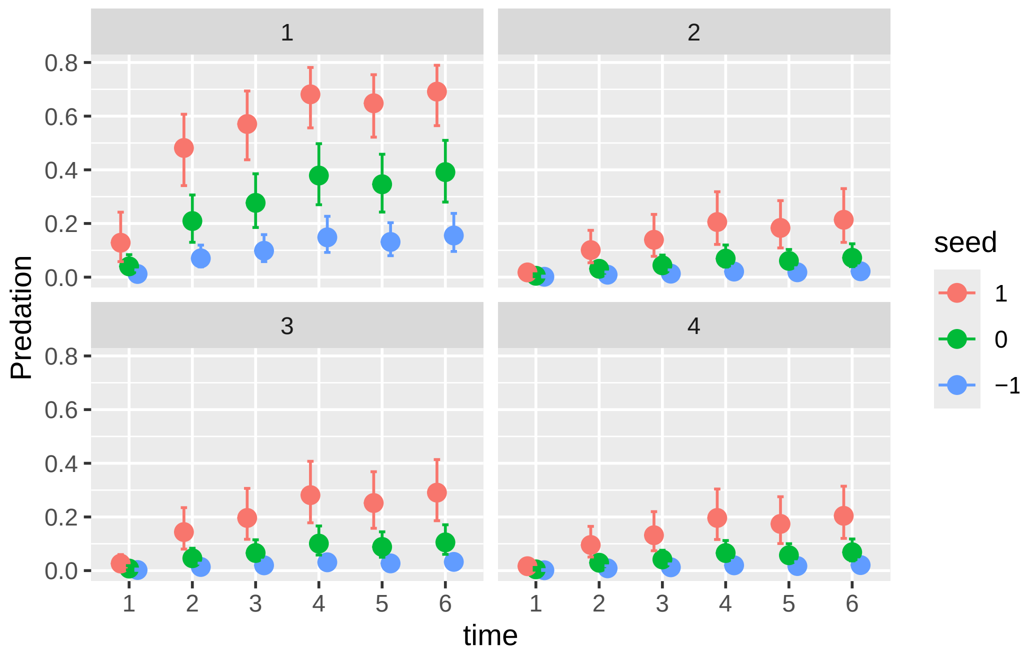




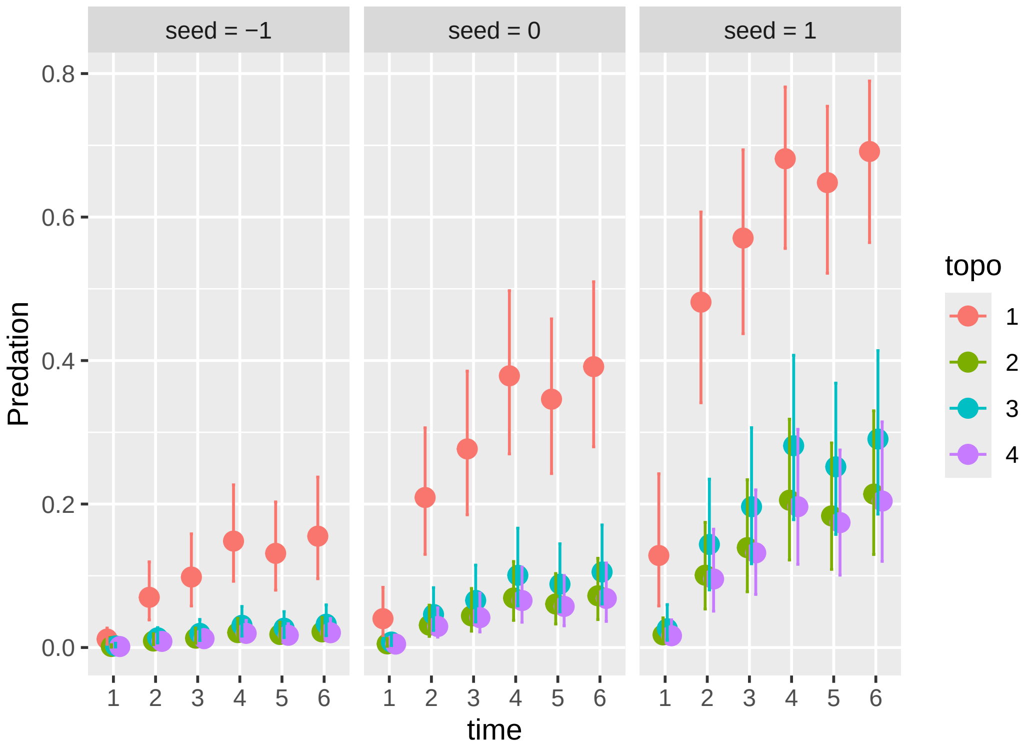
It already looks like predation rate is pretty constant in the last 3 campaigns.

By specifying `conditions=`, we can redo these plots for all levels of the 3rd predictor.

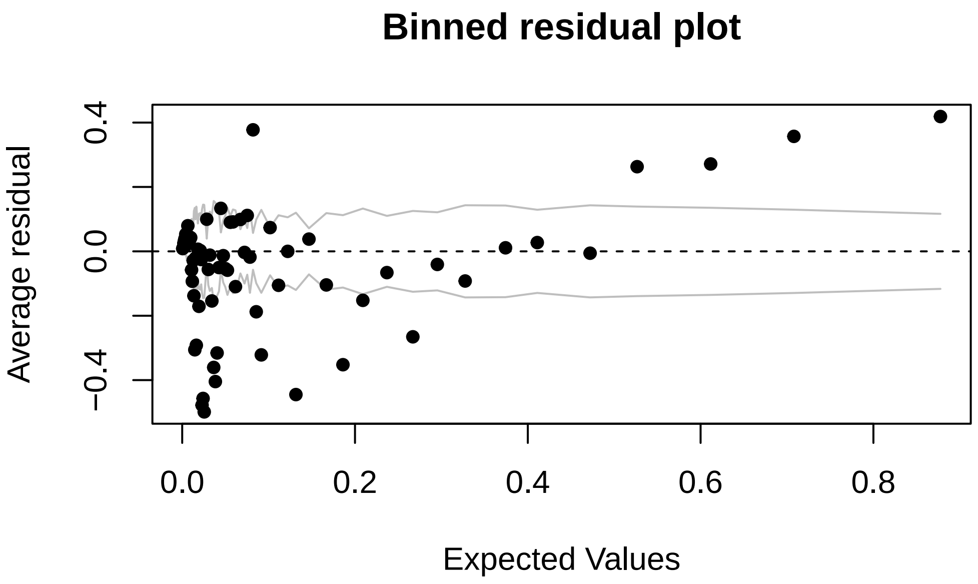
```
plot(conditional_effects(df.seed.3,
                         effect="time:seed",
                         conditions = make_conditions(df.seed.3,vars=c("topo"))))
```



```
plot(conditional_effects(df.seed.3,
                         effect="time:topo",
                         conditions = make_conditions(df.seed.3,vars=c("seed"))))
```



```
fitted = fitted(df.seed.3)
residuals = residuals(df.seed.3)
binnedplot(fitted, residuals)
```



Model comparison reveals that predation rate changes over time.

```r
LOO(df.seed.1, df.seed.2, df.seed.3)
```

```
          elpd_diff se_diff
df.seed.3   0.0       0.0  
df.seed.2 -28.4       5.3  
df.seed.1 -68.7      10.8  
```

But it still doesn't answer the question if predation rate reached a stable limit in the last 3 campaigns.

With emmeans, we can extract mean fitted values versus `time`, averaged over the remaing predictors. By default, the output is on linear scale, but it can also be displayed on response scale

```r
emmeans(df.seed.3, ~time)
```

```
 time emmean lower.HPD upper.HPD
 1     -4.66     -5.57     -3.86
 2     -2.83     -3.37     -2.30
 3     -2.46     -2.94     -1.95
 4     -1.99     -2.43     -1.53
 5     -2.13     -2.60     -1.70
 6     -1.94     -2.38     -1.45

Results are averaged over the levels of: topo 
Point estimate displayed: median 
Results are given on the logit (not the response) scale. 
HPD interval probability: 0.95 
```

```r
emmeans(df.seed.3, ~time, type="response")
```

```
 time response lower.HPD upper.HPD
 1     0.00934   0.00291    0.0188
 2     0.05570   0.03103    0.0873
 3     0.07878   0.04880    0.1218
 4     0.12035   0.07660    0.1713
 5     0.10606   0.06341    0.1488
 6     0.12567   0.07929    0.1803

Results are averaged over the levels of: topo 
```

```
Point estimate displayed: median
Results are back-transformed from the logit scale
HPD interval probability: 0.95
```

We could look at pairwise difference between all levels of `time` (all campaigns), but emmeans also can display only consecutive contrasts with `consec~...`

```
emmeans(df.seed.3, consec~time)
```

```
$emmeans
 time emmean lower.HPD upper.HPD
 1     -4.66     -5.57     -3.86
 2     -2.83     -3.37     -2.30
 3     -2.46     -2.94     -1.95
 4     -1.99     -2.43     -1.53
 5     -2.13     -2.60     -1.70
 6     -1.94     -2.38     -1.45

Results are averaged over the levels of: topo
Point estimate displayed: median
Results are given on the logit (not the response) scale.
HPD interval probability: 0.95

$contrasts
 contrast      estimate lower.HPD upper.HPD
 time2 - time1    1.840     0.902     2.812
 time3 - time2    0.378    -0.287     1.045
 time4 - time3    0.469    -0.132     1.147
 time5 - time4   -0.150    -0.770     0.451
 time6 - time5    0.190    -0.424     0.817

Results are averaged over the levels of: topo
Point estimate displayed: median
Results are given on the log odds ratio (not the response) scale.
HPD interval probability: 0.95
```

The contrasts between the last 3 campaigns are not different from 0 (95% CIs widely cover 0), so we can answer the research question with **yes**.

Alternatively, you could fit a model `Predation ~ seed + topo + time.combined`, where `time.combined` is the same as `time`, but levels 4,5,6 are combined in 1 level. Then do a model comparison versus the full model.