Practical 3: Priors & posteriors

Benjamin Rosenbaum

Table of contents

We learn about the posterior distribution and posterior predictions.
Posterior predictive checks are used for model evaluation.
Prior predictive checks are used to see if prior makes sense.

```
rm(list=ls())
library("brms")
library("bayesplot")
library("performance")
library("ggplot2")
library("ecostats")
try(dev.off())
```

Example 1: Global plants dataset

From the ecostats package.

This dataset contains observations of plant height vs latitude.
There is a negative relationship expected.
We perform a regression of log height vs latitude.

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

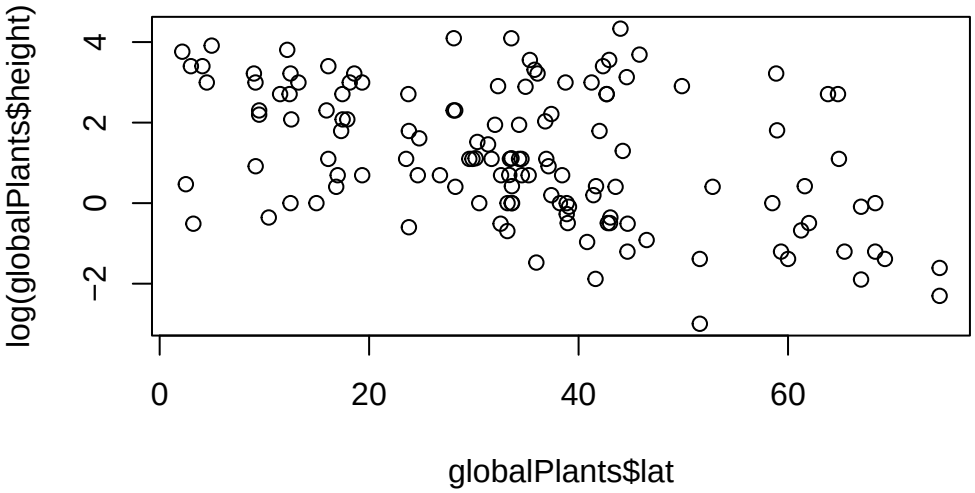
(We're ignoring all the other predictors for now)

```
data(globalPlants)
str(globalPlants)
```

```
'data.frame':   131 obs. of  33 variables:
 $ sort_number     : int  1402 25246 11648 8168 22422 25151 26007 6597 16908 4610 ...
 $ site            : int  193 103 54 144 178 27 118 154 106 201 ...
 $ Genus_species   : Factor w/ 175 levels "_8324","Abies_veitchii",..: 5 143 67 45 127 142 14
 $ Family          : Factor w/ 68 levels "Annonaceae","Asteraceae",..: 60 37 49 12 50 27 55 3
 $ growthform      : Factor w/ 6 levels "Fern","Herb",..: 6 6 2 4 2 4 6 6 2 4 ...
 $ height          : num  22.86 45 0.55 0.6 0.152 ...
 $ Country         : Factor w/ 37 levels "Argentina","Australia",..: 35 26 2 14 35 2 NA 35 2
 $ Site            : Factor w/ 127 levels "a-ngau","ab",..: 84 74 41 52 55 66 75 45 44 NA ..
 $ lat             : num  44.6 12.2 23.8 32.6 41.6 ...
 $ long            : num  -123.3 -70.5 133.8 34.9 -87 ...
 $ alt             : int  179 386 553 115 200 157 2 71 2 28 ...
 $ temp            : num  10.8 24.5 20.9 19.9 9.7 16.8 27.7 15.5 26.4 5.4 ...
 $ diurn.temp      : num  11.8 10.8 16.3 9.7 10.7 10 4.8 11.4 5 6.6 ...
 $ isotherm        : num  4.4 7.4 4.8 4.4 2.8 4.8 8.8 3.2 7.4 2.1 ...
 $ temp.seas       : num  5.2 0.9 6 4.9 9.7 3.9 0.2 8.6 0.6 8.3 ...
 $ temp.max.warm   : num  27 31.2 37 30.7 28.6 26.1 30.6 32.9 29.9 21.2 ...
 $ temp.min.cold   : num  0.3 16.7 3.6 8.7 -9.5 5.5 25.2 -2.6 23.2 -9 ...
 $ temp.ann.range  : num  26.7 14.5 33.4 22 38.1 20.6 5.4 35.5 6.7 30.2 ...
 $ temp.mean.wetqr : num  4.9 25.1 28.1 13.6 21.6 21.2 27.9 15.6 26.8 6.5 ...
 $ temp.mean.dryqr : num  17.4 23.2 14.8 25.3 -3.3 12.3 27.5 21.5 25.7 -1.6 ...
 $ temp.mean.warmqr: num  17.6 25.3 28.1 25.7 21.6 21.4 27.9 26.1 27.1 16.1 ...
 $ temp.mean.coldqr: num  4.5 23.1 12.8 13.6 -3.3 11.5 27.5 3.8 25.5 -5 ...
 $ rain            : int  1208 3015 278 598 976 1283 2585 1262 1704 664 ...
 $ rain.wetm       : int  217 416 37 159 104 157 300 129 309 77 ...
 $ rain.drym       : int  13 99 9 0 44 63 82 66 16 31 ...
 $ rain.seas       : int  69 45 42 115 23 29 34 18 66 28 ...
 $ rain.wetqr      : int  601 1177 109 408 299 450 870 382 806 220 ...
 $ rain.dryqr      : int  68 340 35 0 165 208 305 249 92 106 ...
 $ rain.warmqr     : int  75 928 109 2 299 385 855 268 659 191 ...
 $ rain.coldqr     : int  560 359 42 408 165 279 405 325 135 137 ...
```

```
 $ LAI             : num  2.51 4.26 1.32 1.01 3.26 4.14 NA 3.14 4.51 3.07 ...
 $ NPP             : int  572 1405 756 359 1131 1563 NA 1266 2296 536 ...
 $ hemisphere      : int  1 -1 -1 1 1 -1 1 1 -1 1 ...
```

```
plot(globalPlants$lat, log(globalPlants$height))
```



We fit model with weakly informative prior for negative slope. `class=b` identifies effect variables, `coef=lat` chooses this one predictor. We assume we have this information from similar studies. Note that the slope here is the change in log(*height*) when increasing one degree in *latitude*. If you scale the predictor or the response (e.g. standardize), you have to change the prior accordingly.

```
fit1 = brm(log(height) ~ lat,
           prior = prior(normal(-0.05,0.02), class=b, coef=lat),
           data=globalPlants )
```

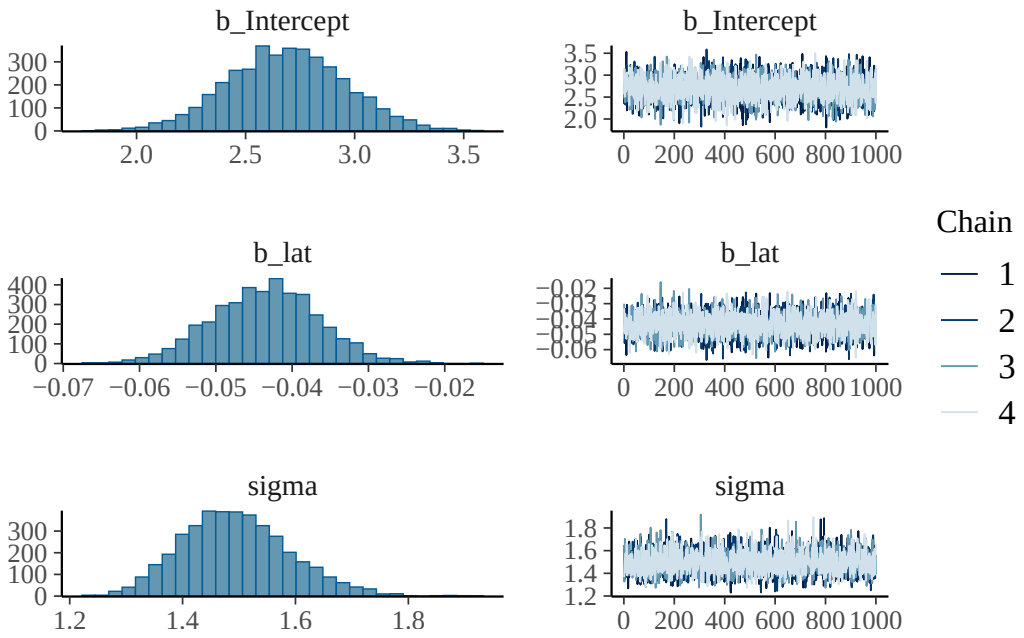
We check Rhat values and the traceplots for convergence.

```
summary(fit1, prior=TRUE)
```

```
 Family: gaussian
  Links: mu = identity; sigma = identity
Formula: log(height) ~ lat
   Data: globalPlants (Number of observations: 131)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000
```

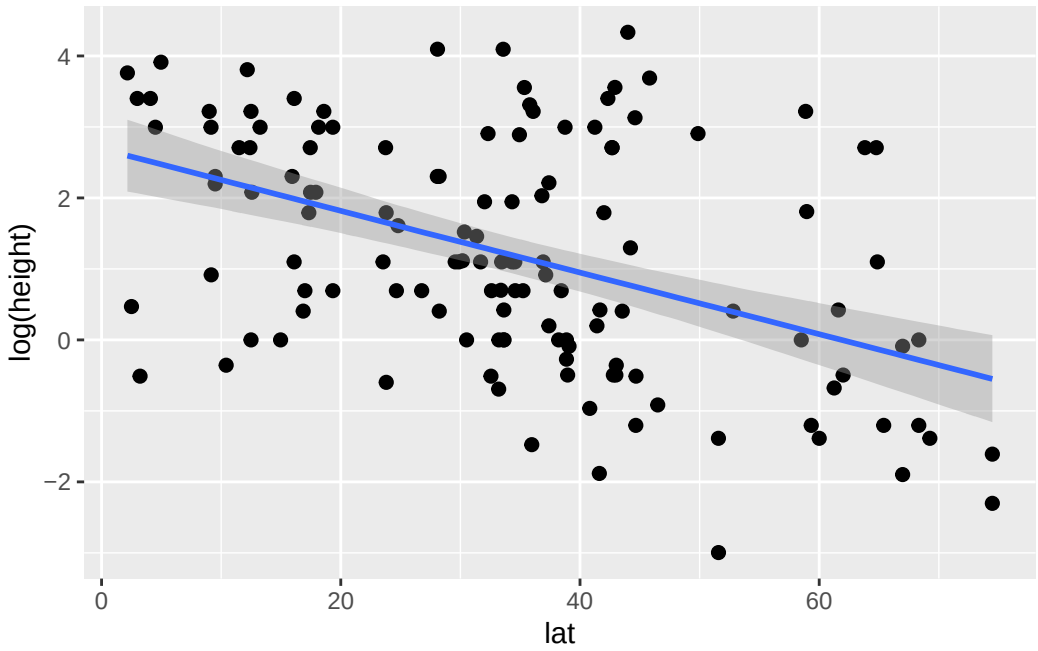
```
Priors:
b_lat ~ normal(-0.05, 0.02)
Intercept ~ student_t(3, 1.1, 2.5)
<lower=0> sigma ~ student_t(3, 0, 2.5)

Regression Coefficients:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept     2.69      0.27     2.16     3.22 1.00     4326     2735
lat          -0.04      0.01    -0.06    -0.03 1.00     4480     2979

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     1.50      0.10     1.33     1.70 1.00     3681     2920

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(fit1)
```



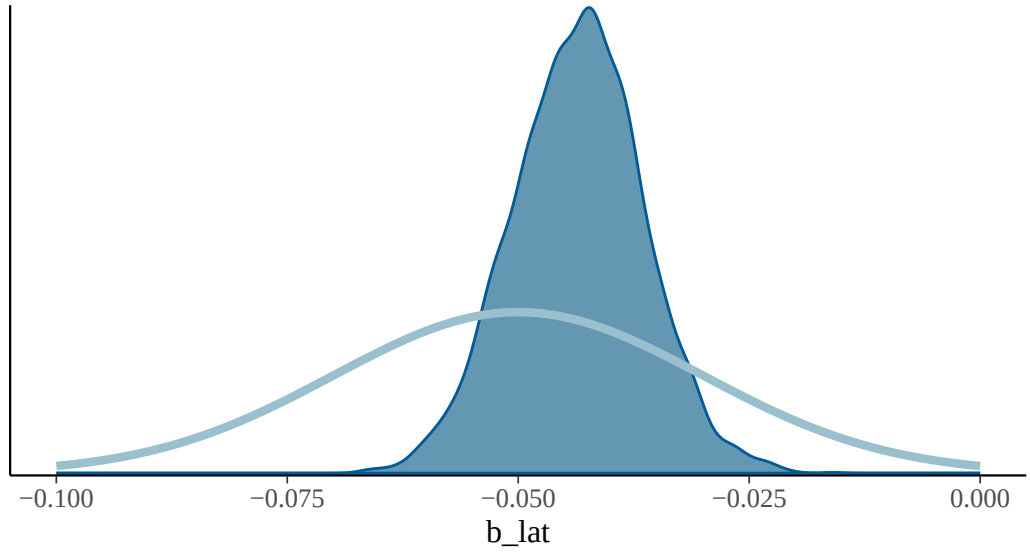
Fitted model predictions (deterministic part μ) are plotted against the data.

```
plot(conditional_effects(fit1), points=T)
```



Plotting prior & posterior for slope `b_lat`. You **can** also change the prior (smaller or bigger standard deviation) and re-fit the model, to see if that changes the posterior substantially (sensitivity analysis).

```
mcmc_dens(fit1, pars=c("b_lat")) +
  geom_function(fun=dnorm, args=list(-0.05, 0.02), colour="lightblue3", linewidth=1.5) +
  xlim(-0.1,0)
```



Next, we are looking at prior predictive checks (does the prior make sense for this data?) and posterior predictive checks (does the model fit the data well?). Usually, you first perform prior predictive checks, then fit the model, and then perform posterior predictive checks. But didactically it's more intuitive to show the posterior first.

The posterior distribution

What exactly is the posterior? It's just a collection of (multivariate) samples of model parameters. They can be extracted as a matrix `as_draws_matrix()` or dataframe `as_draws_df()` (rows=samples, columns=parameters).

```
post = as_draws_df(fit1)
str(post)
```

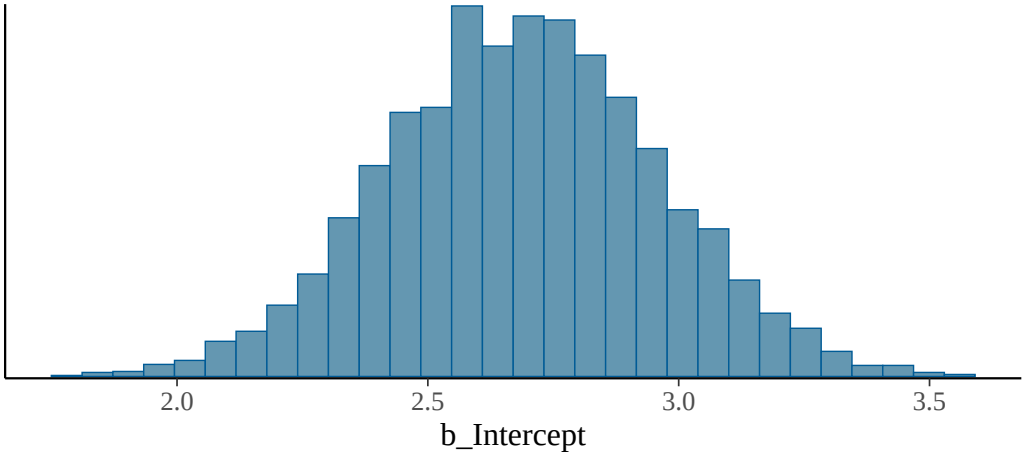
```
draws_df [4,000 x 9] (S3: draws_df/draws/tbl_df/tbl/data.frame)
 $ b_Intercept: num [1:4000] 2.67 2.57 2.64 2.47 2.61 ...
 $ b_lat      : num [1:4000] -0.04 -0.0441 -0.0388 -0.0363 -0.0382 ...
 $ sigma      : num [1:4000] 1.56 1.34 1.65 1.52 1.36 ...
 $ Intercept  : num [1:4000] 1.3 1.06 1.32 1.23 1.3 ...
 $ lprior     : num [1:4000] -0.523 -0.375 -0.581 -0.616 -0.516 ...
 $ lp__       : num [1:4000] -237 -238 -239 -237 -238 ...
 $ .chain     : int [1:4000] 1 1 1 1 1 1 1 1 1 1 ...
 $ .iteration : int [1:4000] 1 2 3 4 5 6 7 8 9 10 ...
 $ .draw      : int [1:4000] 1 2 3 4 5 6 7 8 9 10 ...
```

```
head(post)
```

```
# A draws_df: 6 iterations, 1 chains, and 6 variables
  b_Intercept  b_lat sigma Intercept lprior lp__
1         2.7 -0.040   1.6       1.3  -0.52 -237
2         2.6 -0.044   1.3       1.1  -0.38 -238
3         2.6 -0.039   1.6       1.3  -0.58 -239
4         2.5 -0.036   1.5       1.2  -0.62 -237
5         2.6 -0.038   1.4       1.3  -0.52 -238
6         2.7 -0.048   1.5       1.0  -0.38 -237
# ... hidden reserved variables {'.chain', '.iteration', '.draw'}
```

For example, we can look at the distribution of the intercept, which is the fitted log(height) at the equator (lat=0). We plot a histogram using a function from the `bayesplot` package

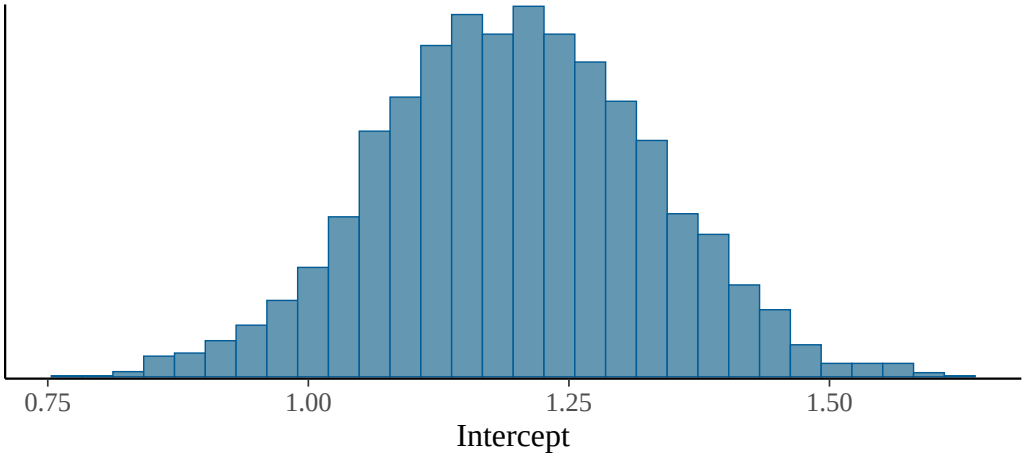
```
mcmc_hist(fit1, pars="b_Intercept")
```



The posterior contains 2 variables for intercept? What's going on?

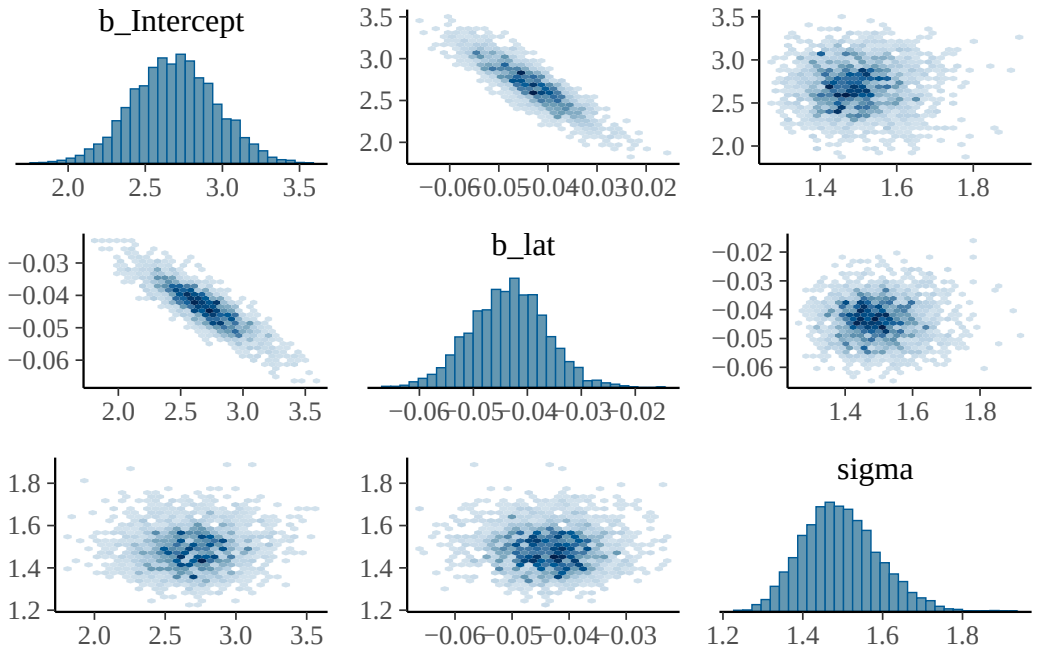
The other variable `Intercept` is used by brms internally for model fitting and describes the intercept for mean centered predictors. It is the fitted log(height), where the predictor is at its empirical mean (lat=mean(lat))

```
mcmc_hist(fit1, pars="Intercept")
```



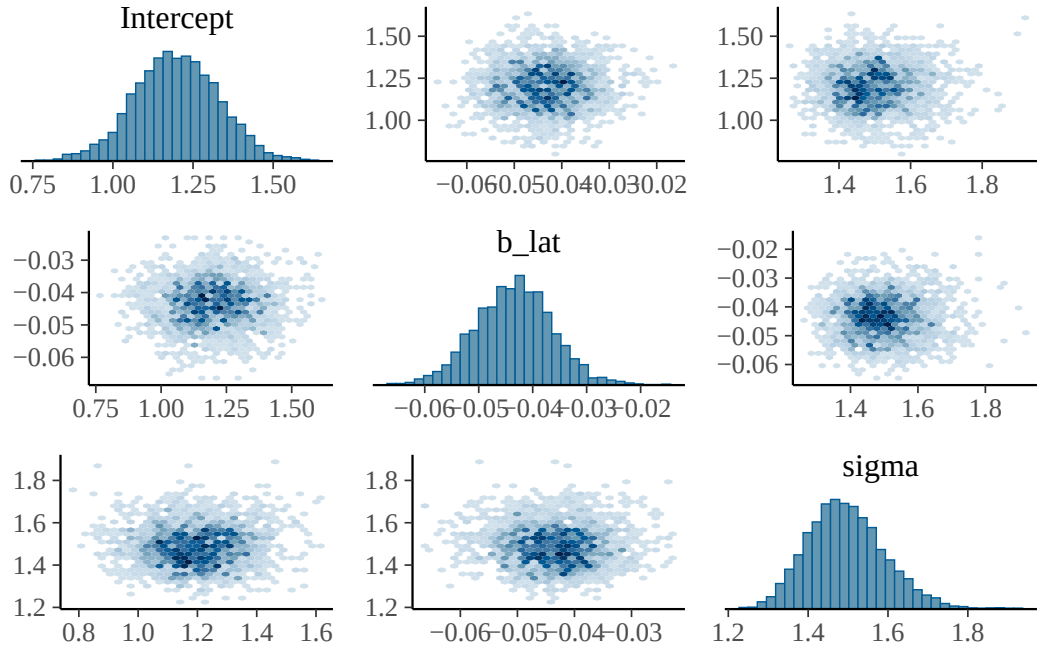
Don't forget that the posterior is a multivariate distribution. When looking at a pairs plot, we observe the classical correlation between intercept (at lat=0) and slope samples.

```
pairs(fit1, variable=c("b_Intercept", "b_lat", "sigma"), off_diag_fun="hex")
```



The correlation disappears if the model is parametrized with mean-centered predictors. It can make the MCMC more efficient, but the statistical model is equivalent.

```
pairs(fit1, variable=c("Intercept", "b_lat", "sigma"), off_diag_fun="hex")
```



Posterior predictions

For any given value of the predictor, predictions can be computed from the samples of the posterior. If we just use the deterministic model part, this is called the fitted distribution (on μ-level), when we additionally include the stochastic model part (here normal distribution with sd=σ), this is called the predictive distribution (on response-level y).

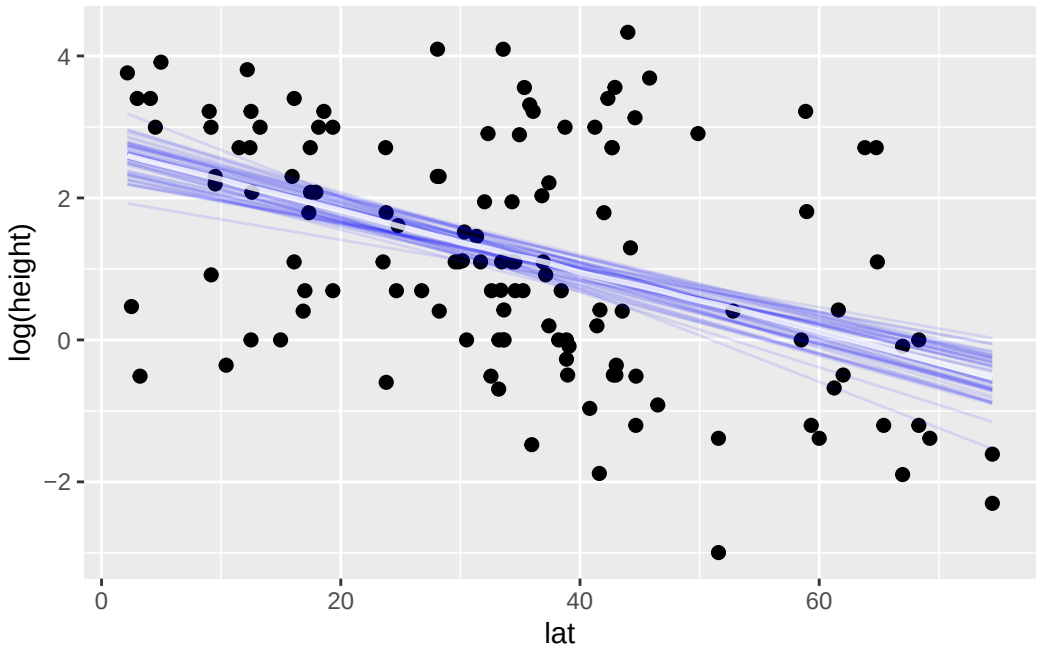
Fitted distribution (deterministic part)

Posterior predictions for $\mu = a + b \cdot x$. Each sample (row in the posterior) produces a regression line

```
head(post[, 1:3])
```

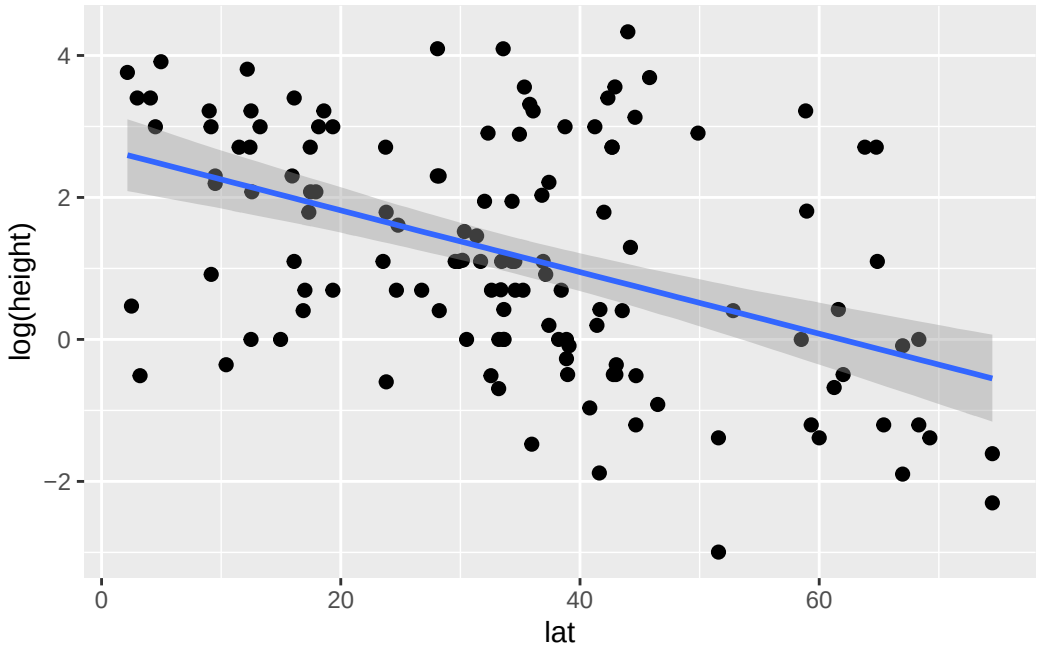
```
# A tibble: 6 x 3
  b_Intercept   b_lat sigma
        <dbl>   <dbl> <dbl>
1        2.67 -0.0400  1.56
2        2.57 -0.0441  1.34
3        2.64 -0.0388  1.65
4        2.47 -0.0363  1.52
5        2.61 -0.0382  1.36
6        2.69 -0.0479  1.49
```

```
plot(conditional_effects(fit1, spaghetti=TRUE, ndraws=50), points=TRUE)
```



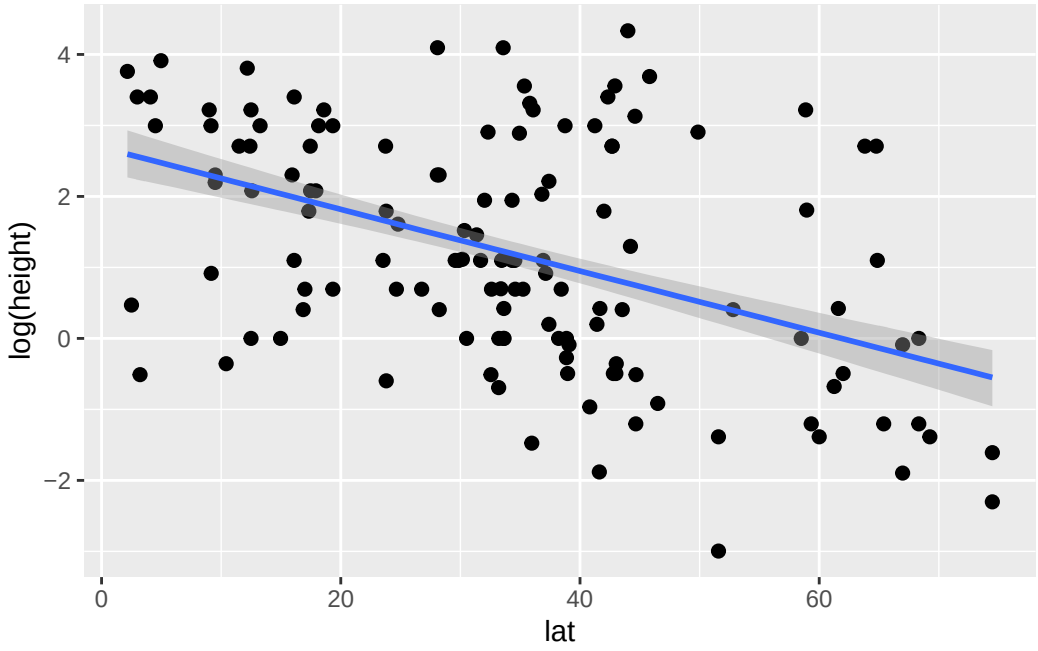
95% credible intervals (the default) display uncertainty of the mean regression line μ

```
plot(conditional_effects(fit1), points=TRUE)
```



If needed, any other uncertainty level can be chosen, e.g. 80%:

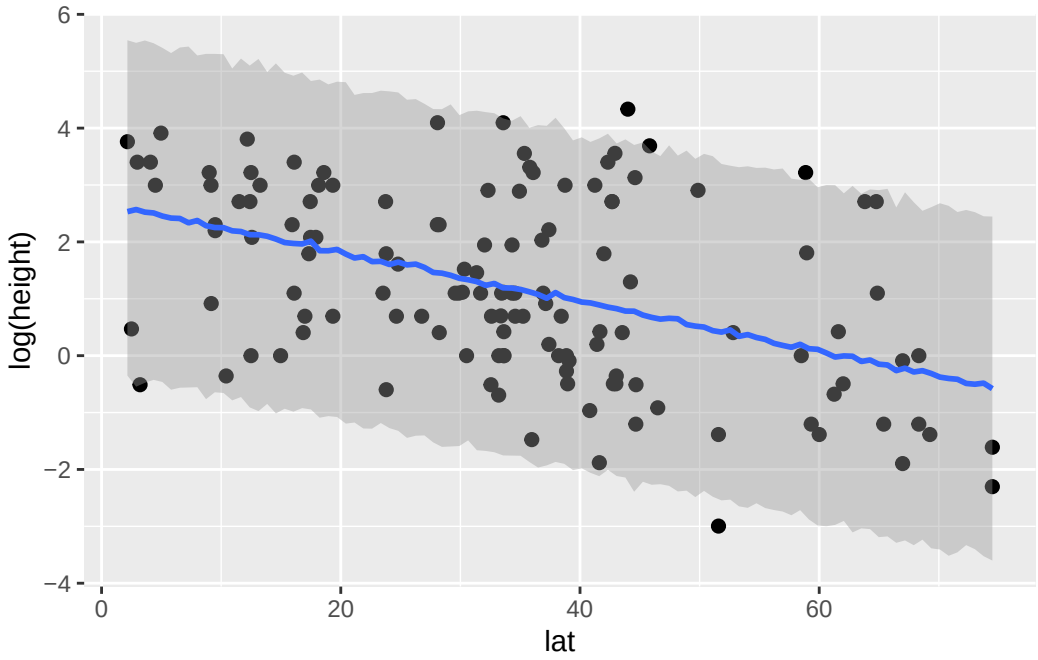
```
plot(conditional_effects(fit1, prob=0.80), points=TRUE)
```



Predicted distribution (deterministic + stochastic part)

Posterior predictions for response $y = \mu + \varepsilon$, with $\varepsilon \sim \text{Normal}(0, \sigma)$. The 95% prediction intervals should contain around 95% of datapoints.

```
plot(conditional_effects(fit1, method="posterior_predict"), points=TRUE)
```



The functions `fitted()` and `predict()` extract fitted and prediction means & intervals for all observations (each row = 1 datapoint)

```
fitted(fit1) |> round(3) |> head()
```

```
     Estimate Est.Error  Q2.5 Q97.5
[1,]    0.748     0.150 0.459 1.039
[2,]    2.159     0.199 1.770 2.548
[3,]    1.653     0.147 1.362 1.944
[4,]    1.272     0.130 1.015 1.524
[5,]    0.878     0.141 0.605 1.153
[6,]    1.225     0.130 0.966 1.474
```

```
predict(fit1) |> round(3) |> head()
```

```
     Estimate Est.Error   Q2.5 Q97.5
[1,]    0.746     1.507 -2.241 3.755
[2,]    2.127     1.518 -0.820 5.112
[3,]    1.634     1.484 -1.240 4.553
[4,]    1.255     1.480 -1.651 4.129
[5,]    0.883     1.515 -2.110 3.896
[6,]    1.220     1.520 -1.811 4.203
```

They can also produce fitted and predicted values for any new datapoint, e.g. `lat=60`

```
fitted(fit1, newdata=data.frame(lat=60)) |> round(3)
```

```
     Estimate Est.Error   Q2.5 Q97.5
[1,]    0.078     0.226 -0.355 0.519
```

```
predict(fit1, newdata=data.frame(lat=60)) |> round(3)
```

```
     Estimate Est.Error   Q2.5 Q97.5
[1,]    0.085     1.532 -2.934  3.15
```

Inference

Depending on the scientific question or hypothesis, we can make inference on model parameters, or on model predictions (note that predictions are just derived quantities from model parameters)

The `hypothesis()` function can be used for inference on parameters. Here, column `Post.Prob` counts the ratio of samples that corresponds to the hypothesis (`b_lat<0`), in this case 100%.

```
hypothesis(fit1, "lat<0")
```

```
Hypothesis Tests for class b:
  Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio Post.Prob Star
1  (lat) < 0    -0.04      0.01    -0.06    -0.03        Inf         1    *
---
'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
'*': For one-sided hypotheses, the posterior probability exceeds 95%;
for two-sided hypotheses, the value tested against lies outside the 95%-CI.
Posterior probabilities of point hypotheses assume equal prior probabilities.
```

But what can we do if we want to test a hypothesis on predictions? What is the expected difference in log(height) between 40 and 60 degree latitude $\mu(40) - \mu(60)$?

Just looking at 2 distributions of fitted values at 40 and 60 degrees latitude doesn't answer our question.

```
fitted(fit1, newdata=data.frame(lat=40)) |> round(3)
```

```
     Estimate Est.Error  Q2.5 Q97.5
[1,]    0.948     0.137 0.681 1.213
```

```
fitted(fit1, newdata=data.frame(lat=60)) |> round(3)
```

```
     Estimate Est.Error   Q2.5 Q97.5
[1,]    0.078     0.226 -0.355 0.519
```

We need to look at the posterior distribution of their difference!

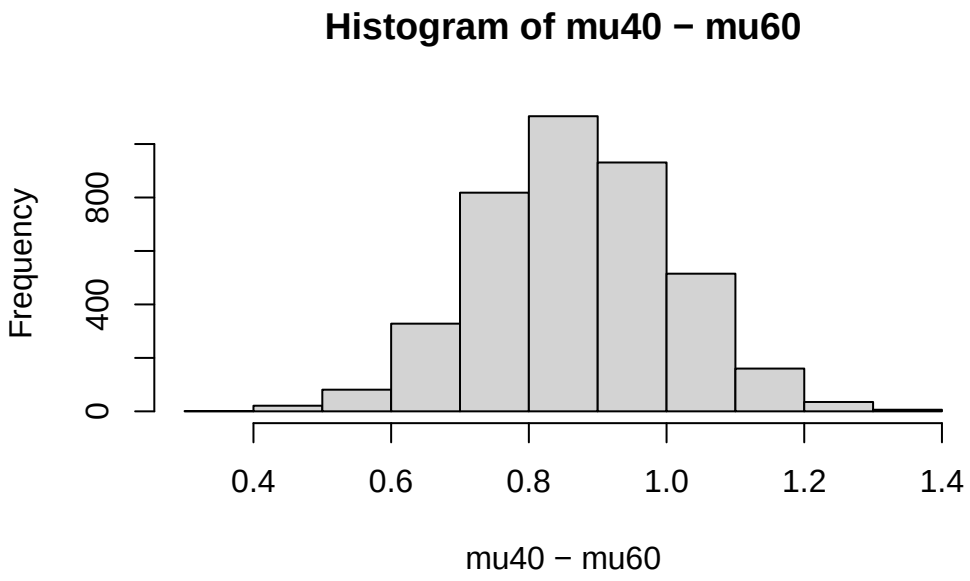
To extract full all posterior samples of the fitted values, we can either use `fitted(..., summary=FALSE)` or use the `posterior_epred()` function. Both give the same output.

```
mu40 = posterior_epred(fit1, newdata=data.frame(lat=40))
mu60 = posterior_epred(fit1, newdata=data.frame(lat=60))
cbind(mu40,mu60) |> head()
```

```
          [,1]        [,2]
[1,] 1.0717166  0.27178387
[2,] 0.8016191 -0.08117541
[3,] 1.0932128  0.31806955
[4,] 1.0214892  0.29490351
[5,] 1.0792303  0.31604986
[6,] 0.7738572 -0.18322599
```

Now we have the full posterior distribution of the difference $\mu(40) - \mu(60)$ and can look at its distribution. We can calculate mean difference, and quantiles etc. Also, the probability of the statement $\mu(40) > \mu(60)$ is the number of samples satifying this condition, divided by total number of samples.

```
hist(mu40-mu60)
```




```
mean(mu40-mu60)
```

```
[1] 0.8702583
```

```
sd(mu40-mu60)
```

```
[1] 0.1404822
```

```
quantile(mu40-mu60, probs=c(0.05, 0.95))
```

```
       5%       95%
0.6374935 1.1000554
```

```
sum(mu40>mu60)/length(mu40)
```

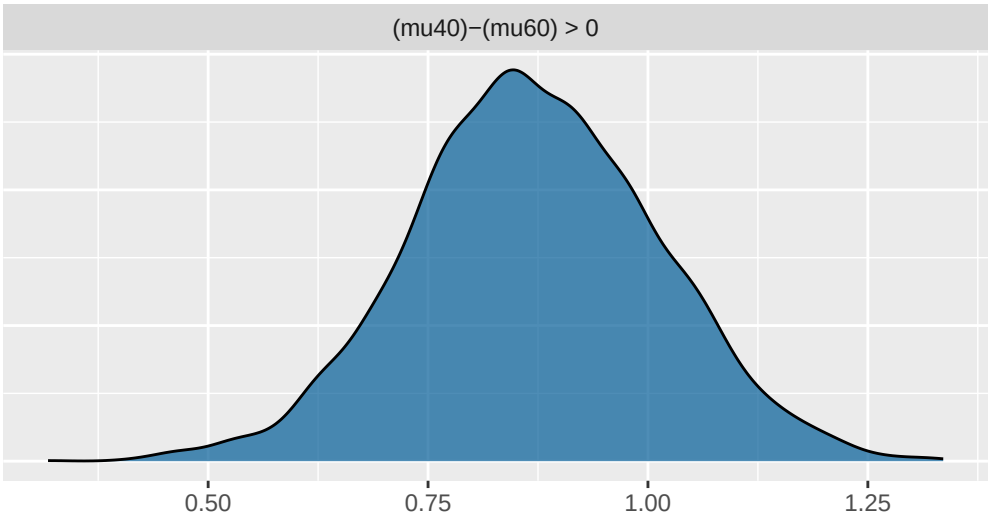
```
[1] 1
```

Alternatively, we can also use the `hypothesis()` function for comparing predictions. We have to extract posterior fitted values $\mu(40)$ and $\mu(60)$ in **one** object, give them names and use these names in the hypothesis statement.

```
mus = fitted(fit1,
             newdata=data.frame(lat=c(40,60)),
             summary=F)
mus = as.data.frame(mus)
names(mus)=c("mu40","mu60")
hypothesis(mus, "mu40>mu60")
```

```
Hypothesis Tests for class :
         Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio Post.Prob Star
1 (mu40)-(mu60) > 0     0.87      0.14     0.64      1.1        Inf         1    *
---
'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
'*': For one-sided hypotheses, the posterior probability exceeds 95%;
for two-sided hypotheses, the value tested against lies outside the 95%-CI.
Posterior probabilities of point hypotheses assume equal prior probabilities.
```

```
plot(hypothesis(mus, "mu40>mu60"))
```



We get the same results as above.

Posterior predictive checks

How well does the model fit the data?

Bayes R2 quantifies how much of the variation (in data y) is explained by the model (deterministic part μ). It is calculated a bit different from frequentist R2, but it means the same.

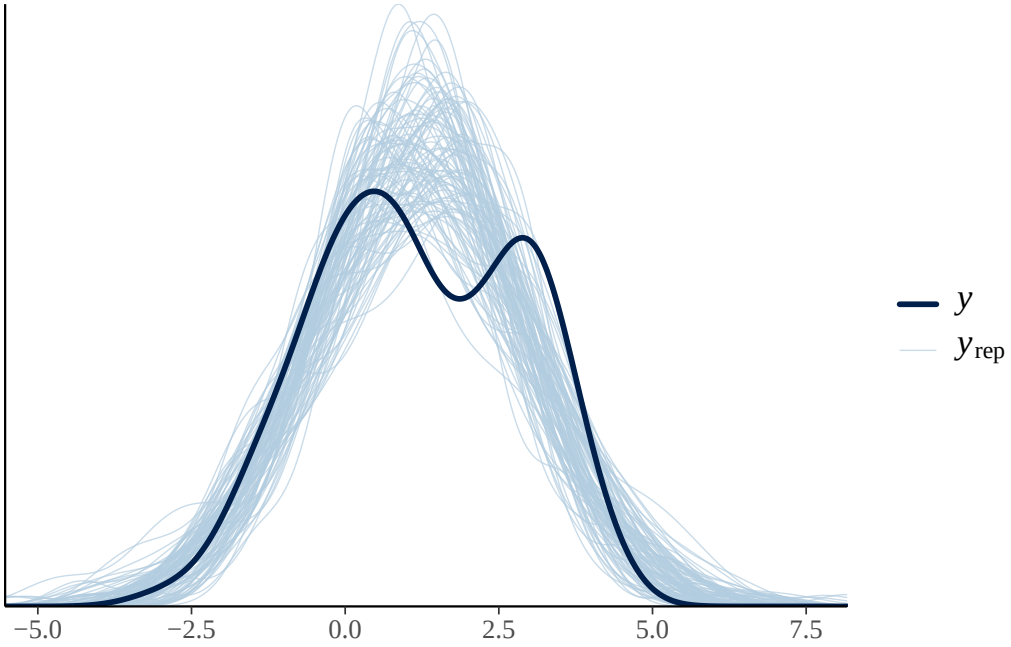
```
bayes_R2(fit1)
```

```
    Estimate Est.Error      Q2.5     Q97.5
R2 0.2090685  0.051546 0.1092974 0.3085452
```

A helpful graphical tool is the posterior predictive check. A histogram of observed data y is plotted against histograms of predicted data (including det. and stoch. part). Each of the 100 lines is a simulated dataset from 1 sample of the posterior.

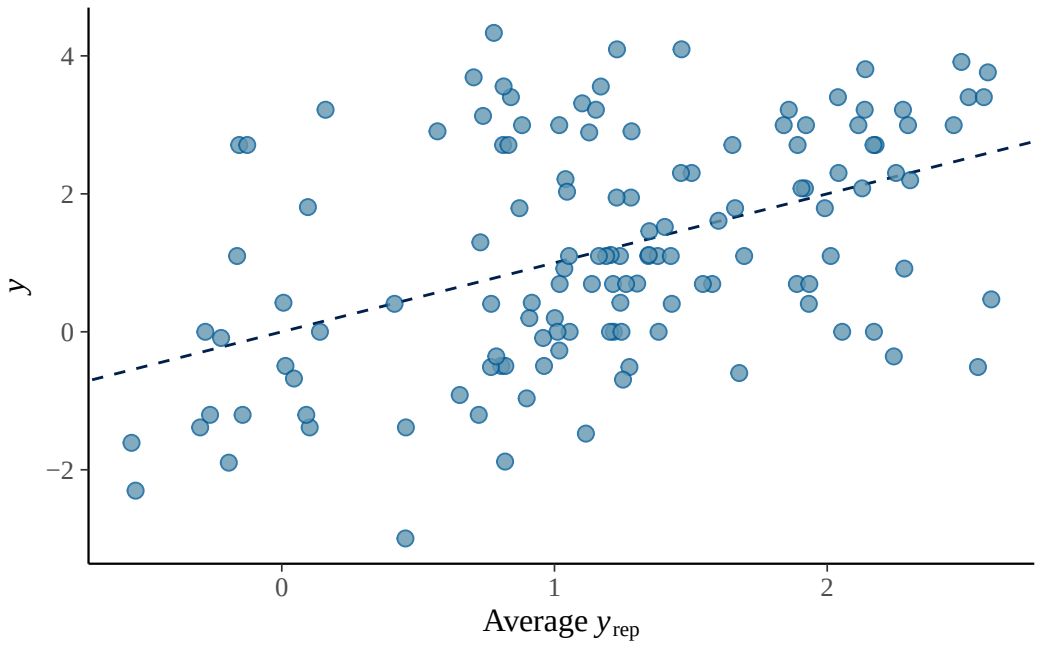
Here, the simple linear regression model can't reproduce the "dip" in response values in the middle. This could hint at a missing predictor. Strong deviations from expexted behavior could hint at a misspecified model. E.g. likelihood function (normal) is not appropriate, and some GLM would be a better choice.

```
pp_check(fit1, ndraws=100)
```



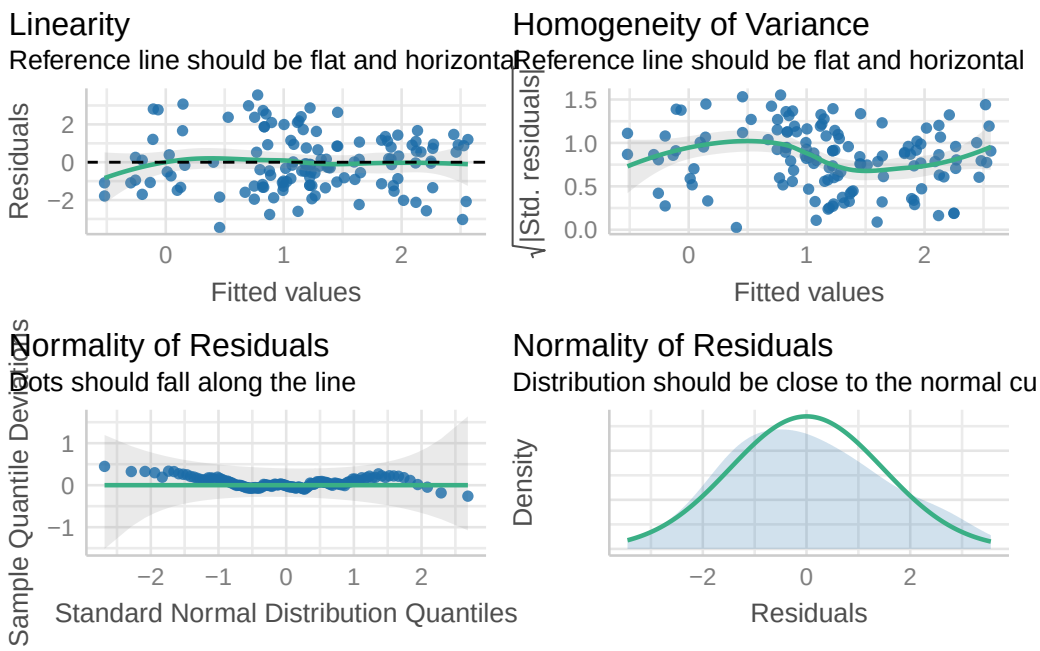
The classical observed vs predicted plot, where dots close to the diagonal 1-1-line would indicate a good model fit.

```
pp_check(fit1, type="scatter_avg")
```



The `performance` package (mostly) supports brms and includes some helpful tools for checking (linear) models. More on that in the next session.

```
check_model(fit1, check=c("linearity","homogeneity","qq","normality"))
```



Deviations

Prior predictive checks

Now that we understand how to do posterior predictive checks, we can also do prior predictive checks.

Unless we have prior information from previous studies / experiments, how do we know if we have a good / meaningful prior? Parameters sampled from the prior distribution should generate predictions which are roughly in the range of observed data. Sure, the information contained in the data goes into the model via the likelihood, but priors that generate predictions of a total different magnitude as the data do not make much sense.

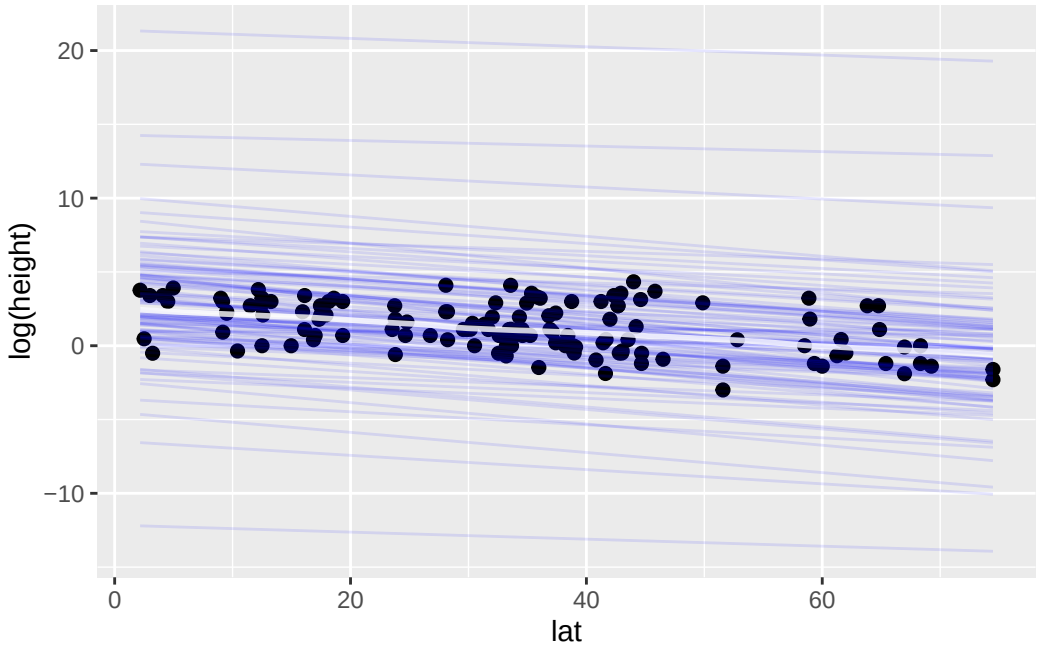
Instead of sampling from the posterior = prior * likelihood, we can just sample from the prior. Why? We can check if predictions from prior are approximately in line with the data. For linear models this may be obvious (intercept & slope), but for GLMs etc. prior predictive checks can be a helpful tool.

We start with the previous model, where we used a custom prior for the slope and default priors for intercept and sd. By specifying `sample_prior="only"`, parameters are only sampled from the prior, ignoring the likelihood and therefore the data.

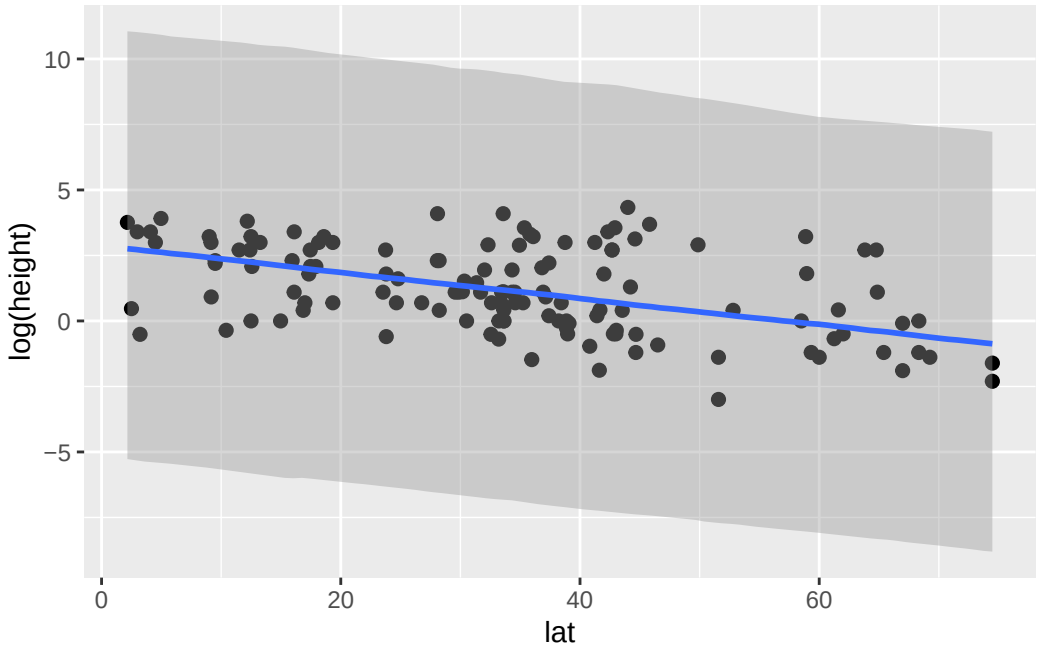
```
fit1.prior = brm(log(height) ~ lat,
                 prior = prior(normal(-0.05,0.02), class=b, coef=lat),
                 sample_prior = "only",
                 data=globalPlants )
```

Now we can use the same tools as in posterior predictive checks to compare predictions with observed data.

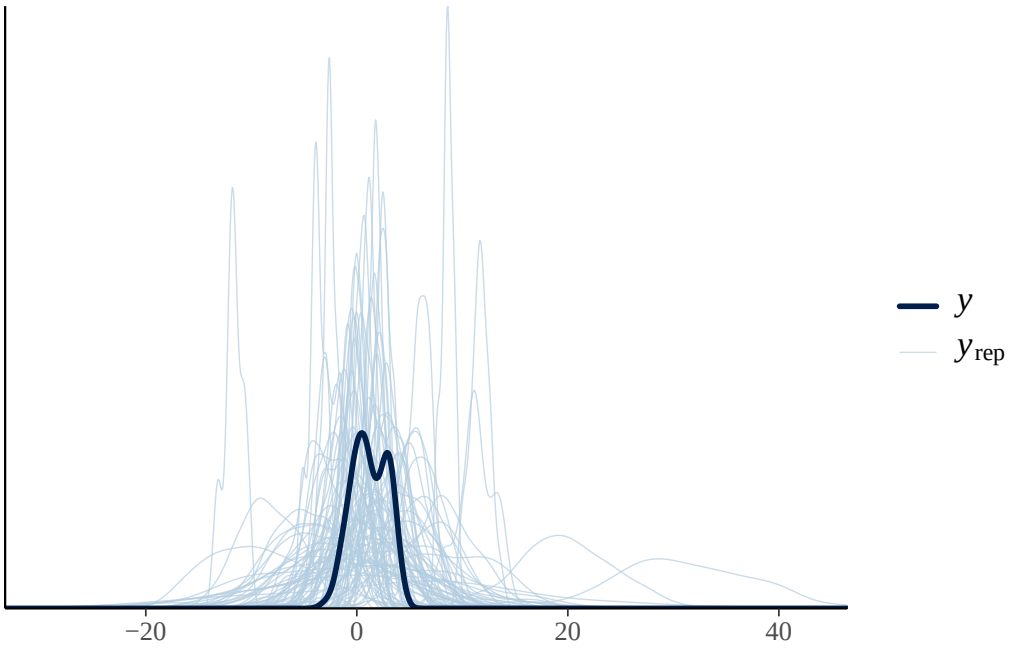
```
plot(conditional_effects(fit1.prior, spaghetti=TRUE, ndraws=100), points=TRUE)
```




```
plot(conditional_effects(fit1.prior), points=TRUE)
```



```
pp_check(fit1.prior, ndraws=100)
```

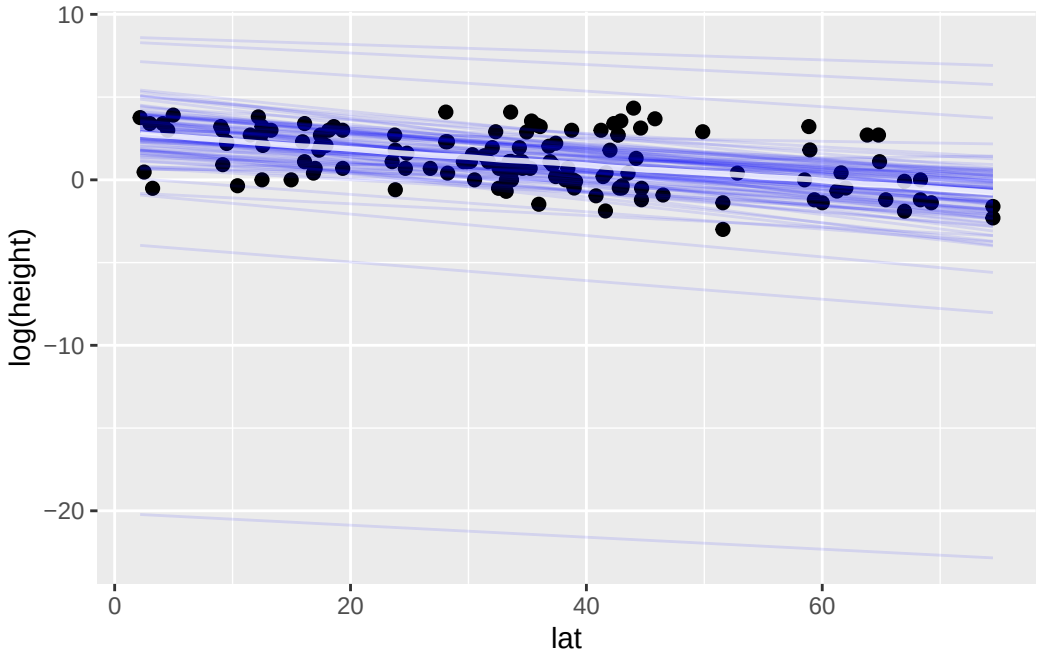


Prior predictions are covering the data well. They are the same order of magnitude, which is good. At this point, we could stop and use this as a weakly informative prior.

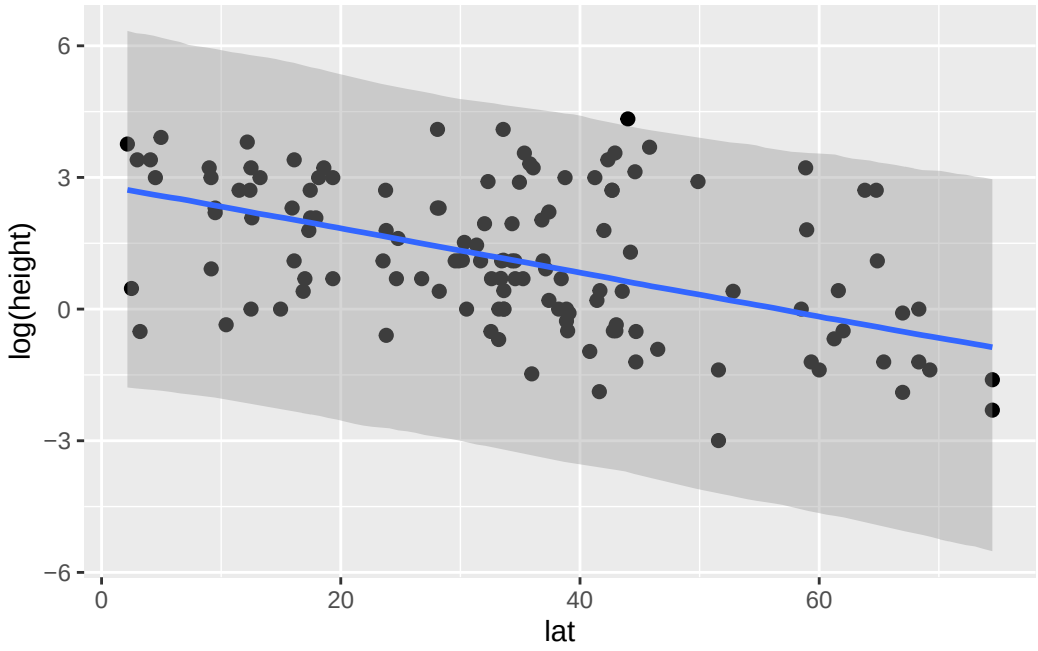
Alternatively, we could tighten the prior a bit (especially brms default for intercept), but it's actually not that important here.

```
fit2.prior = brm(log(height) ~ lat,
                 prior = c(prior(normal(-0.05,0.02), class=b, coef=lat),
                           prior(student_t(3, 1.1, 1), class=Intercept)),
                 sample_prior = "only",
                 data=globalPlants )
```

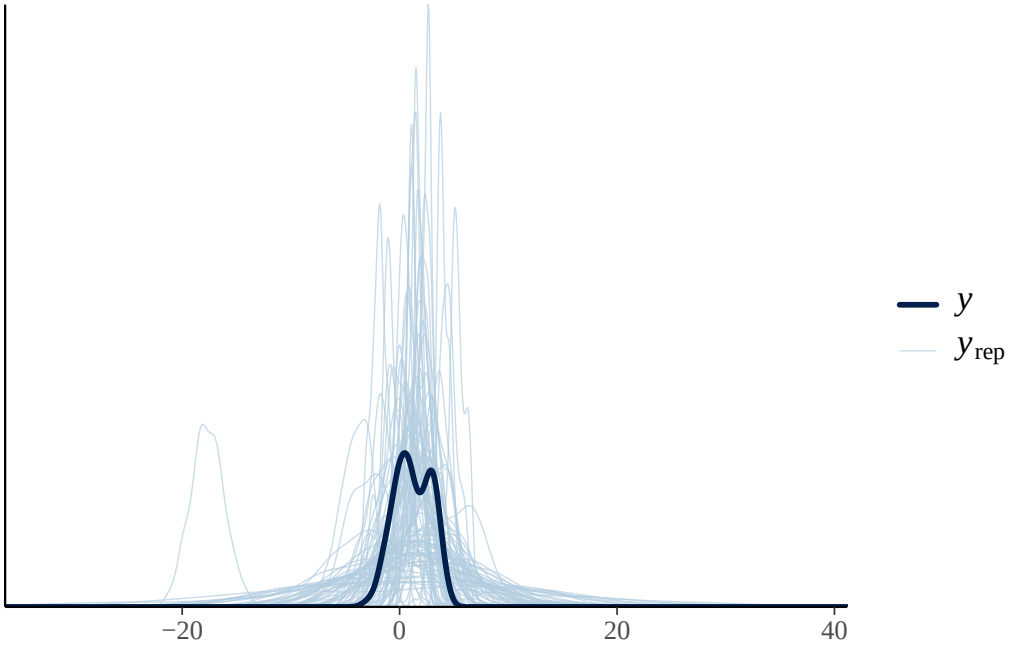
```
plot(conditional_effects(fit2.prior, spaghetti=TRUE, ndraws=100), points=TRUE)
```




```
plot(conditional_effects(fit2.prior), points=TRUE)
```




```
pp_check(fit2.prior, ndraws=100)
```



Finally, we fit the model with the new prior distribution.

```
fit2 = brm(log(height) ~ lat,
           prior = c(prior(normal(-0.05,0.02), class=b, coef=lat),
                     prior(student_t(3, 1.1, 1), class=Intercept)),
           data=globalPlants )
```

```
summary(fit2, prior=TRUE)
```

```
 Family: gaussian
  Links: mu = identity; sigma = identity
Formula: log(height) ~ lat
   Data: globalPlants (Number of observations: 131)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Priors:
b_lat ~ normal(-0.05, 0.02)
Intercept ~ student_t(3, 1.1, 1)
<lower=0> sigma ~ student_t(3, 0, 2.5)

Regression Coefficients:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
```

```
Intercept     2.69      0.27     2.15     3.22 1.00     4163     3008
lat          -0.04      0.01    -0.06    -0.03 1.00     4598     3134

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     1.50      0.10     1.32     1.70 1.00     3948     2834

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).
```

Here, prior choice affects the posterior only marginally (fit1 vs fit2).

Exercise 1: Water quality dataset

Analyze the data of water quality in several rivers vs catchment area (log), to test if river health deteriorates downstream (higher catchment area).

Now perform previous steps in the correct logical order

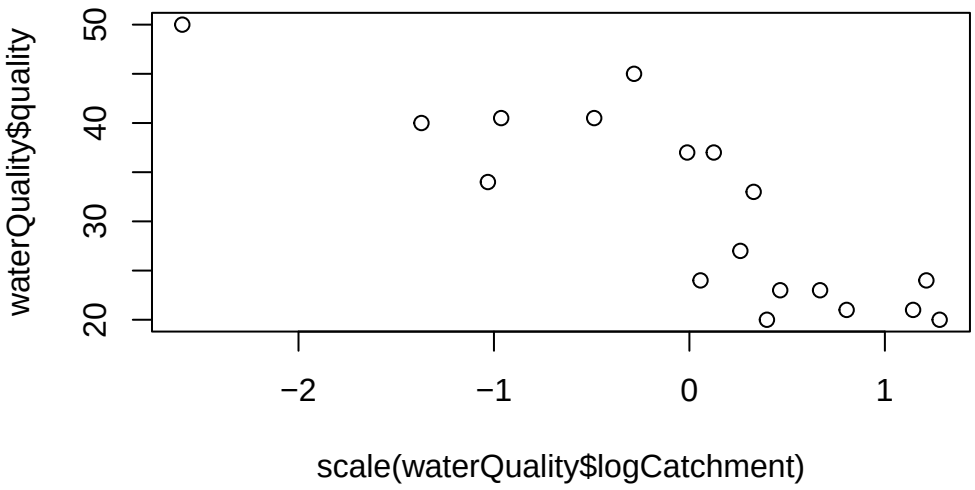
1. check / plot data. scale predictor with `scale()`
2. assign priors + prior predictive checks
3. fit model
4. check convergence
5. check model fit
6. inference: What is the difference (and its 90%-quantile) in water quality between mean and mean+1sd of the predictor?

(1) Check data

```
data("waterQuality")
str(waterQuality)
```

```
'data.frame':   18 obs. of  3 variables:
 $ catchment   : num  100 794 1413 1585 3548 ...
 $ quality     : num  50 40 34 40.5 40.5 45 37 24 37 27 ...
 $ logCatchment: num  2 2.9 3.15 3.2 3.55 3.7 3.9 3.95 4 4.1 ...
```

```
plot(scale(waterQuality$logCatchment), waterQuality$quality)
```




```
waterQuality$log.catch.z = scale(waterQuality$logCatchment)
```

(2) Assign priors

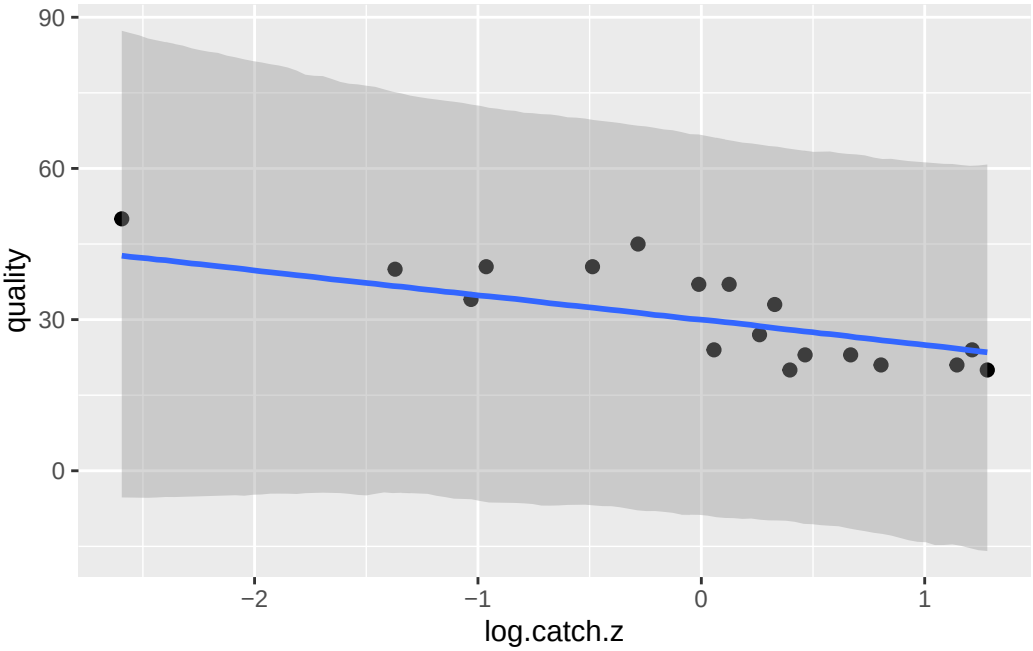
```
options(width = 200)
default_prior(quality ~ log.catch.z,
              data = waterQuality)
```

```
                  prior     class        coef group resp dpar nlpar lb ub       source
                 (flat)         b                                              default
                 (flat)         b log.catch.z                                (vectorized)
 student_t(3, 30, 11.9) Intercept                                              default
  student_t(3, 0, 11.9)     sigma                                  0          default
```

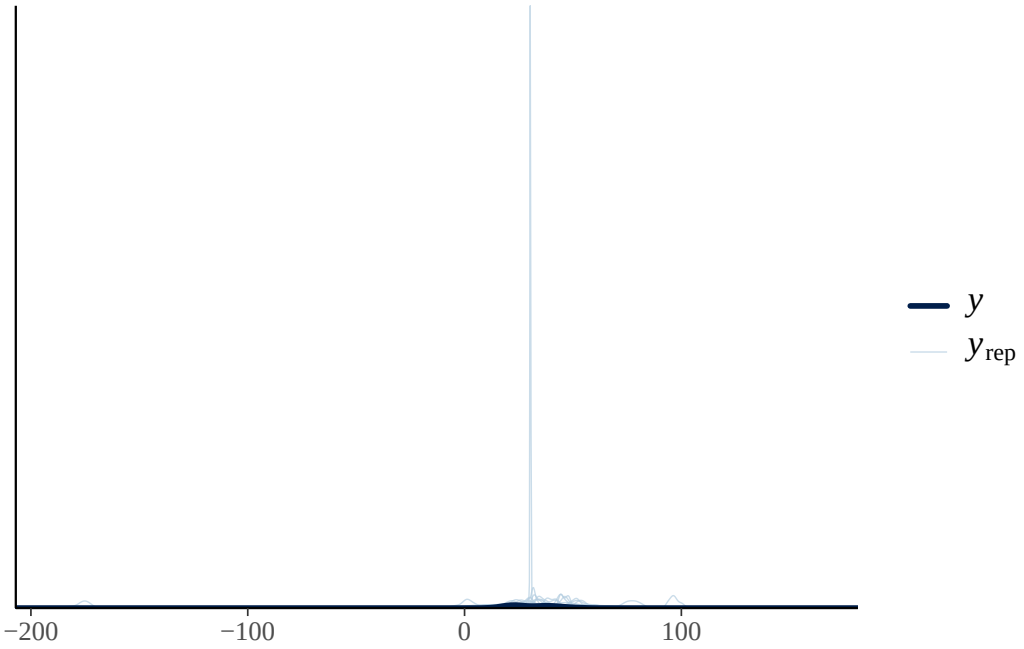
We assign weakly informative priors, expecting a negative trend. A prior predictive check is performed.

```
fit1.prior = brm(quality ~ log.catch.z,
                 prior = prior(normal(-5,5), class=b, coef=log.catch.z),
                 sample_prior = "only",
                 data = waterQuality )
```

```
plot(conditional_effects(fit1.prior), points=TRUE)
```




```
pp_check(fit1.prior, ndraws=100)
```

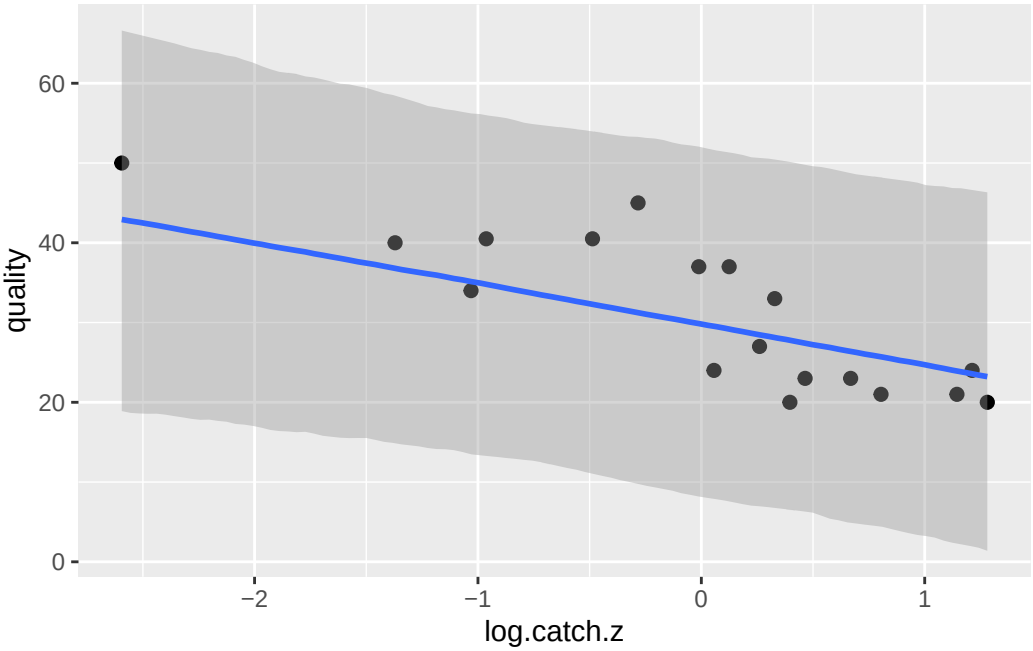


The prior predictions cover the range of the data, but sometimes predictions can be really far off.

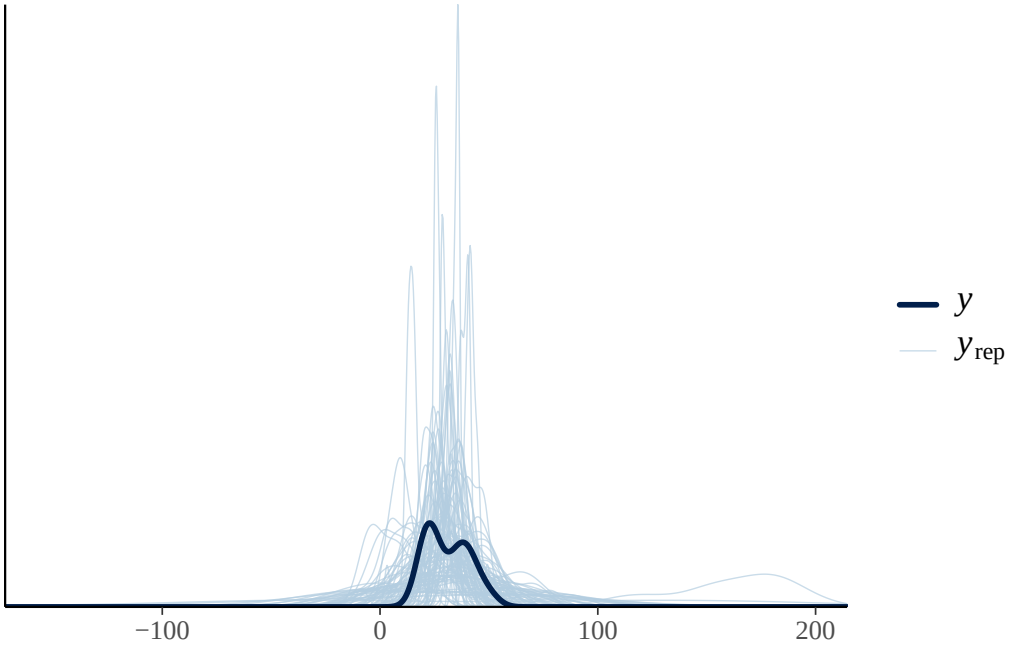
We tighten the prior for the slope and also the intercept a bit.

```
fit2.prior = brm(quality ~ log.catch.z,
                 prior = c(prior(normal(-5,2.5), class=b, coef=log.catch.z),
                           prior(student_t(3, 30, 6), class=Intercept)),
                 sample_prior = "only",
                 data = waterQuality)
```

```
plot(conditional_effects(fit2.prior), points=TRUE)
```




```
pp_check(fit2.prior, ndraws=100)
```



This looks a bit better and we will use this prior.

(3) Fit model

```
fit2 = brm(quality ~ log.catch.z,
           prior = c(prior(normal(-5,2.5), class=b, coef=log.catch.z),
                     prior(student_t(3, 30, 6), class=Intercept)),
           data=waterQuality)
```

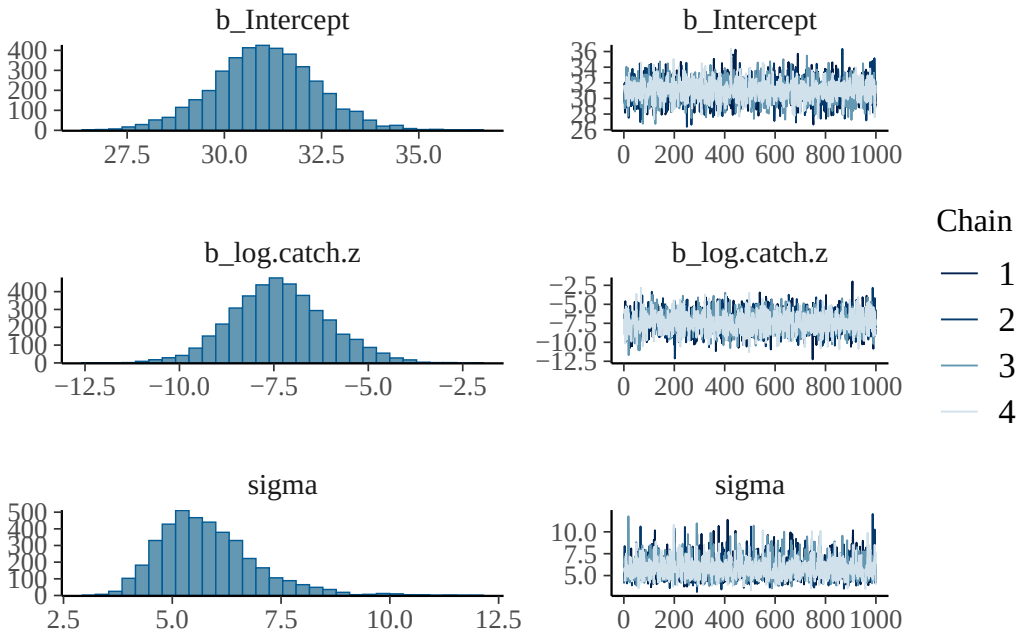
```
summary(fit2, prior=TRUE)
```

```
 Family: gaussian
  Links: mu = identity; sigma = identity
Formula: quality ~ log.catch.z
   Data: waterQuality (Number of observations: 18)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Priors:
b_log.catch.z ~ normal(-5, 2.5)
Intercept ~ student_t(3, 30, 6)
<lower=0> sigma ~ student_t(3, 0, 11.9)
```

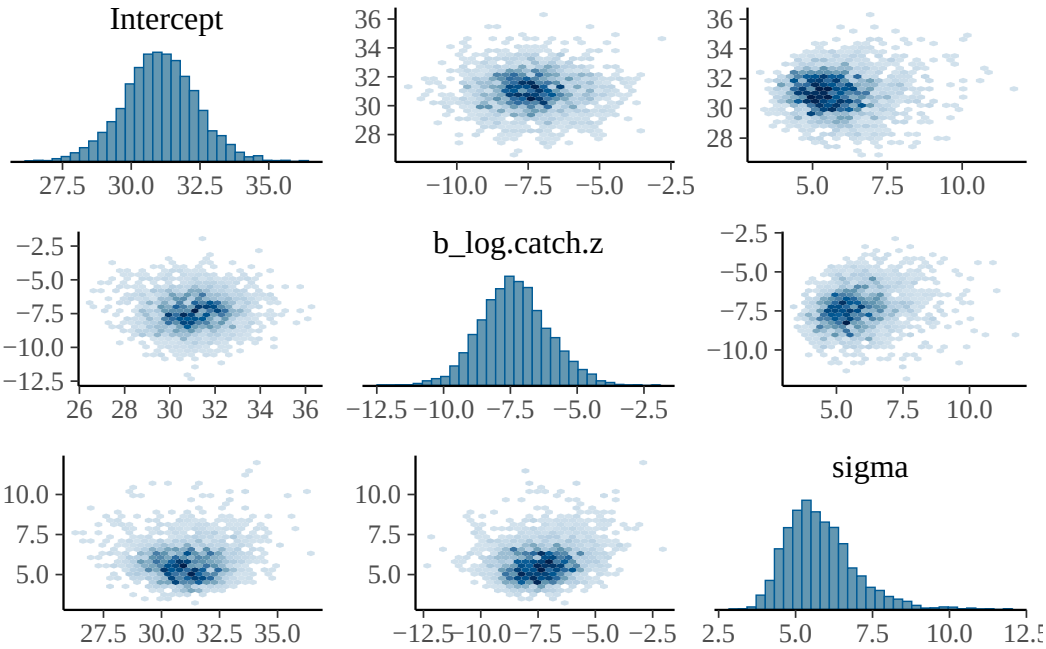
```
Regression Coefficients:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept    31.04      1.34    28.32    33.66 1.00     3179     2605
log.catch.z  -7.34      1.28    -9.76    -4.74 1.00     3015     2667

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     5.81      1.13     4.06     8.48 1.00     2698     2973

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).
```

(4) Check convergence

```
plot(fit2)
```




```
pairs(fit2, variable=c("Intercept", "b_log.catch.z", "sigma"), off_diag_fun="hex")
```



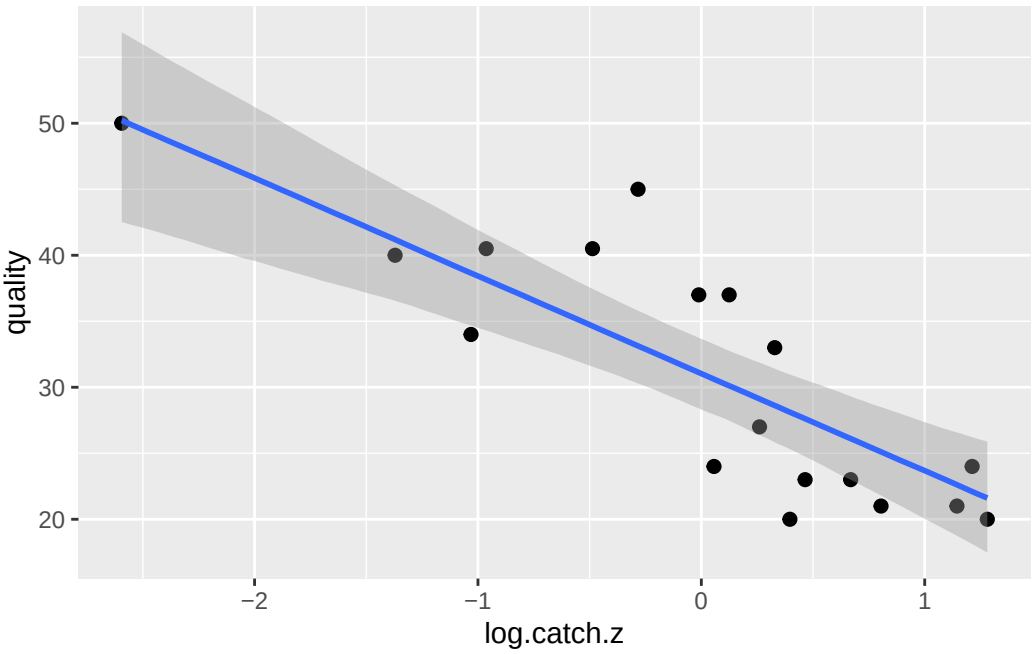
Looks all good!

(5) Check model fit

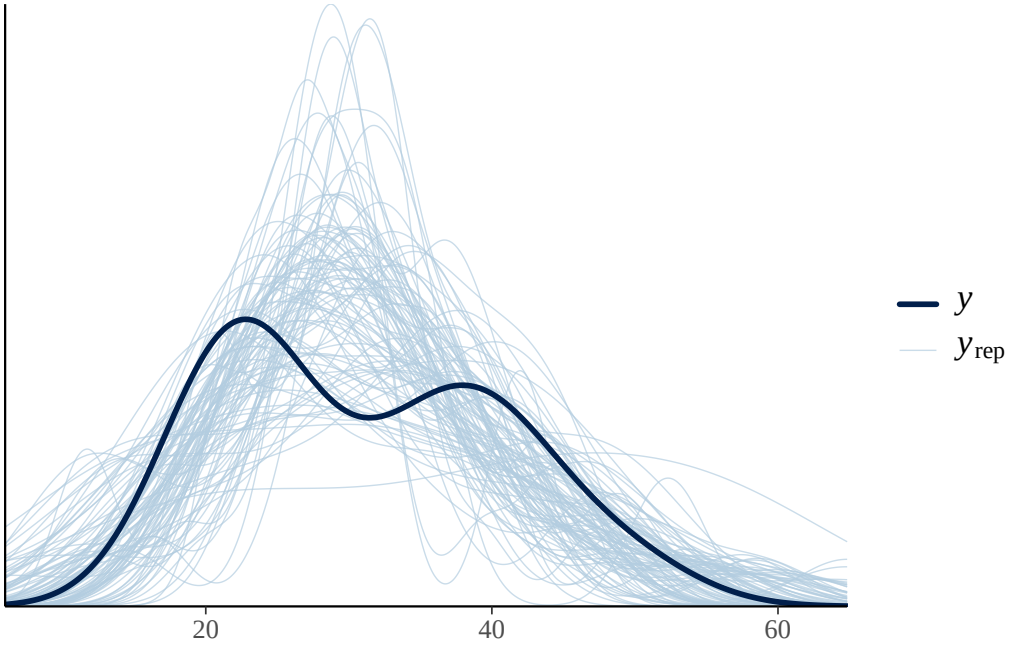
```
bayes_R2(fit2)
```

```
    Estimate Est.Error      Q2.5     Q97.5
R2 0.6346649 0.1031679 0.3670953 0.7598177
```

```
plot(conditional_effects(fit2), points=TRUE)
```

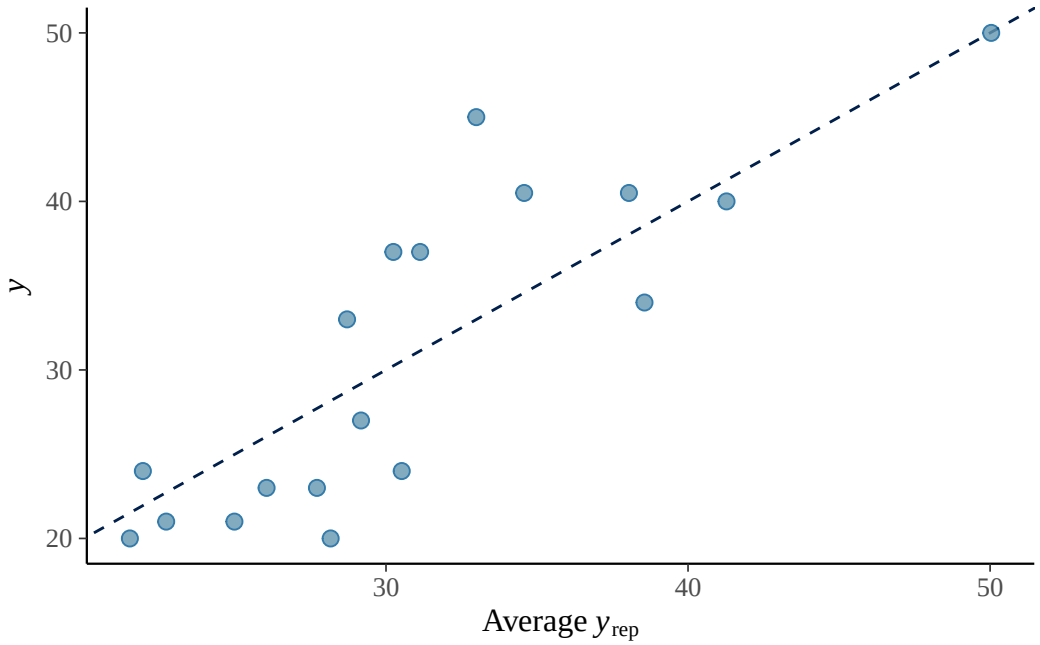



```
pp_check(fit2, ndraws=100)
```




```
pp_check(fit2, type="scatter_avg")
```

```
Using all posterior draws for ppc type 'scatter_avg' by default.
```



We see some deviation and maybe a pattern in the residuals, but since we only have 18 datapoints this could be merely by chance. We don't want to include any other predictors, so we keep the model as is.

(6) Model inference

With the scaled predictor, mean(x) and mean(x)+sd(x) correspond to predictor values of 0 and 1.

We generate fitted distributions of $\mu(0)$ and $\mu(1)$, and a distribution of the difference $\mu(0)-\mu(1)$. Then, compute its mean and 90%-quantile.

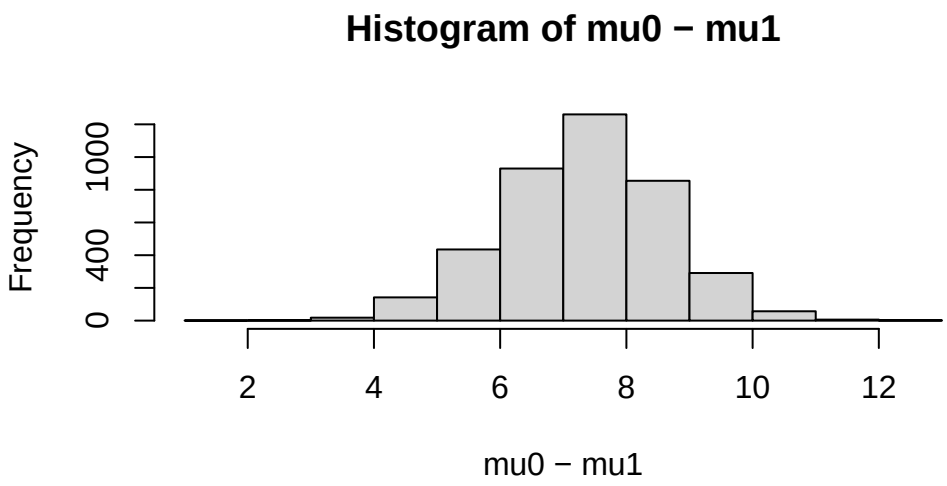
```
fitted(fit2, newdata=data.frame(log.catch.z=0.0))
```

```
     Estimate Est.Error   Q2.5    Q97.5
[1,]  31.0414     1.339 28.317 33.65993
```

```
fitted(fit2, newdata=data.frame(log.catch.z=1.0))
```

```
     Estimate Est.Error     Q2.5    Q97.5
[1,] 23.69945  1.863835 20.03582 27.36779
```

```
mu0 = posterior_epred(fit2, newdata=data.frame(log.catch.z=0.0))
mu1 = posterior_epred(fit2, newdata=data.frame(log.catch.z=1.0))
hist(mu0-mu1)
```




```
mean(mu0-mu1)
```

```
[1] 7.341951
```

```
quantile(mu0-mu1, probs=c(0.05, 0.95))
```

```
      5%      95%
5.162335 9.338708
```

Between mean location `log.catch.z=0.0` in the river and a typical downstream location `log.catch.z=0.0`, the water quality is on average 7.3 [CI 5.1,9.3] better in mean location

Alternatively, we can use the hypothesis function to get the same results.

```
mus = fitted(fit2,
             newdata=data.frame(log.catch.z=c(0,1)),
             summary=F)
mus = as.data.frame(mus)
names(mus)=c("mu0","mu1")
hypothesis(mus, "mu0>mu1")
```

```
Hypothesis Tests for class :
       Hypothesis Estimate Est.Error CI.Lower CI.Upper Evid.Ratio Post.Prob Star
1 (mu0)-(mu1) > 0     7.34      1.28     5.16     9.34        Inf         1    *
---
'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.
'*': For one-sided hypotheses, the posterior probability exceeds 95%;
for two-sided hypotheses, the value tested against lies outside the 95%-CI.
Posterior probabilities of point hypotheses assume equal prior probabilities.
```

```
plot(hypothesis(mus, "mu0>mu1"))
```