

# Practical 2: The brms package

Benjamin Rosenbaum

## Table of contents

<b>Example 1: Linear regression</b>	<b>1</b>
Basic brms functions . . . . .	2
brms specifications & priors . . . . .	9
Model analysis . . . . .	15
Inference . . . . .	16
<b>Example 2: Survival rate</b>	<b>19</b>

We learn about the brms package and how to fit simple regression models.

Focus on: model output, convergence checks.

Also some basic infos on priors and model predictions.

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

## Example 1: Linear regression

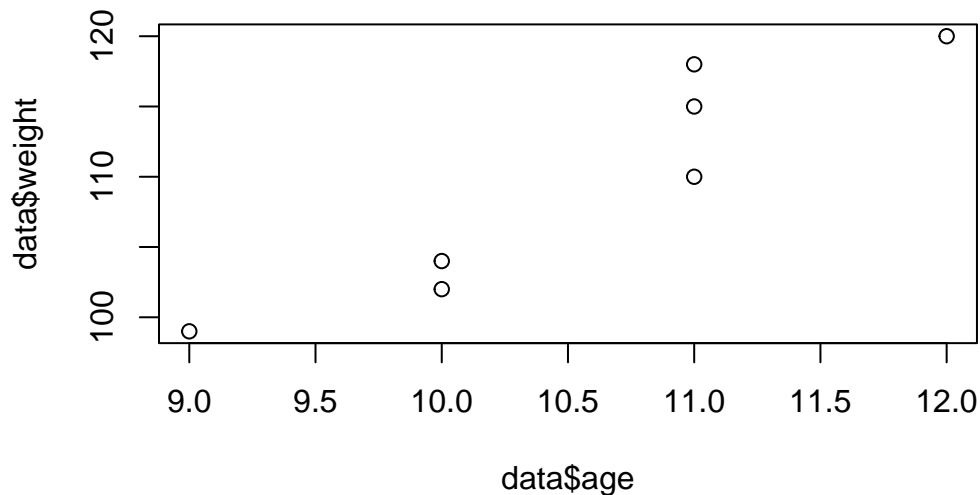
We start with our deer population and the simple weight~age example

**Question:** What's the average growth per year? (Slope in age)

Deterministic part:  $\mu = a + b \cdot \text{age}$

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

```
data = data.frame(weight = c(104, 120, 118, 115, 99, 110, 102),
                  age     = c(10, 12, 11, 11, 9, 11, 10))
plot(data$age, data$weight)
```



## Basic brms functions

Instead of `lm()`, we use the `brm()` function. The formula notation is designed to be identical to `lm`, `glm`, `lme4` (with few exceptions)

```
fit1 = brm(weight ~ age, data=data)
```

Looking at the summary table, we get a lot of infos:

If not specified otherwise, brms uses a normal distribution for the residuals:  
`family=gaussian()`.

brms by default uses 4 chains, each with 1000 warmup and 1000 sampling iterations.  
 The first thing you should look at are not parameter estimates, but Rhat and ESS.  
 These indicate if the MCMC converged and the posterior distribution is properly sampled.  
 Check if  $R_{hat} < 1.01$  and compare ESS to total number of draws.

```
summary(fit1)
```

```
Family: gaussian
Links: mu = identity; sigma = identity
Formula: weight ~ age
```

Data: data (Number of observations: 7)  
 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	25.56	21.23	-17.04	68.48	1.00	2328	1800
age	7.96	1.99	3.87	11.97	1.00	2336	1834

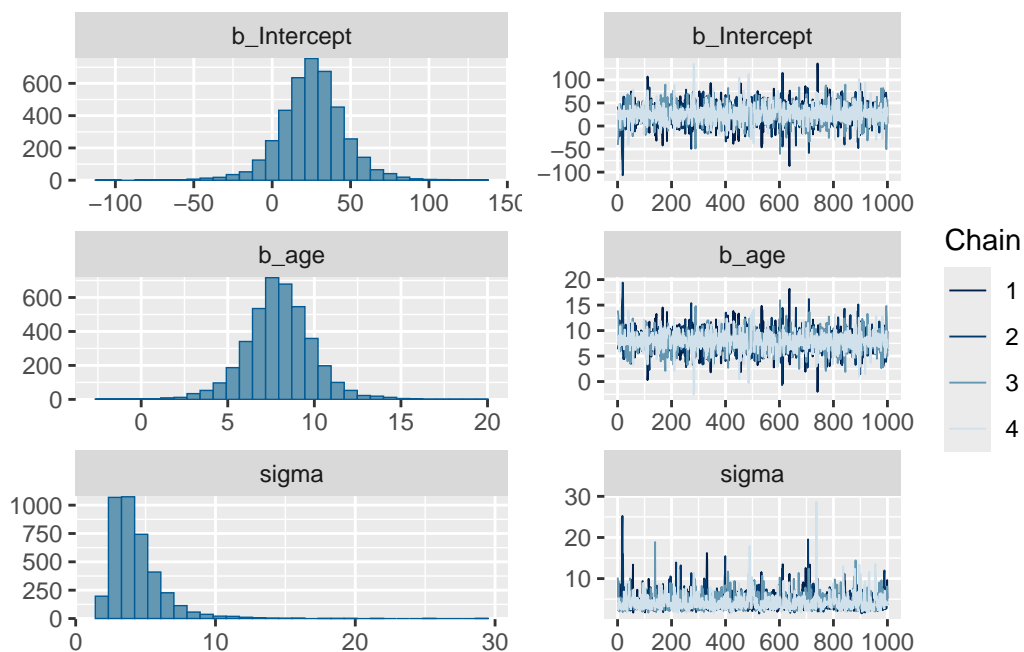
#### Further Distributional Parameters:

	Estimate	Est.Error	l-95% CI	u-95% CI	Rhat	Bulk_ESS	Tail_ESS
sigma	4.31	1.90	2.18	9.12	1.00	1691	2047

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).

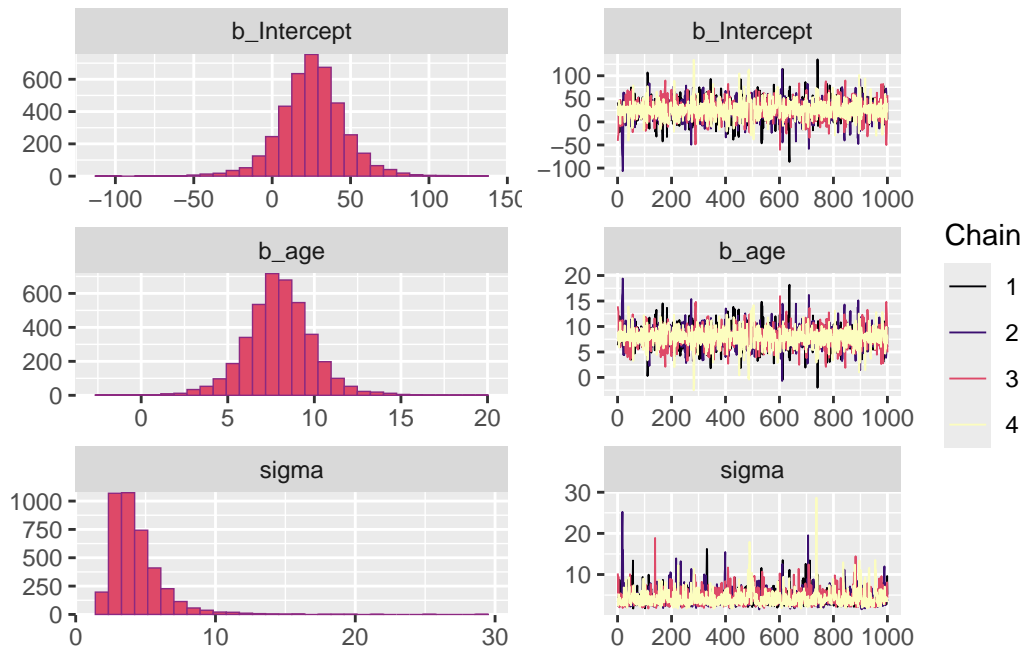
Additionally, you should do a visual inspection of the MCMC. You get a histogram and a traceplot per parameter, which should look like a fuzzy caterpillar

```
plot(fit1)
```



You can change the color palette if you like

```
color_scheme_set("viridisA")
plot(fit1)
```

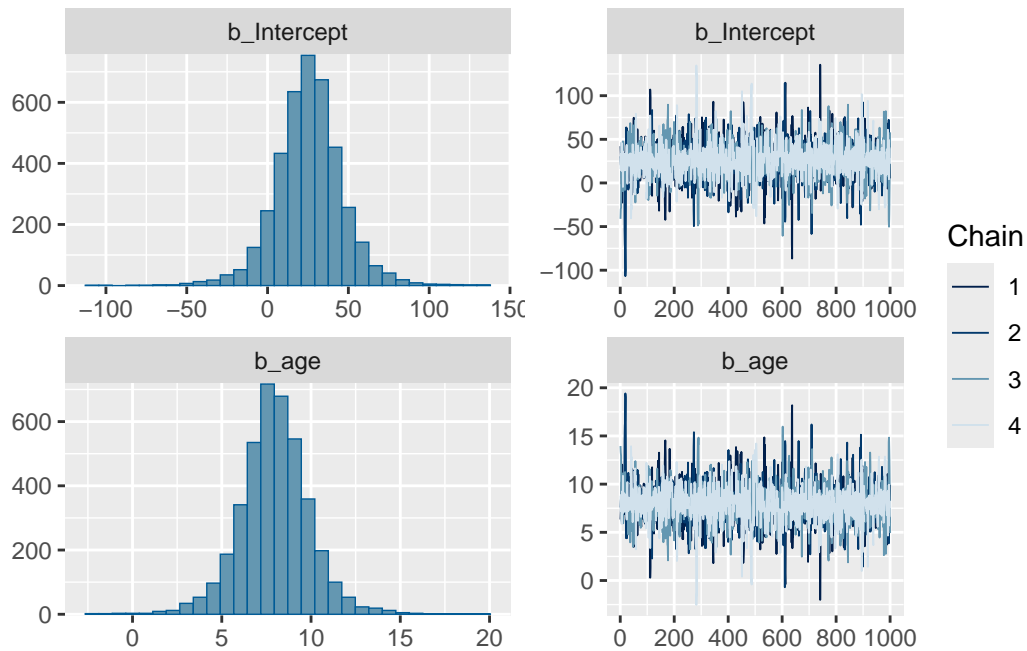


and back to the default palette

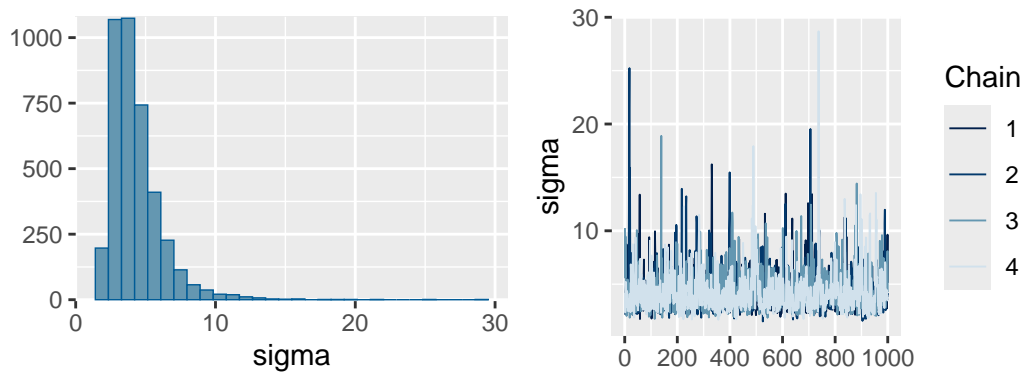
```
color_scheme_set("blue")
```

You can also specify to display just some selected parameters. Parameters of the deterministic model part begin with `b_`, the residual standard deviation is `sigma`

```
plot(fit1, variable=c("b_Intercept", "b_age"))
```

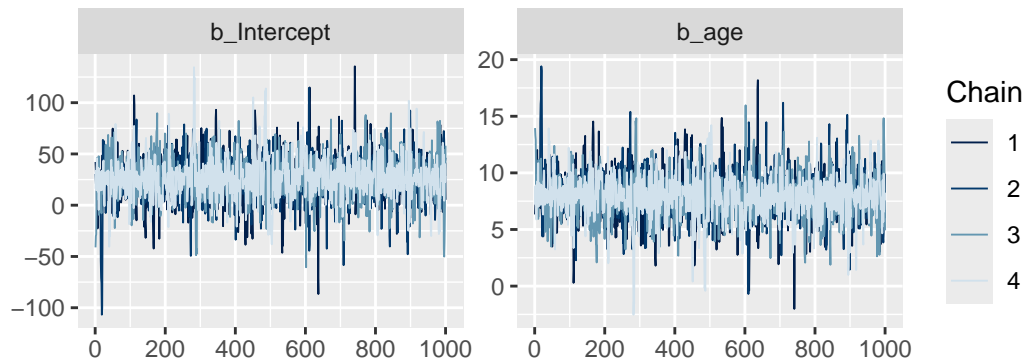


```
plot(fit1, variable=c("sigma"))
```

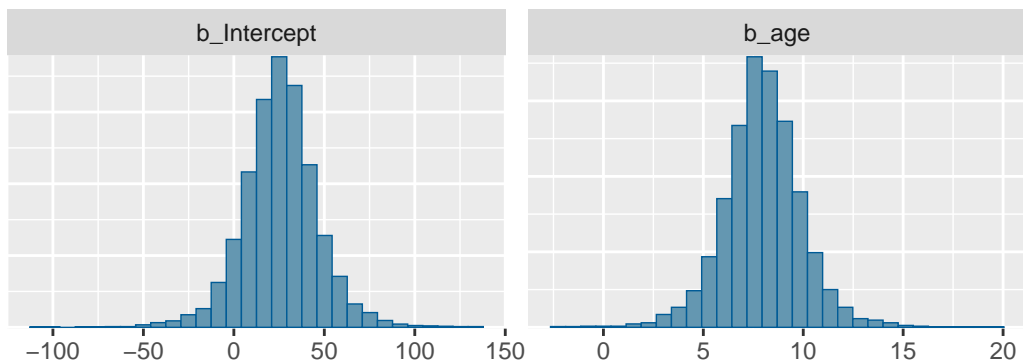


Histograms and traceplots can be plotted individually

```
mcmc_trace(fit1, pars=c("b_Intercept", "b_age"))
```

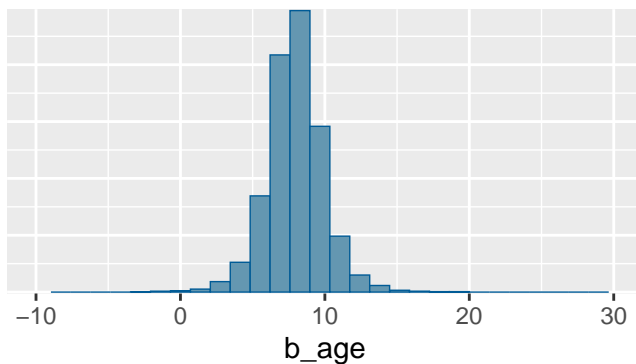


```
mcmc_hist(fit1, pars=c("b_Intercept", "b_age"))
```



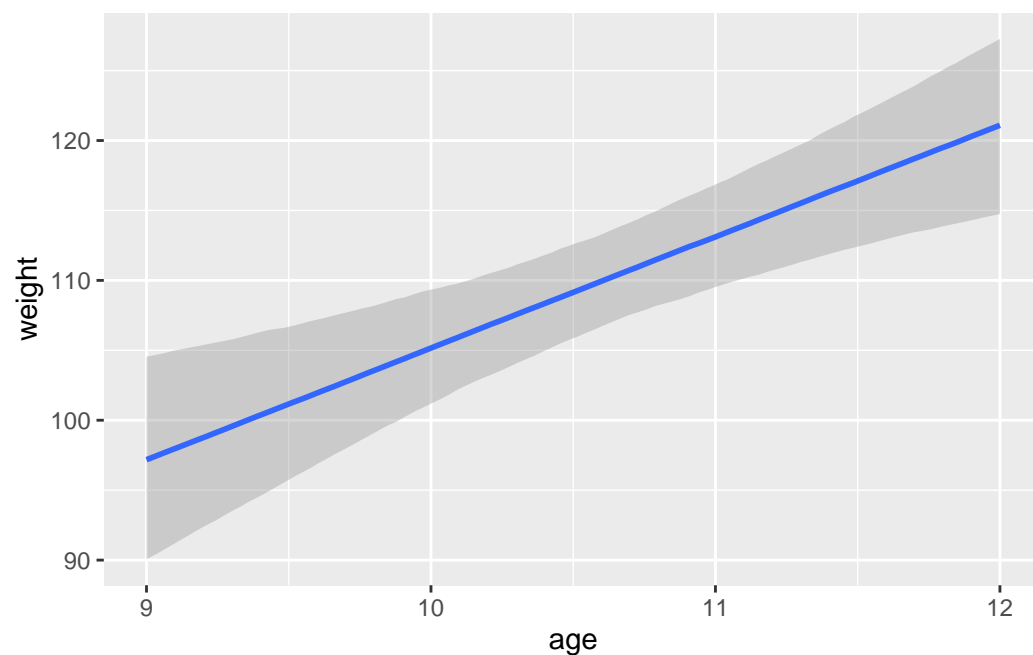
All the brms plots are done with ggplot2, so you can extract & modify them

```
plot1 = mcmc_hist(fit1, pars=c("b_age"))
plot1 + xlim(-10,30)
```

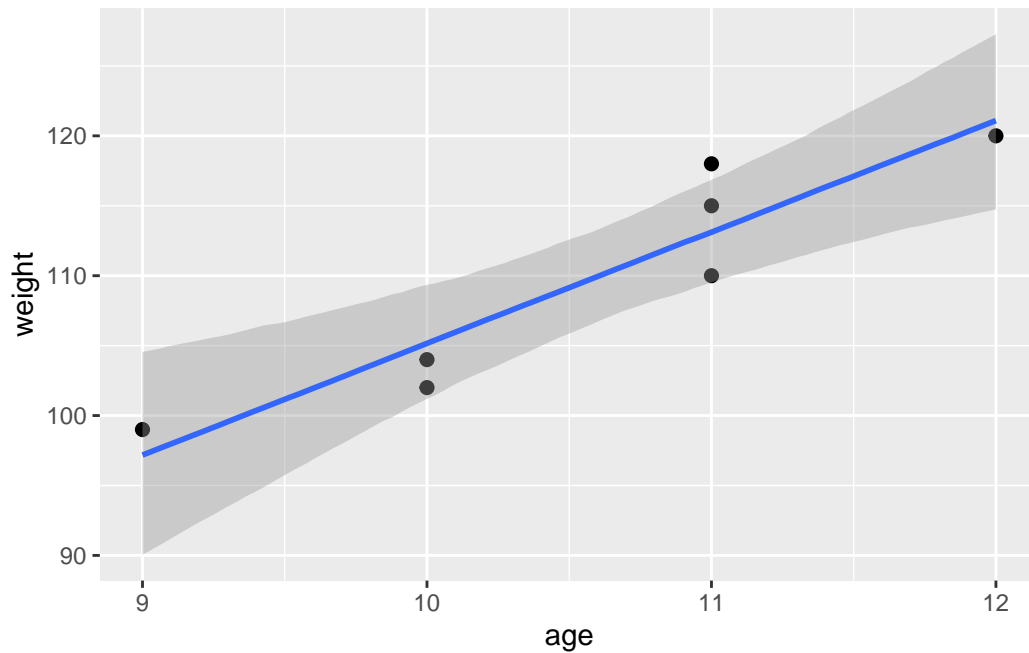


In this simple 1-predictor regression, model predictions are easily plotted vs data. `conditional_effects()` is a powerful function which we will use throughout the course.

```
plot(conditional_effects(fit1))
```

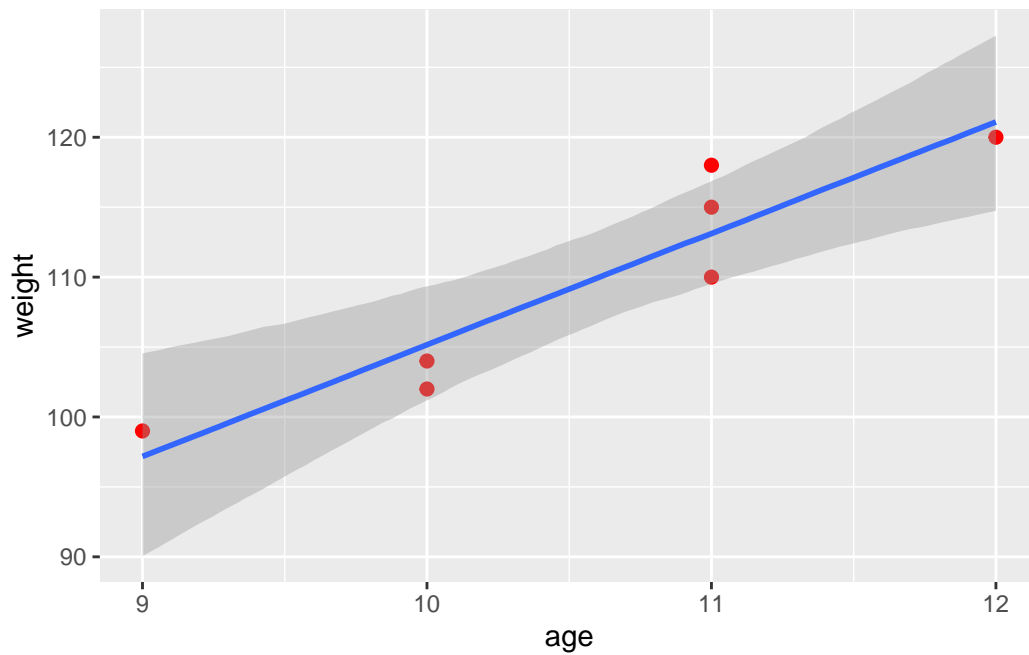


```
plot(conditional_effects(fit1),  
     points=TRUE)
```



Again, this generates a ggplot object which can be modified, with some options in the plot function, or full ggplot options if you save the object

```
plot(conditional_effects(fit1),
     points=TRUE, point_args=list(col="red"))
```





Note that we here only plot the uncertainty of the deterministic model part  $\mu$ , more on that tomorrow. `fitted()` computes predictions of  $\mu$  for each datapoint.

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

	Estimate	Est.Error	Q2.5	Q97.5
[1,]	105.18401	2.084831	101.18471	109.3446
[2,]	121.10918	3.186816	114.74524	127.2614
[3,]	113.14659	1.813679	109.52796	116.8500
[4,]	113.14659	1.813679	109.52796	116.8500
[5,]	97.22142	3.650626	90.06248	104.5488
[6,]	113.14659	1.813679	109.52796	116.8500

The `brms` package does not only offer model fitting via MCMC, it also has a lot of functions for model analysis and is compatible with a lot of other packages (e.g. `emmeans`). We will learn about some of these in the next days.

```
methods(class="brmsfit")
```

[1] add_criterion	add_ic	as_draws_array	as_draws_df
[9] as.array	as.data.frame	as.matrix	as.mcmc
[17] coef	conditional_effects	conditional_smooths	control_params
[25] fixef	formula	getCall	hypothesis
[33] loo_compare	loo_epred	loo_linpred	loo_model_weights
[41] loo_subsample	loo	L00	marginal_effects
[49] nchains	ndraws	neff_ratio	ngrps
[57] nvariables	pairs	parnames	plot
[65] posterior_linpred	posterior_predict	posterior_samples	posterior_smooths
[73] predict	predictive_error	predictive_interval	prepare_predictions
[81] ranef	reloo	residuals	restructure
[89] summary	update	VarCorr	variables

see '?methods' for accessing help and source code

## brms specifications & priors

When we compare the results to frequentist `lm`-model, the slope is pretty close but there's ~0.5 difference in intercepts.

```
fixef(fit1)
```

	Estimate	Est.Error	Q2.5	Q97.5
Intercept	25.558166	21.229873	-17.041362	68.47664
age	7.962584	1.990409	3.872543	11.97261

```
lm(weight ~ age, data=data) |> coef()
```

(Intercept)	age
26.2	7.9

So why are they different? What about priors, did we specify any?

The `brm()` function has **A TON OF** specifications, which we did not specify in the simple `brm(weight~age, data=data)` model, see the help function with `?brm`. So `brms` uses default values.

```
?brm
```

E.g., we can specify the number of chains & iterations manually. Per default, half of the iterations are used for warmup and are discarded from the posterior sample.

```
fit2 = brm(weight ~ age,
           data = data,
           chains = 4,
           iter = 5000
)
```

With a larger number of samples (`iter`), we expect a more accurate approximation of the true posterior. Parameter means usually are quite correct even for low numbers, while outer quantiles (e.g. 90%, 95%) require larger numbers of samples.

```
summary(fit2)
```

```
Family: gaussian
Links: mu = identity; sigma = identity
Formula: weight ~ age
Data: data (Number of observations: 7)
Draws: 4 chains, each with iter = 5000; warmup = 2500; thin = 1;
       total post-warmup draws = 10000

Regression Coefficients:
      Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
```

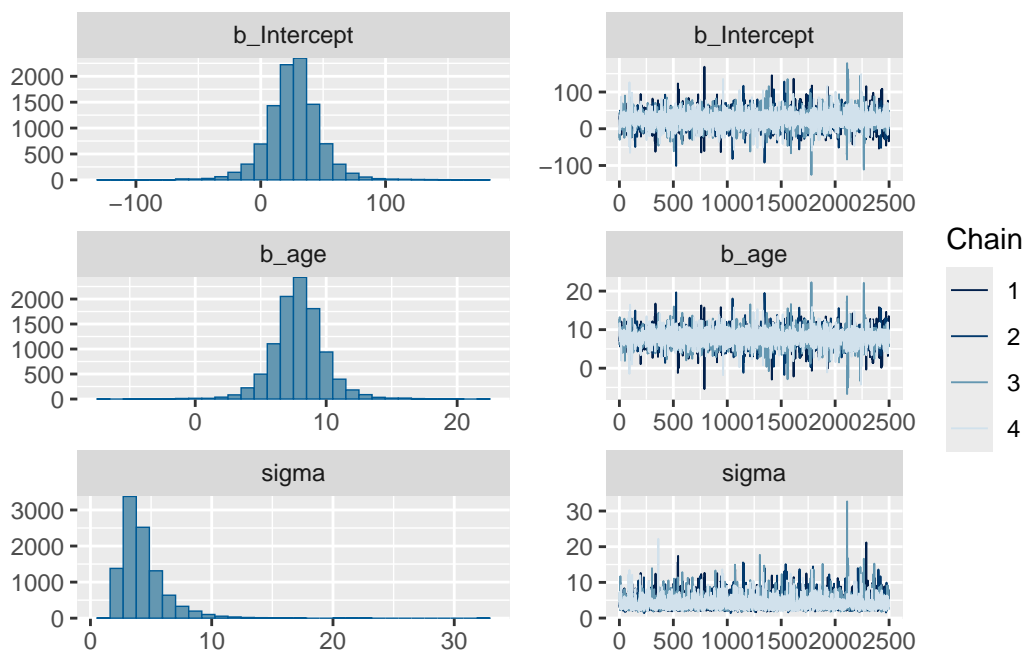
Intercept	26.25	21.61	-17.79	68.66	1.00	5245	4110
age	7.89	2.04	3.94	12.02	1.00	5257	4131

Further Distributional Parameters:

	Estimate	Est.Error	l-95% CI	u-95% CI	Rhat	Bulk_ESS	Tail_ESS
sigma	4.30	1.87	2.13	9.10	1.00	3621	3481

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(fit2)
```



We can check the defaults for any model with `default_prior()`. The model does not have to be fitted, just model formula and data must be specified.

```
default_prior(weight ~ age,
              data = data)
```

	prior	class	coef	group	resp	dpar	nlpar	lb	ub	source
	(flat)	b								default
	(flat)	b	age							(vectorized)
student_t(3, 110, 11.9)		Intercept								default
student_t(3, 0, 11.9)		sigma						0		default

Alternatively, you can display the priors of any fitted model. Since we had not specified any priors, both outputs are the same here.

```
prior_summary(fit2)
```

	prior	class	coef	group	resp	dpar	nlpar	lb	ub	source
	(flat)	b								default
	(flat)	b	age							(vectorized)
student_t(3, 110, 11.9)		Intercept								default
student_t(3, 0, 11.9)		sigma						0		default

This table can be a bit confusing, but look at the column **class**: **b** is for effects / slopes. The first line tells you if there is a prior used for ALL effects, which is not the case (**prior=flat**). Second line is the prior for a specific coefficient (**coef=age**), there's also no prior specified.

But brms chooses a prior for **Intercept** and for the residual sdev **sigma**. These are automatically generated from the mean and the spread of the response. Note that internally, the brms machine uses mean-centered predictors. The **Intercept** parameter (and its prior) are based on mean-centered variables. What's displayed in the model summary is actually **b\_Intercept** which is the intercept parameter transformed to the original, non mean-centered scale.

A short form is presented in the summary for **prior=TRUE**

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

```
Family: gaussian
Links: mu = identity; sigma = identity
Formula: weight ~ age
Data: data (Number of observations: 7)
Draws: 4 chains, each with iter = 5000; warmup = 2500; thin = 1;
       total post-warmup draws = 10000
```

```
Priors:
Intercept ~ student_t(3, 110, 11.9)
<lower=0> sigma ~ student_t(3, 0, 11.9)
```

```
Regression Coefficients:
      Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept    26.25     21.61  -17.79   68.66 1.00    5245    4110
age           7.89      2.04   3.94   12.02 1.00    5257    4131
```

```
Further Distributional Parameters:
```

	Estimate	Est.Error	l-95% CI	u-95% CI	Rhat	Bulk_ESS	Tail_ESS
sigma	4.30	1.87	2.13	9.10	1.00	3621	3481

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).

Unless necessary, I would leave the brms defaults for `Intercept` & `sigma`. However, you should choose a prior for the slope, which currently has none.

This would set a prior for all slopes (if you have >1 predictors)

```
my_priors = prior(normal(5,1), class=b)
```

For setting a prior for a specific predictor, you specify it in `coef`. Since this model only has 1 predictor, both formulations are the same.

```
my_priors = prior(normal(5,1), class=b, coef=age)
```

```
fit3 = brm(weight ~ age,
  prior = my_priors,
  data = data,
  chains = 4,
  iter = 5000
)
```

```
summary(fit3, prior=TRUE)
```

```
Family: gaussian
Links: mu = identity; sigma = identity
Formula: weight ~ age
Data: data (Number of observations: 7)
Draws: 4 chains, each with iter = 5000; warmup = 2500; thin = 1;
       total post-warmup draws = 10000
```

```
Priors:
b_age ~ normal(5, 1)
Intercept ~ student_t(3, 110, 11.9)
<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	49.08	10.00	30.68	69.54	1.00	6562	6790
age	5.74	0.93	3.84	7.48	1.00	6627	6763

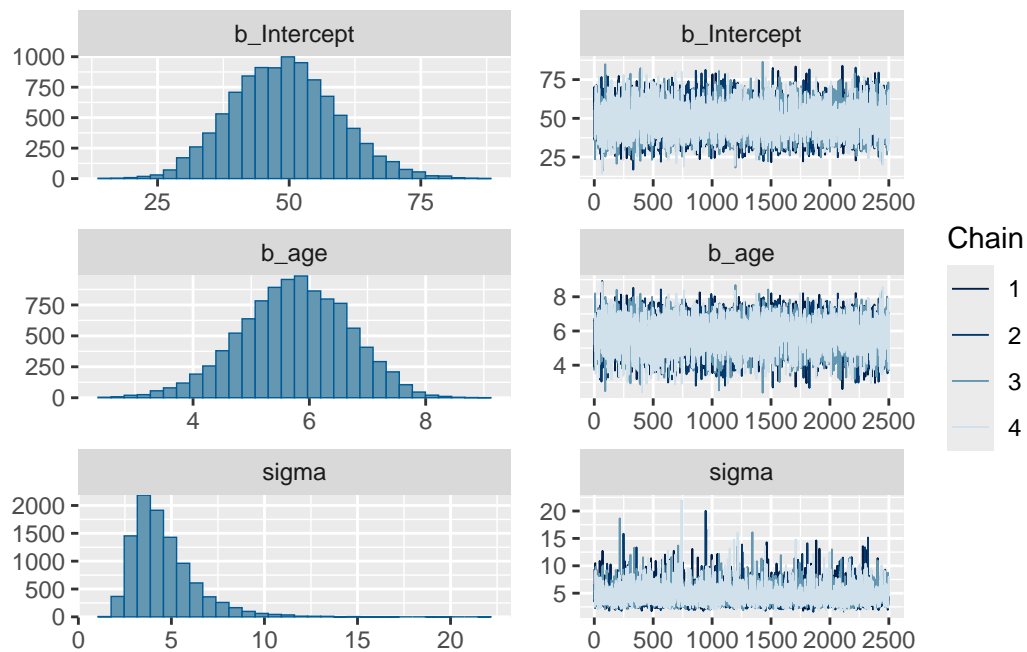
Further Distributional Parameters:

	Estimate	Est.Error	l-95% CI	u-95% CI	Rhat	Bulk_ESS	Tail_ESS
sigma	4.59	1.75	2.36	9.06	1.00	4814	5306

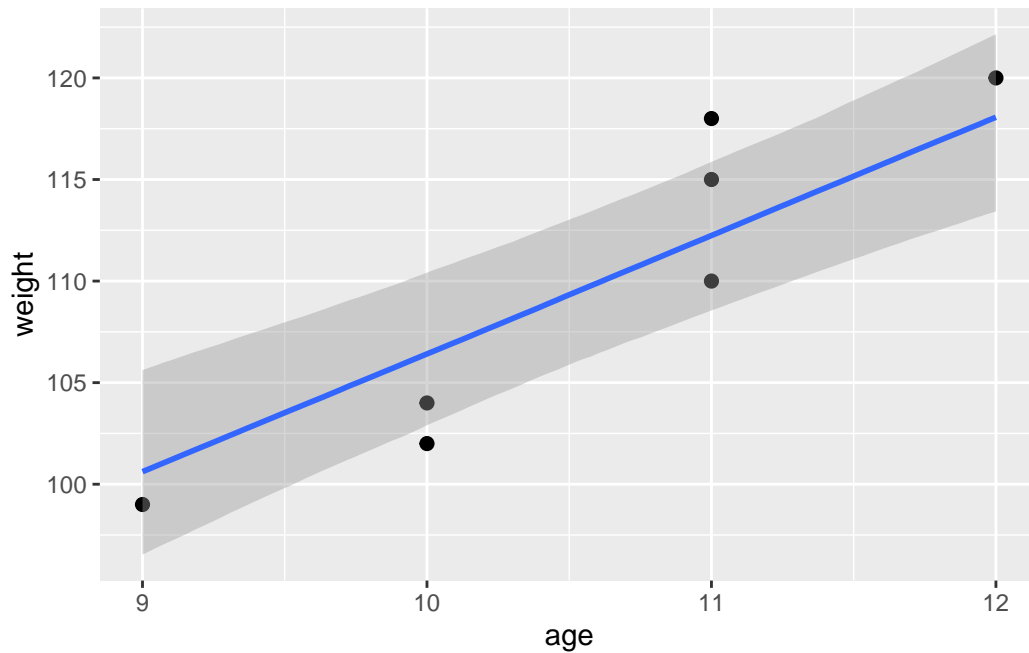
Draws were sampled using sampling(NUTS). For each parameter, Bulk\_ESS and Tail\_ESS are effective sample size measures, and Rhat is the potential scale reduction factor on split chains (at convergence, Rhat = 1).

Here we only have a small dataset and an informative prior (mean=5) changes the posterior estimate of the slope.

```
plot(fit3)
```



```
plot(conditional_effects(fit3),
     points=TRUE)
```



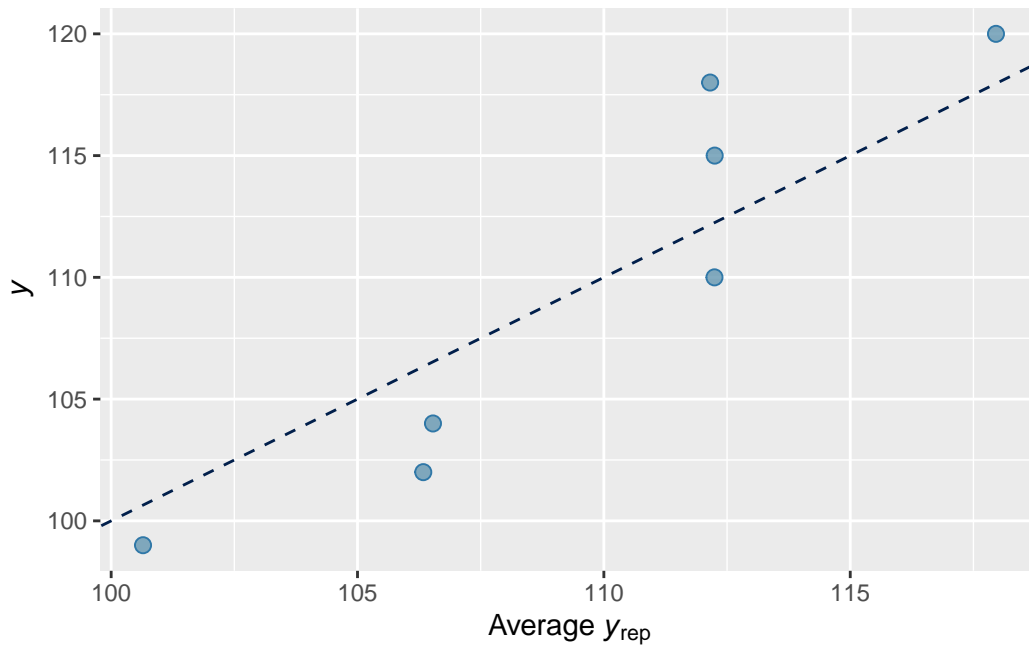
## Model analysis

Only after we checked MCMC convergence, we can go to the next step: Model evaluation / model checking. How well does our model describe the data?

A classical visual tool is observed vs predicted, which also works if you have multiple predictors.

```
pp_check(fit3, "scatter_avg")
```

Using all posterior draws for ppc type 'scatter\_avg' by default.



`bayes_R2` is the amount of explained variation. Its computation is a bit different from the classical frequentist `R2`, but conceptually it means the same.

```
bayes_R2(fit3)
```

	Estimate	Est.Error	Q2.5	Q97.5
R2	0.680979	0.129089	0.3647646	0.8557636

More on that tomorrow, e.g. checking model assumptions.

## Inference

OK, so we know that (a) MCMC converged and (b) model describes the data well. Only now can we make inference, i.e. quantitative statements about research questions. The summary already tells us mean and 95% confidence intervals for the slope (growth per year of age).

Different Credible intervals can be chosen in the summary, e.g. 90%-CI. 90% of posterior samples for slope were in this interval, we are 90% sure that the slope is in this interval.

```
summary(fit3, prob=0.90)
```



```

Family: gaussian
Links: mu = identity; sigma = identity
Formula: weight ~ age
Data: data (Number of observations: 7)
Draws: 4 chains, each with iter = 5000; warmup = 2500; thin = 1;
       total post-warmup draws = 10000

```

Regression Coefficients:

	Estimate	Est.Error	l-90% CI	u-90% CI	Rhat	Bulk_ESS	Tail_ESS
Intercept	49.08	10.00	33.04	65.99	1.00	6562	6790
age	5.74	0.93	4.16	7.24	1.00	6627	6763

Further Distributional Parameters:

	Estimate	Est.Error	l-90% CI	u-90% CI	Rhat	Bulk_ESS	Tail_ESS
sigma	4.59	1.75	2.56	7.96	1.00	4814	5306

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).

Or you can extract specific quantiles of parameter estimates. `fixef` means “fixed effects” here.

```
fixef(fit3)
```

	Estimate	Est.Error	Q2.5	Q97.5
Intercept	49.079222	9.9968130	30.675897	69.541695
age	5.741318	0.9277682	3.842057	7.475878

```
fixef(fit3, probs=c(0.25, 0.5, 0.75))
```

	Estimate	Est.Error	Q25	Q50	Q75
Intercept	49.079222	9.9968130	42.043930	48.890111	55.640038
age	5.741318	0.9277682	5.127492	5.761661	6.395498

In a frequentist analysis you would want to know if the effect of age is “significant”: p-values quantify the probability of observing such a pattern the data if the null hypothesis ( $b_{age}=0$ ) was true (p small  $\rightarrow$  reject  $H_0$ ).

Here, we can just calculate the probability that the slope is positive,  $P(b_{age} > 0)$ , with the `hypothesis` function. The column `Post.Prob` is the value of interest. It’s =1 because all samples of slope were positive

```
hypothesis(fit3, "age>0")
```

Hypothesis Tests for class b:

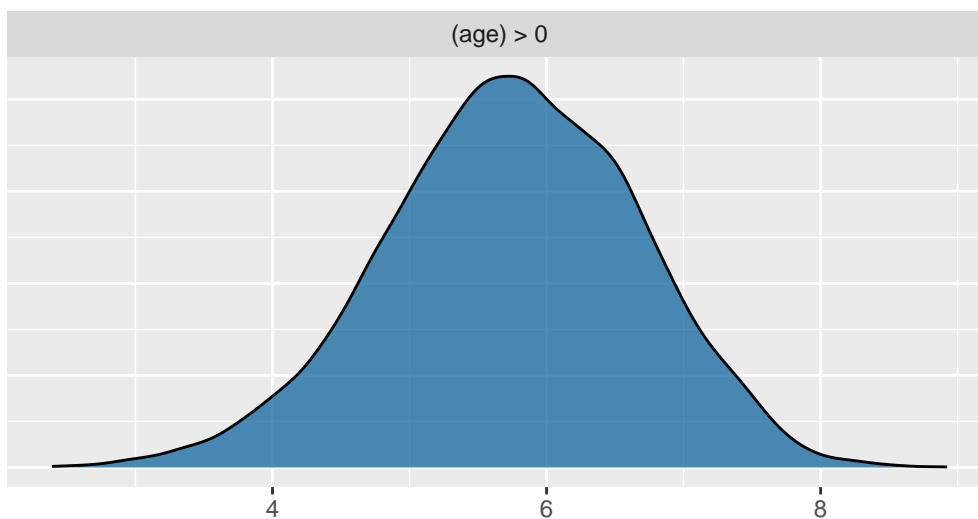
	Hypothesis	Estimate	Est.Error	CI.Lower	CI.Upper	Evid.Ratio	Post.Prob	Star
1	(age) > 0	5.74	0.93	4.16	7.24	Inf	1	*

---

'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(fit3, "age>0"))
```



You can test all kinds of hypotheses for the parameters! If we were interested in the question if growth per year is bigger than 4,  $P(b_{age} > 4)$ , just put it in the hypothesis. The function is quite powerful and can handle all kinds of transformations of parameters.

```
hypothesis(fit3, "age>4")
```

Hypothesis Tests for class b:

	Hypothesis	Estimate	Est.Error	CI.Lower	CI.Upper	Evid.Ratio	Post.Prob	Star
1	(age)-(4) > 0	1.74	0.93	0.16	3.24	27.01	0.96	*

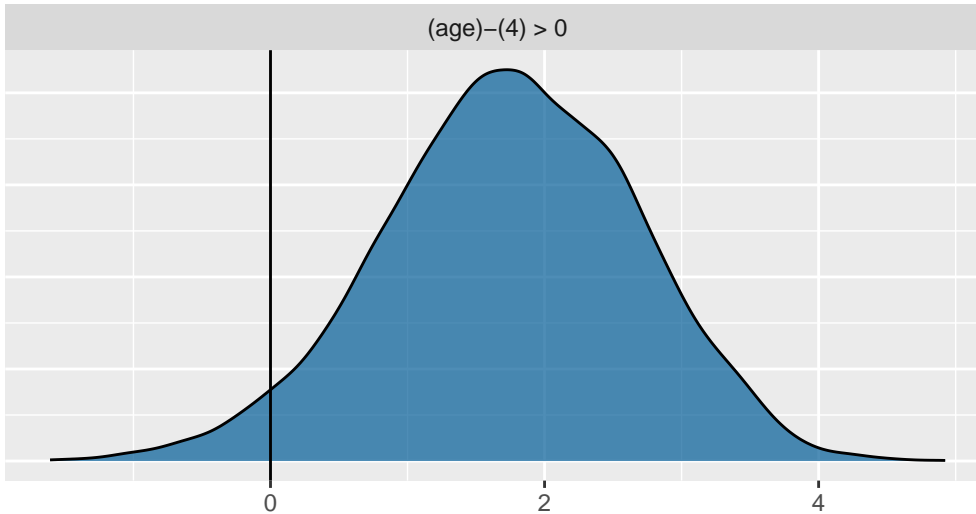
---

'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.

It's transformed in the equivalent formulation  $\text{age}-4 > 0$ , and this is the probability distribution which is actually plotted.

```
plot1 = plot(hypothesis(fit3, "age>4"), plot=FALSE)
plot1[[1]] + geom_vline(xintercept=0)
```



The posterior probability is 0.96, which is also the integral (area under the curve) right of zero ( $\text{age}-4 > 0$ )

## Example 2: Survival rate

Population counts from different habitats before and after winter.

**Question:** Is the average survival rate bigger than  $2/3$  ?

Deterministic part:  $\mu = \theta, \quad \theta \in [0, 1]$

Stochastic part:  $\text{survived}_i \sim \text{Binomial}(\text{total}_i, \mu)$

Check the default prior!

Choose a meaningful prior for the **Intercept** parameter, use  $\text{lb}=0$ ,  $\text{ub}=1$ .

Fit the model & verify convergence.

Re-run the analysis for different priors.

```
data = data.frame(total = c(22,22,29,21,25,30,24,23,25,28),
                   survived = c(19,14,23,19,20,18,15,16,18,15))

default_prior(survived | trials(total) ~ 1,
```

```
family = binomial(link="identity"),
data = data)
```

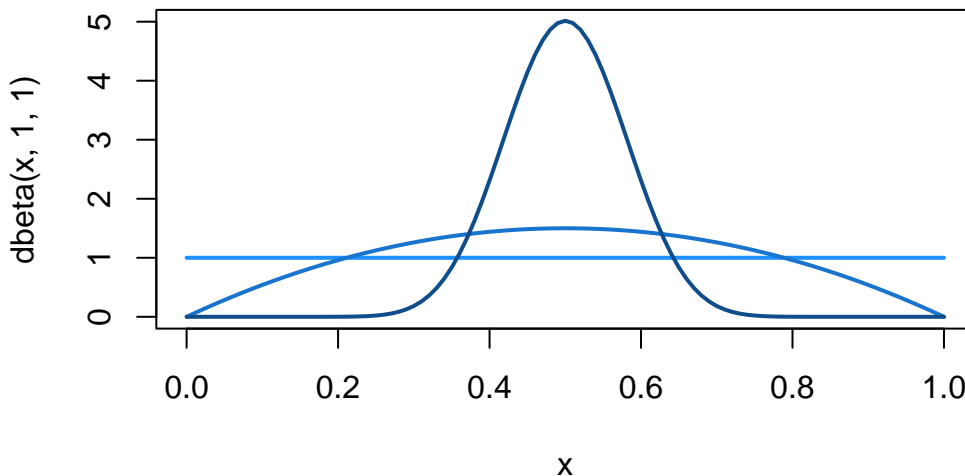
```
Intercept ~ student_t(3, 18, 3)
```

This brms default prior choice does not make any sense (mean=18), since it is chosen from the mean of the response **survived** (I guess??), while the parameter is actually a rate / probability  $0 < \theta < 1$ . Here brms messed up because we have overwritten the default link="logit" of the binomial distribution (again, I guess??). In most cases the default prior is fine, but better check it for generalized linear models (see also part 5 on GLMs).

We use 3 different beta distribution priors (defined on interval  $[0,1]$ ). Two shape parameters  $s_1, s_2$  describe concentration to its mean  $s_1/(s_1 + s_2)$

- (1)  $\text{beta}(1,1)$  = uniform distribution
- (2)  $\text{beta}(2,2)$  = weak prior, mean=0.5
- (3)  $\text{beta}(20,20)$  = informative prior, mean=0.5

```
curve(dbeta(x,1,1), ylim=c(0,5), col="dodgerblue", lwd=2)
curve(dbeta(x,2,2), add=TRUE, col="dodgerblue3", lwd=2)
curve(dbeta(x,20,20), add=TRUE, , col="dodgerblue4", lwd=2)
```



```
fit4 = brm(survived | trials(total) ~ 1,
family = binomial(link="identity"),
prior = prior(uniform(0,1), class=Intercept, lb=0, ub=1),
data = data)
```

```
fit5 = brm(survived | trials(total) ~ 1,
  family = binomial(link="identity"),
  prior = prior(beta(2,2), class=Intercept, lb=0, ub=1),
  data = data)
```

```
fit6 = brm(survived | trials(total) ~ 1,
  family = binomial(link="identity"),
  prior = prior(beta(20,20), class=Intercept, lb=0, ub=1),
  data = data)
```

```
summary(fit4)
```

```
Family: binomial
Links: mu = identity
Formula: survived | trials(total) ~ 1
Data: data (Number of observations: 10)
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	0.71	0.03	0.65	0.77	1.00	1518	1824

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, the prior has only little effect on the outcome:

```
fixef(fit4)
```

	Estimate	Est.Error	Q2.5	Q97.5
Intercept	0.708836	0.02882774	0.6515685	0.7655459

```
fixef(fit5)
```

	Estimate	Est.Error	Q2.5	Q97.5
Intercept	0.7077716	0.02844394	0.6496433	0.7619462

```
fixef(fit6)
```

	Estimate	Est.Error	Q2.5	Q97.5
Intercept	0.6818631	0.02743494	0.6242358	0.7356291

We use the weakly informative prior model to test if survival rate is  $> 2/3$

```
hypothesis(fit5, "Intercept>2/3")
```

Hypothesis Tests for class b:

	Hypothesis	Estimate	Est.Error	CI.Lower	CI.Upper	Evid.Ratio	Post.Prob	Star
1	(Intercept)-(2/3) > 0	0.04	0.03	-0.01	0.09	11.31	0.92	

---

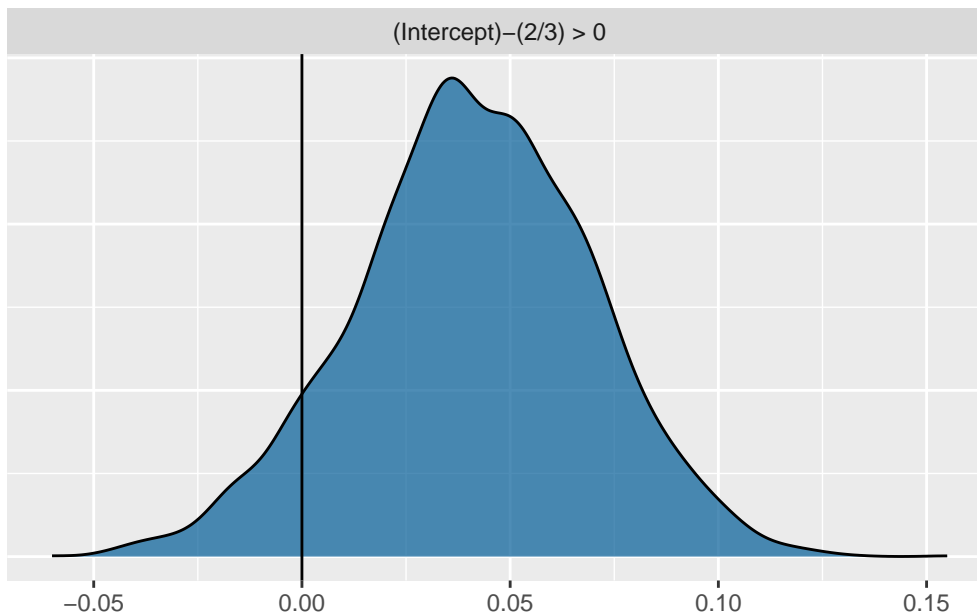
'CI': 90%-CI for one-sided and 95%-CI for two-sided hypotheses.

'\*': 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.

```
plot1 = plot(hypothesis(fit5, "Intercept>2/3"), plot=FALSE)  
plot1[[1]] + geom_vline(xintercept=0)
```



We estimate a mean survival rate of 0.708 with a 95% credible interval [0.649,0.762]. Also, posterior probability  $P(\theta > 2/3) = 0.92$ , which means we are only 92% certain that average survival rate is larger than  $2/3$ .