Homework 8

Emily Maloney

February 25, 2019

Chapter 8 Homework

Easy Problems

8E1

The requirement of the simple Metropolis algorithm is:
(3) The proposal distribution must be symmetric.

8E2

Gibbs sampling achieves greater efficiency than the Metropolis algorithm because it uses adaptive proposals. In other words, it intelligently updates the proposed posterior distribution by using conjugate pairs of elements from the prior distributions and likelihoods. These pairs each have a solutions for the structure of the posterior distribution that guide the algorithm as it moves through the posterior distributions.

8E3

The Hamiltonian Monte Carlo cannot handle discrete parameters, because discrete parameters would require distinct steps rather than moving continuously according to rates.

8E4

The actual number of samples is the number of complete cases in your data set. The effective number of samples is the number of independent samples, meaning that they are not autocorrelated with other samples in the Markov chain.

8E5

Rhat should approach 1.00 when a chain is sampling the posterior distribution correctly.

8E6

A healthy trace plot of a chain has two elements. First, they exhibit stationarity, meaning that the traces fall around a central value inside of the posterior distribution. Second, they show good mixing, in that the start value of a trace is not correlated with the one that came previously, providing the zig-zag shape seen in the sketch below on the left.

An unhealthy trace plot can be identified by the wandering nature of the trace. It is not centered around the mean value but instead jumps from high to low and back to high values. Additionally, it seems as though the successive samples are correlated with previous ones, such that the trace gets "stuck" and doesn't show random mixing. Both of these elements are illustrated in the sketch on the right side of the image below.

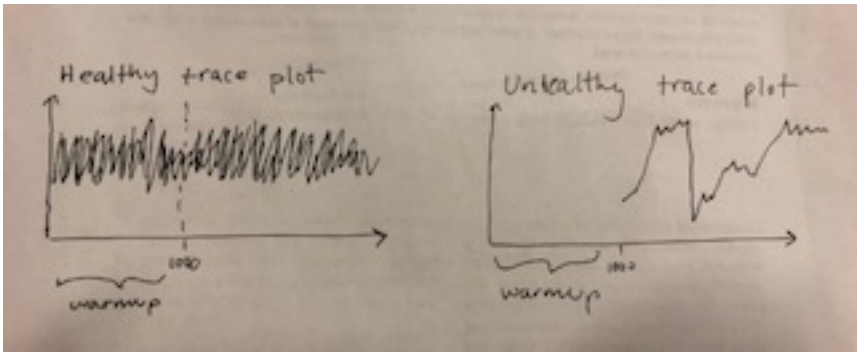


Figure 1: Trace Plots

Medium Problems

8M1

```
library(brms)
library(rethinking)
library(tidyverse)
library(knitr)

data("rugged")
d <- rugged
dd <- d %>% mutate(log_gdp = log(rgdppc_2000)) %>% filter(!is.na(log_gdp))

suppressMessages(h8.0 <- brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
                prior(normal(0, 10), class = b),
                prior(cauchy(0, 2), class = sigma))))
```

```
##
## SAMPLING FOR MODEL 'ae400a3fc447dbdc9dc2cf4b2f0adb9b' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.001 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
```

```
## Chain 1:  Elapsed Time: 0.085 seconds (Warm-up)
## Chain 1:                0.079 seconds (Sampling)
## Chain 1:                0.164 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'ae400a3fc447dbdc9dc2cf4b2f0adb9b' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.078 seconds (Warm-up)
## Chain 2:                0.081 seconds (Sampling)
## Chain 2:                0.159 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'ae400a3fc447dbdc9dc2cf4b2f0adb9b' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.08 seconds (Warm-up)
## Chain 3:                0.075 seconds (Sampling)
## Chain 3:                0.155 seconds (Total)
## Chain 3:
```

```
##
## SAMPLING FOR MODEL 'ae400a3fc447dbdc9dc2cf4b2f0adb9b' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.084 seconds (Warm-up)
## Chain 4:                0.081 seconds (Sampling)
## Chain 4:                0.165 seconds (Total)
## Chain 4:
```

```
h8.1 <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
                prior(normal(0, 10), class = b),
                prior(uniform(0, 10), class = sigma)))
```

```
##
## SAMPLING FOR MODEL '3c58df1a6d7319d9a8db62de7a83d284' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.083 seconds (Warm-up)
```

```
## Chain 1:                0.074 seconds (Sampling)
## Chain 1:                0.157 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL '3c58df1a6d7319d9a8db62de7a83d284' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.08 seconds (Warm-up)
## Chain 2:                0.078 seconds (Sampling)
## Chain 2:                0.158 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL '3c58df1a6d7319d9a8db62de7a83d284' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.083 seconds (Warm-up)
## Chain 3:                0.078 seconds (Sampling)
## Chain 3:                0.161 seconds (Total)
## Chain 3:
##
```

```
## SAMPLING FOR MODEL '3c58df1a6d7319d9a8db62de7a83d284' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.085 seconds (Warm-up)
## Chain 4:                0.092 seconds (Sampling)
## Chain 4:                0.177 seconds (Total)
## Chain 4:
```

```
h8.2 <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
                prior(normal(0, 10), class = b),
                prior(exponential(1), class = sigma)))
```

```
##
## SAMPLING FOR MODEL 'c6c3453012689a25e9ad9865ececf1ca' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.001 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.081 seconds (Warm-up)
## Chain 1:                0.082 seconds (Sampling)
```

```
## Chain 1:                0.163 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'c6c3453012689a25e9ad9865ececf1ca' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.076 seconds (Warm-up)
## Chain 2:                0.074 seconds (Sampling)
## Chain 2:                0.15 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'c6c3453012689a25e9ad9865ececf1ca' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.081 seconds (Warm-up)
## Chain 3:                0.088 seconds (Sampling)
## Chain 3:                0.169 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'c6c3453012689a25e9ad9865ececf1ca' NOW (CHAIN 4).
```

```
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.078 seconds (Warm-up)
## Chain 4:                0.071 seconds (Sampling)
## Chain 4:                0.149 seconds (Total)
## Chain 4:
```

```
print(h8.0)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.22      0.14     8.95     9.50       2614 1.00
## rugged               -0.20      0.08    -0.36    -0.05       2481 1.00
## cont_africa          -1.94      0.23    -2.39    -1.49       2305 1.00
## rugged:cont_africa    0.39      0.13     0.13     0.64       2151 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.86     1.06       4440 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.1)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
```

```
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.22      0.14     8.94     9.50       2780 1.00
## rugged               -0.20      0.08    -0.36    -0.05       2521 1.00
## cont_africa          -1.95      0.23    -2.40    -1.50       2272 1.00
## rugged:cont_africa    0.39      0.13     0.15     0.65       2301 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.85     1.06       4080 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.2)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.22      0.14     8.95     9.50       2570 1.00
## rugged               -0.20      0.08    -0.36    -0.06       2296 1.00
## cont_africa          -1.95      0.23    -2.40    -1.51       2274 1.00
## rugged:cont_africa    0.39      0.13     0.14     0.66       2132 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.85     1.05       4226 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

In this case, the different priors do not have any detectible influence on the posterior. For all three models, h8.0 (cauchy prior), h8.1 (uniform prior), and h8.2 (exponential prior) have almost exactly the same estimates for the parameters. The differences between parameters are all around +/- 0.1, which does not change the interpretation of the model. Similarly, there does not seem to be much of a difference between the number of effective samples in each model. This is could either be because the priors are weak or because there is enough data to overwhelm the priors.

8M2

```
h8.3 <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
```

```
                prior(normal(0, 10), class = b),
                prior(cauchy(0, 1), class = sigma)))
```

```
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.078 seconds (Warm-up)
## Chain 1:                0.074 seconds (Sampling)
## Chain 1:                0.152 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.082 seconds (Warm-up)
## Chain 2:                0.076 seconds (Sampling)
## Chain 2:                0.158 seconds (Total)
## Chain 2:
##
```

```
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.079 seconds (Warm-up)
## Chain 3:                0.09 seconds (Sampling)
## Chain 3:                0.169 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.077 seconds (Warm-up)
## Chain 4:                0.071 seconds (Sampling)
## Chain 4:                0.148 seconds (Total)
## Chain 4:
```

```
h8.4 <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
```

```
                prior(normal(0, 10), class = b),
                prior(cauchy(0, 0.1), class = sigma)))
```

```
##
## SAMPLING FOR MODEL '05e77ec8fc13c439e6a28b92e23f519e' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.08 seconds (Warm-up)
## Chain 1:                0.071 seconds (Sampling)
## Chain 1:                0.151 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL '05e77ec8fc13c439e6a28b92e23f519e' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.079 seconds (Warm-up)
## Chain 2:                0.075 seconds (Sampling)
## Chain 2:                0.154 seconds (Total)
## Chain 2:
##
```

```
## SAMPLING FOR MODEL '05e77ec8fc13c439e6a28b92e23f519e' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.08 seconds (Warm-up)
## Chain 3:                0.079 seconds (Sampling)
## Chain 3:                0.159 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL '05e77ec8fc13c439e6a28b92e23f519e' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.081 seconds (Warm-up)
## Chain 4:                0.078 seconds (Sampling)
## Chain 4:                0.159 seconds (Total)
## Chain 4:
```

```
h8.5 <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
```

```
                prior(normal(0, 10), class = b),
                prior(cauchy(0, 0.01), class = sigma)))
```

```
##
## SAMPLING FOR MODEL '9a6d059446445837221659 4cf67add27' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.08 seconds (Warm-up)
## Chain 1:                0.076 seconds (Sampling)
## Chain 1:                0.156 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL '9a6d059446445837221659 4cf67add27' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.084 seconds (Warm-up)
## Chain 2:                0.086 seconds (Sampling)
## Chain 2:                0.17 seconds (Total)
## Chain 2:
##
```

```
## SAMPLING FOR MODEL '9a6d0594464458372216594cf67add27' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0.002 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 20 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.081 seconds (Warm-up)
## Chain 3:                0.077 seconds (Sampling)
## Chain 3:                0.158 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL '9a6d0594464458372216594cf67add27' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.083 seconds (Warm-up)
## Chain 4:                0.076 seconds (Sampling)
## Chain 4:                0.159 seconds (Total)
## Chain 4:
```

```
h8.6 <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
```

```
              prior(normal(0, 10), class = b),
              prior(exponential(1), class = sigma)))
```

```
##
## SAMPLING FOR MODEL 'c6c3453012689a25e9ad9865ececf1ca' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.077 seconds (Warm-up)
## Chain 1:                0.075 seconds (Sampling)
## Chain 1:                0.152 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'c6c3453012689a25e9ad9865ececf1ca' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.09 seconds (Warm-up)
## Chain 2:                0.081 seconds (Sampling)
## Chain 2:                0.171 seconds (Total)
## Chain 2:
##
```

```
## SAMPLING FOR MODEL 'c6c3453012689a25e9ad9865ececf1ca' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.075 seconds (Warm-up)
## Chain 3:                0.071 seconds (Sampling)
## Chain 3:                0.146 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'c6c3453012689a25e9ad9865ececf1ca' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.078 seconds (Warm-up)
## Chain 4:                0.088 seconds (Sampling)
## Chain 4:                0.166 seconds (Total)
## Chain 4:
```

```
h8.7 <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
```

```
                prior(normal(0, 10), class = b),
                prior(exponential(0.1), class = sigma)))
```

```
##
## SAMPLING FOR MODEL 'f699d6f2b79eeeb421e019d529e275fb' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.082 seconds (Warm-up)
## Chain 1:                0.09 seconds (Sampling)
## Chain 1:                0.172 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'f699d6f2b79eeeb421e019d529e275fb' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.08 seconds (Warm-up)
## Chain 2:                0.078 seconds (Sampling)
## Chain 2:                0.158 seconds (Total)
## Chain 2:
##
```

```
## SAMPLING FOR MODEL 'f699d6f2b79eeeb421e019d529e275fb' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.078 seconds (Warm-up)
## Chain 3:                0.087 seconds (Sampling)
## Chain 3:                0.165 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'f699d6f2b79eeeb421e019d529e275fb' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.078 seconds (Warm-up)
## Chain 4:                0.077 seconds (Sampling)
## Chain 4:                0.155 seconds (Total)
## Chain 4:
```

```
h8.8 <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
```

```
              prior(normal(0, 10), class = b),
              prior(exponential(0.01), class = sigma)))
```

```
##
## SAMPLING FOR MODEL '7ee606aced45c4bff7b8bd8a76a84718' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.079 seconds (Warm-up)
## Chain 1:                0.073 seconds (Sampling)
## Chain 1:                0.152 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL '7ee606aced45c4bff7b8bd8a76a84718' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.08 seconds (Warm-up)
## Chain 2:                0.08 seconds (Sampling)
## Chain 2:                0.16 seconds (Total)
## Chain 2:
##
```

```
## SAMPLING FOR MODEL '7ee606aced45c4bff7b8bd8a76a84718' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.079 seconds (Warm-up)
## Chain 3:                0.08 seconds (Sampling)
## Chain 3:                0.159 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL '7ee606aced45c4bff7b8bd8a76a84718' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.077 seconds (Warm-up)
## Chain 4:                0.083 seconds (Sampling)
## Chain 4:                0.16 seconds (Total)
## Chain 4:
```

```
print(h8.3)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
```

```
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.23      0.14     8.96     9.49       3032 1.00
## rugged               -0.20      0.08    -0.36    -0.05       2769 1.00
## cont_africa          -1.95      0.23    -2.41    -1.52       2383 1.00
## rugged:cont_africa    0.40      0.13     0.15     0.66       2180 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.85     1.05       4194 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.4)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.22      0.14     8.95     9.50       2682 1.00
## rugged               -0.20      0.08    -0.36    -0.05       2623 1.00
## cont_africa          -1.95      0.23    -2.40    -1.49       2346 1.00
## rugged:cont_africa    0.39      0.14     0.12     0.66       2279 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.94      0.05     0.84     1.06       3846 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.5)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.22      0.14     8.95     9.50       2764 1.00
```

```
## rugged                -0.20      0.08    -0.36    -0.05       2515 1.00
## cont_africa           -1.94      0.23    -2.39    -1.48       2403 1.00
## rugged:cont_africa     0.39      0.13     0.13     0.65       2124 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.85     1.05       3729 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.6)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                    Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept              9.22      0.14     8.95     9.50       2928 1.00
## rugged                -0.20      0.08    -0.35    -0.05       2810 1.00
## cont_africa           -1.95      0.23    -2.40    -1.50       2496 1.00
## rugged:cont_africa     0.39      0.13     0.14     0.64       2547 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.86     1.05       3971 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.7)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                    Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept              9.22      0.15     8.93     9.51       2628 1.00
## rugged                -0.20      0.08    -0.35    -0.05       2518 1.00
## cont_africa           -1.94      0.23    -2.40    -1.48       2058 1.00
## rugged:cont_africa     0.39      0.14     0.12     0.65       1860 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.86     1.06       3889 1.00
```

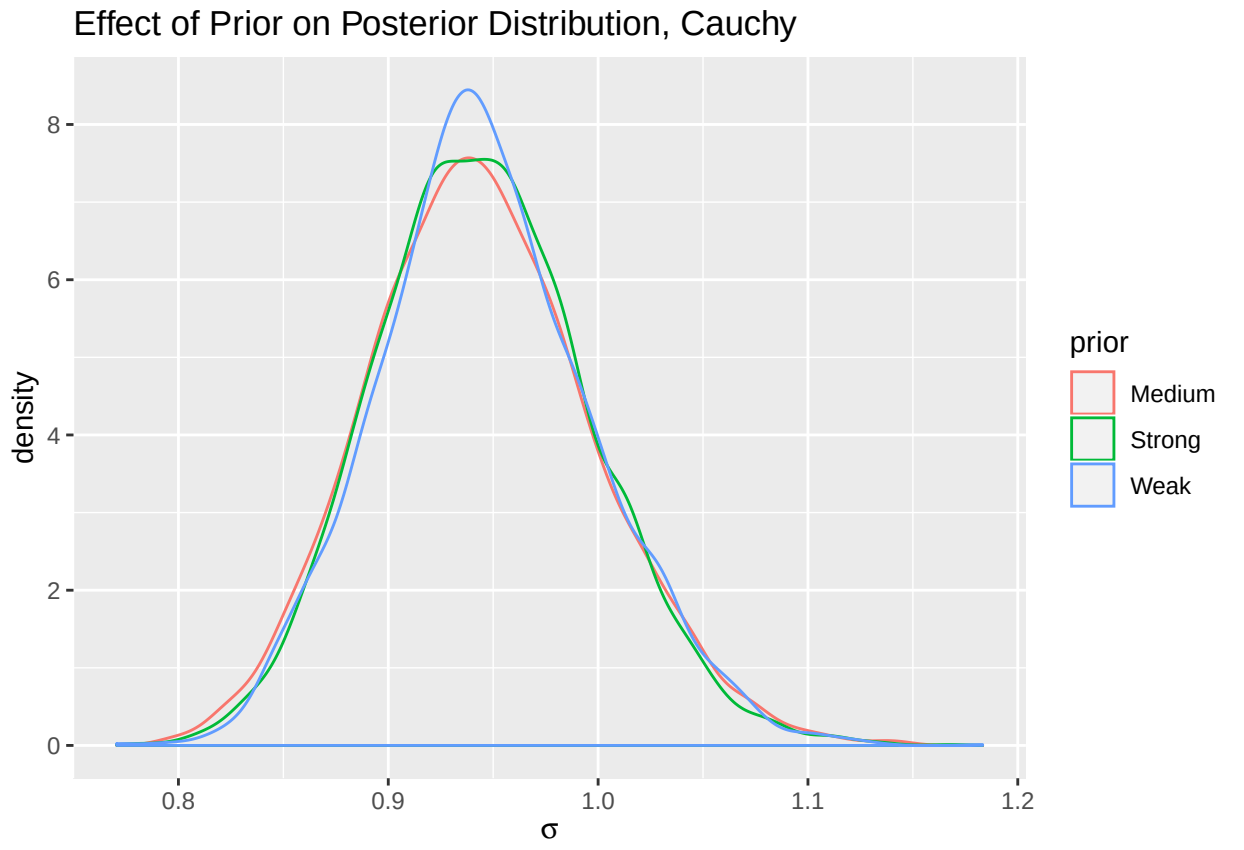
```
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.8)
```

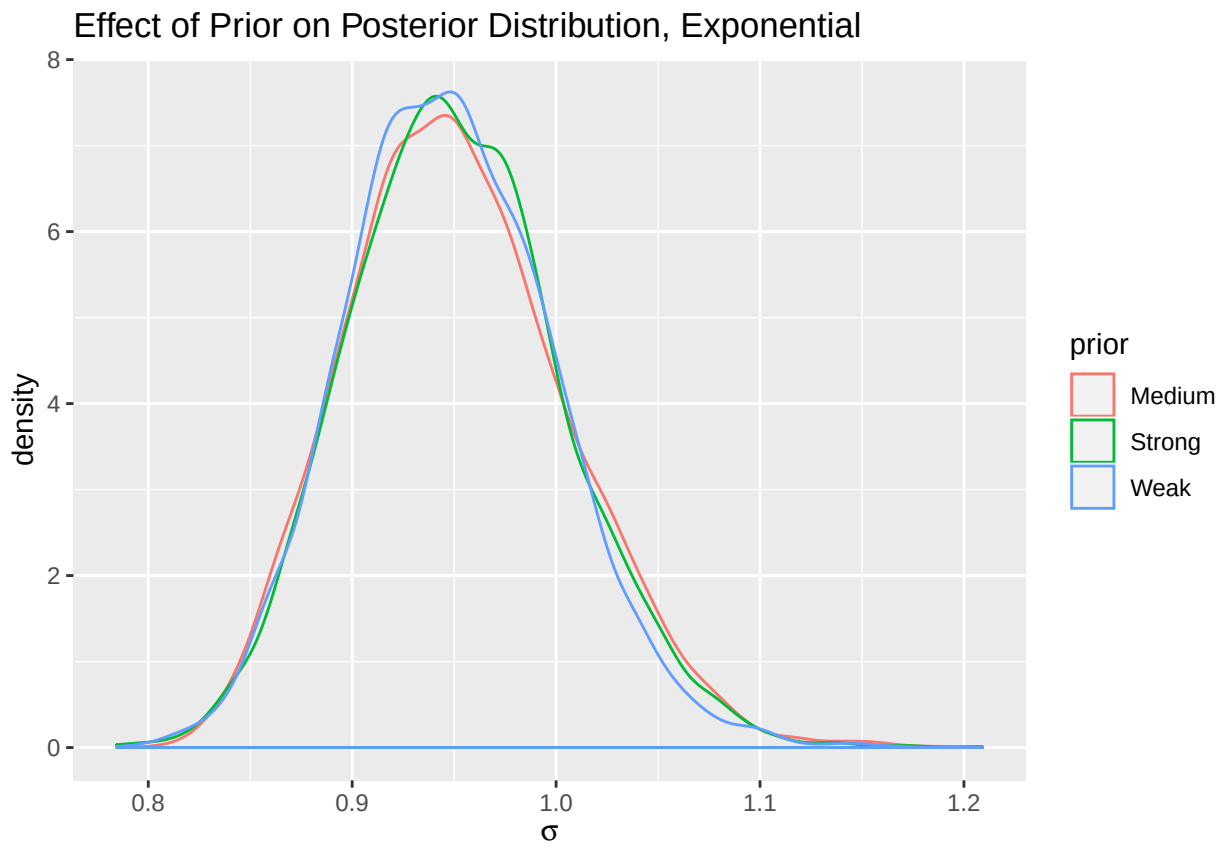
```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                    Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept              9.22      0.14     8.95     9.50       2901 1.00
## rugged                -0.21      0.08    -0.36    -0.05       2620 1.00
## cont_africa           -1.95      0.23    -2.40    -1.51       2594 1.00
## rugged:cont_africa     0.40      0.13     0.14     0.66       2377 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.85     1.06       4025 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
post_c1 <- posterior_samples(h8.3)
post_c2 <- posterior_samples(h8.4)
post_c3 <- posterior_samples(h8.5)

post_c <- cbind(post_c1$sigma, post_c2$sigma, post_c3$sigma)
colnames(post_c) <- c("Weak", "Medium", "Strong")
post_c <- as.data.frame(post_c)
post_c <- gather(post_c, key = prior, value = sigma, Weak:Strong)
post_c$prior <- as.factor(post_c$prior)

c <- ggplot(data = post_c,
      aes(x = sigma, color = prior)) +
      geom_density() +
      xlab(expression(sigma)) +
      ggtitle("Effect of Prior on Posterior Distribution, Cauchy")

post_e1 <- posterior_samples(h8.6)
post_e2 <- posterior_samples(h8.7)
post_e3 <- posterior_samples(h8.8)

post_e <- cbind(post_e1$sigma, post_e2$sigma, post_e3$sigma)
colnames(post_e) <- c("Weak", "Medium", "Strong")
post_e <- as.data.frame(post_e)
post_e <- gather(post_e, key = prior, value = sigma, Weak:Strong)
post_e$prior <- as.factor(post_e$prior)
```

```
e <- ggplot(data = post_e,
       aes(x = sigma, color = prior)) +
       geom_density() +
       xlab(expression(sigma)) +
       ggtitle("Effect of Prior on Posterior Distribution, Exponential")

c
```




```
e
```



These two plots show the posterior distribution of sigma for the models with weak, medium, and strong priors that are either specified to be of the cauchy functional form or the exponential functional form. Interestingly, neither plot shows much difference in the shape or precision of the posterior distribution across prior strengths. The strongest prior density line on the plot with the information from the models estimated with the cauchy prior does show slightly more precision and a higher peak than the weak or medium priors, but not by much. Similarly, the output shows that the estimates are all about the same across the priors. The plot with the posterior samples drawn from models estimated using the exponential prior are even more similar, such that there are very few differences between the density lines for the priors of different strengths.

The reason for this similarity across prior strengths is likely because there is enough data to overwhelm the prior, meaning that the specification of the prior has very little effect on the final estimates produced by the model.

8M3

```
h8.9 <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
                prior(normal(0, 10), class = b),
                prior(cauchy(0, 1), class = sigma)),
      iter = 2000, warmup = 25)
```

```
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 1).
```

```
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: WARNING: There aren't enough warmup iterations to fit the
## Chain 1:          three stages of adaptation as currently configured.
## Chain 1:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 1:          the given number of warmup iterations:
## Chain 1:            init_buffer = 3
## Chain 1:            adapt_window = 20
## Chain 1:            term_buffer = 2
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:   26 / 2000 [  1%]  (Sampling)
## Chain 1: Iteration:  225 / 2000 [ 11%]  (Sampling)
## Chain 1: Iteration:  425 / 2000 [ 21%]  (Sampling)
## Chain 1: Iteration:  625 / 2000 [ 31%]  (Sampling)
## Chain 1: Iteration:  825 / 2000 [ 41%]  (Sampling)
## Chain 1: Iteration: 1025 / 2000 [ 51%]  (Sampling)
## Chain 1: Iteration: 1225 / 2000 [ 61%]  (Sampling)
## Chain 1: Iteration: 1425 / 2000 [ 71%]  (Sampling)
## Chain 1: Iteration: 1625 / 2000 [ 81%]  (Sampling)
## Chain 1: Iteration: 1825 / 2000 [ 91%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.002 seconds (Warm-up)
## Chain 1:                0.199 seconds (Sampling)
## Chain 1:                0.201 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: WARNING: There aren't enough warmup iterations to fit the
## Chain 2:          three stages of adaptation as currently configured.
## Chain 2:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 2:          the given number of warmup iterations:
## Chain 2:            init_buffer = 3
## Chain 2:            adapt_window = 20
## Chain 2:            term_buffer = 2
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:   26 / 2000 [  1%]  (Sampling)
## Chain 2: Iteration:  225 / 2000 [ 11%]  (Sampling)
## Chain 2: Iteration:  425 / 2000 [ 21%]  (Sampling)
## Chain 2: Iteration:  625 / 2000 [ 31%]  (Sampling)
## Chain 2: Iteration:  825 / 2000 [ 41%]  (Sampling)
## Chain 2: Iteration: 1025 / 2000 [ 51%]  (Sampling)
```

```
## Chain 2: Iteration: 1225 / 2000 [ 61%]  (Sampling)
## Chain 2: Iteration: 1425 / 2000 [ 71%]  (Sampling)
## Chain 2: Iteration: 1625 / 2000 [ 81%]  (Sampling)
## Chain 2: Iteration: 1825 / 2000 [ 91%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.002 seconds (Warm-up)
## Chain 2:                0.211 seconds (Sampling)
## Chain 2:                0.213 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: WARNING: There aren't enough warmup iterations to fit the
## Chain 3:          three stages of adaptation as currently configured.
## Chain 3:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 3:          the given number of warmup iterations:
## Chain 3:            init_buffer = 3
## Chain 3:            adapt_window = 20
## Chain 3:            term_buffer = 2
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:   26 / 2000 [  1%]  (Sampling)
## Chain 3: Iteration:  225 / 2000 [ 11%]  (Sampling)
## Chain 3: Iteration:  425 / 2000 [ 21%]  (Sampling)
## Chain 3: Iteration:  625 / 2000 [ 31%]  (Sampling)
## Chain 3: Iteration:  825 / 2000 [ 41%]  (Sampling)
## Chain 3: Iteration: 1025 / 2000 [ 51%]  (Sampling)
## Chain 3: Iteration: 1225 / 2000 [ 61%]  (Sampling)
## Chain 3: Iteration: 1425 / 2000 [ 71%]  (Sampling)
## Chain 3: Iteration: 1625 / 2000 [ 81%]  (Sampling)
## Chain 3: Iteration: 1825 / 2000 [ 91%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.003 seconds (Warm-up)
## Chain 3:                0.208 seconds (Sampling)
## Chain 3:                0.211 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: WARNING: There aren't enough warmup iterations to fit the
## Chain 4:          three stages of adaptation as currently configured.
## Chain 4:          Reducing each adaptation stage to 15%/75%/10% of
```

```
## Chain 4:          the given number of warmup iterations:
## Chain 4:            init_buffer = 3
## Chain 4:            adapt_window = 20
## Chain 4:            term_buffer = 2
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:   26 / 2000 [  1%]  (Sampling)
## Chain 4: Iteration:  225 / 2000 [ 11%]  (Sampling)
## Chain 4: Iteration:  425 / 2000 [ 21%]  (Sampling)
## Chain 4: Iteration:  625 / 2000 [ 31%]  (Sampling)
## Chain 4: Iteration:  825 / 2000 [ 41%]  (Sampling)
## Chain 4: Iteration: 1025 / 2000 [ 51%]  (Sampling)
## Chain 4: Iteration: 1225 / 2000 [ 61%]  (Sampling)
## Chain 4: Iteration: 1425 / 2000 [ 71%]  (Sampling)
## Chain 4: Iteration: 1625 / 2000 [ 81%]  (Sampling)
## Chain 4: Iteration: 1825 / 2000 [ 91%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.003 seconds (Warm-up)
## Chain 4:                0.201 seconds (Sampling)
## Chain 4:                0.204 seconds (Total)
## Chain 4:
```

```
h8.10 <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
                prior(normal(0, 10), class = b),
                prior(cauchy(0, 1), class = sigma)),
      iter = 2000, warmup = 50)
```

```
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: WARNING: There aren't enough warmup iterations to fit the
## Chain 1:          three stages of adaptation as currently configured.
## Chain 1:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 1:          the given number of warmup iterations:
## Chain 1:            init_buffer = 7
## Chain 1:            adapt_window = 38
## Chain 1:            term_buffer = 5
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:   51 / 2000 [  2%]  (Sampling)
## Chain 1: Iteration:  250 / 2000 [ 12%]  (Sampling)
## Chain 1: Iteration:  450 / 2000 [ 22%]  (Sampling)
## Chain 1: Iteration:  650 / 2000 [ 32%]  (Sampling)
## Chain 1: Iteration:  850 / 2000 [ 42%]  (Sampling)
## Chain 1: Iteration: 1050 / 2000 [ 52%]  (Sampling)
## Chain 1: Iteration: 1250 / 2000 [ 62%]  (Sampling)
```

```
## Chain 1: Iteration: 1450 / 2000 [ 72%]  (Sampling)
## Chain 1: Iteration: 1650 / 2000 [ 82%]  (Sampling)
## Chain 1: Iteration: 1850 / 2000 [ 92%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.006 seconds (Warm-up)
## Chain 1:                0.258 seconds (Sampling)
## Chain 1:                0.264 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: WARNING: There aren't enough warmup iterations to fit the
## Chain 2:          three stages of adaptation as currently configured.
## Chain 2:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 2:          the given number of warmup iterations:
## Chain 2:            init_buffer = 7
## Chain 2:            adapt_window = 38
## Chain 2:            term_buffer = 5
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:   51 / 2000 [  2%]  (Sampling)
## Chain 2: Iteration:  250 / 2000 [ 12%]  (Sampling)
## Chain 2: Iteration:  450 / 2000 [ 22%]  (Sampling)
## Chain 2: Iteration:  650 / 2000 [ 32%]  (Sampling)
## Chain 2: Iteration:  850 / 2000 [ 42%]  (Sampling)
## Chain 2: Iteration: 1050 / 2000 [ 52%]  (Sampling)
## Chain 2: Iteration: 1250 / 2000 [ 62%]  (Sampling)
## Chain 2: Iteration: 1450 / 2000 [ 72%]  (Sampling)
## Chain 2: Iteration: 1650 / 2000 [ 82%]  (Sampling)
## Chain 2: Iteration: 1850 / 2000 [ 92%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.005 seconds (Warm-up)
## Chain 2:                0.257 seconds (Sampling)
## Chain 2:                0.262 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: WARNING: There aren't enough warmup iterations to fit the
## Chain 3:          three stages of adaptation as currently configured.
## Chain 3:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 3:          the given number of warmup iterations:
```

```
## Chain 3:            init_buffer = 7
## Chain 3:            adapt_window = 38
## Chain 3:            term_buffer = 5
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:   51 / 2000 [  2%]  (Sampling)
## Chain 3: Iteration:  250 / 2000 [ 12%]  (Sampling)
## Chain 3: Iteration:  450 / 2000 [ 22%]  (Sampling)
## Chain 3: Iteration:  650 / 2000 [ 32%]  (Sampling)
## Chain 3: Iteration:  850 / 2000 [ 42%]  (Sampling)
## Chain 3: Iteration: 1050 / 2000 [ 52%]  (Sampling)
## Chain 3: Iteration: 1250 / 2000 [ 62%]  (Sampling)
## Chain 3: Iteration: 1450 / 2000 [ 72%]  (Sampling)
## Chain 3: Iteration: 1650 / 2000 [ 82%]  (Sampling)
## Chain 3: Iteration: 1850 / 2000 [ 92%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.006 seconds (Warm-up)
## Chain 3:                0.23 seconds (Sampling)
## Chain 3:                0.236 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: WARNING: There aren't enough warmup iterations to fit the
## Chain 4:          three stages of adaptation as currently configured.
## Chain 4:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 4:          the given number of warmup iterations:
## Chain 4:            init_buffer = 7
## Chain 4:            adapt_window = 38
## Chain 4:            term_buffer = 5
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:   51 / 2000 [  2%]  (Sampling)
## Chain 4: Iteration:  250 / 2000 [ 12%]  (Sampling)
## Chain 4: Iteration:  450 / 2000 [ 22%]  (Sampling)
## Chain 4: Iteration:  650 / 2000 [ 32%]  (Sampling)
## Chain 4: Iteration:  850 / 2000 [ 42%]  (Sampling)
## Chain 4: Iteration: 1050 / 2000 [ 52%]  (Sampling)
## Chain 4: Iteration: 1250 / 2000 [ 62%]  (Sampling)
## Chain 4: Iteration: 1450 / 2000 [ 72%]  (Sampling)
## Chain 4: Iteration: 1650 / 2000 [ 82%]  (Sampling)
## Chain 4: Iteration: 1850 / 2000 [ 92%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.006 seconds (Warm-up)
## Chain 4:                0.242 seconds (Sampling)
## Chain 4:                0.248 seconds (Total)
## Chain 4:
```

```
h8.11 <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
                prior(normal(0, 10), class = b),
                prior(cauchy(0, 1), class = sigma)),
      iter = 2000, warmup = 100)
```

```
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.001 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: WARNING: There aren't enough warmup iterations to fit the
## Chain 1:          three stages of adaptation as currently configured.
## Chain 1:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 1:          the given number of warmup iterations:
## Chain 1:            init_buffer = 15
## Chain 1:            adapt_window = 75
## Chain 1:            term_buffer = 10
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  101 / 2000 [  5%]  (Sampling)
## Chain 1: Iteration:  300 / 2000 [ 15%]  (Sampling)
## Chain 1: Iteration:  500 / 2000 [ 25%]  (Sampling)
## Chain 1: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 1: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 1: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 1: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 1: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 1: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 1: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.011 seconds (Warm-up)
## Chain 1:                0.191 seconds (Sampling)
## Chain 1:                0.202 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: WARNING: There aren't enough warmup iterations to fit the
## Chain 2:          three stages of adaptation as currently configured.
## Chain 2:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 2:          the given number of warmup iterations:
## Chain 2:            init_buffer = 15
```

```
## Chain 2:            adapt_window = 75
## Chain 2:            term_buffer = 10
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  101 / 2000 [  5%]  (Sampling)
## Chain 2: Iteration:  300 / 2000 [ 15%]  (Sampling)
## Chain 2: Iteration:  500 / 2000 [ 25%]  (Sampling)
## Chain 2: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 2: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 2: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 2: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 2: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 2: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.011 seconds (Warm-up)
## Chain 2:                0.19 seconds (Sampling)
## Chain 2:                0.201 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: WARNING: There aren't enough warmup iterations to fit the
## Chain 3:          three stages of adaptation as currently configured.
## Chain 3:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 3:          the given number of warmup iterations:
## Chain 3:            init_buffer = 15
## Chain 3:            adapt_window = 75
## Chain 3:            term_buffer = 10
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  101 / 2000 [  5%]  (Sampling)
## Chain 3: Iteration:  300 / 2000 [ 15%]  (Sampling)
## Chain 3: Iteration:  500 / 2000 [ 25%]  (Sampling)
## Chain 3: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 3: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 3: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 3: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 3: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 3: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 3: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.012 seconds (Warm-up)
## Chain 3:                0.216 seconds (Sampling)
## Chain 3:                0.228 seconds (Total)
## Chain 3:
##
```

```
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: WARNING: There aren't enough warmup iterations to fit the
## Chain 4:          three stages of adaptation as currently configured.
## Chain 4:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 4:          the given number of warmup iterations:
## Chain 4:            init_buffer = 15
## Chain 4:            adapt_window = 75
## Chain 4:            term_buffer = 10
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  101 / 2000 [  5%]  (Sampling)
## Chain 4: Iteration:  300 / 2000 [ 15%]  (Sampling)
## Chain 4: Iteration:  500 / 2000 [ 25%]  (Sampling)
## Chain 4: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 4: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 4: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 4: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 4: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 4: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 4: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.011 seconds (Warm-up)
## Chain 4:                0.197 seconds (Sampling)
## Chain 4:                0.208 seconds (Total)
## Chain 4:
```

```
h8.11a <-   brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
                prior(normal(0, 10), class = b),
                prior(cauchy(0, 1), class = sigma)),
      iter = 2000, warmup = 250)
```

```
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.001 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  251 / 2000 [ 12%]  (Sampling)
## Chain 1: Iteration:  450 / 2000 [ 22%]  (Sampling)
## Chain 1: Iteration:  650 / 2000 [ 32%]  (Sampling)
## Chain 1: Iteration:  850 / 2000 [ 42%]  (Sampling)
## Chain 1: Iteration: 1050 / 2000 [ 52%]  (Sampling)
```

```
## Chain 1: Iteration: 1250 / 2000 [ 62%]  (Sampling)
## Chain 1: Iteration: 1450 / 2000 [ 72%]  (Sampling)
## Chain 1: Iteration: 1650 / 2000 [ 82%]  (Sampling)
## Chain 1: Iteration: 1850 / 2000 [ 92%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.05 seconds (Warm-up)
## Chain 1:                0.299 seconds (Sampling)
## Chain 1:                0.349 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  251 / 2000 [ 12%]  (Sampling)
## Chain 2: Iteration:  450 / 2000 [ 22%]  (Sampling)
## Chain 2: Iteration:  650 / 2000 [ 32%]  (Sampling)
## Chain 2: Iteration:  850 / 2000 [ 42%]  (Sampling)
## Chain 2: Iteration: 1050 / 2000 [ 52%]  (Sampling)
## Chain 2: Iteration: 1250 / 2000 [ 62%]  (Sampling)
## Chain 2: Iteration: 1450 / 2000 [ 72%]  (Sampling)
## Chain 2: Iteration: 1650 / 2000 [ 82%]  (Sampling)
## Chain 2: Iteration: 1850 / 2000 [ 92%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.049 seconds (Warm-up)
## Chain 2:                0.298 seconds (Sampling)
## Chain 2:                0.347 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  251 / 2000 [ 12%]  (Sampling)
## Chain 3: Iteration:  450 / 2000 [ 22%]  (Sampling)
## Chain 3: Iteration:  650 / 2000 [ 32%]  (Sampling)
## Chain 3: Iteration:  850 / 2000 [ 42%]  (Sampling)
## Chain 3: Iteration: 1050 / 2000 [ 52%]  (Sampling)
## Chain 3: Iteration: 1250 / 2000 [ 62%]  (Sampling)
## Chain 3: Iteration: 1450 / 2000 [ 72%]  (Sampling)
## Chain 3: Iteration: 1650 / 2000 [ 82%]  (Sampling)
## Chain 3: Iteration: 1850 / 2000 [ 92%]  (Sampling)
```

```
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.05 seconds (Warm-up)
## Chain 3:                0.265 seconds (Sampling)
## Chain 3:                0.315 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  251 / 2000 [ 12%]  (Sampling)
## Chain 4: Iteration:  450 / 2000 [ 22%]  (Sampling)
## Chain 4: Iteration:  650 / 2000 [ 32%]  (Sampling)
## Chain 4: Iteration:  850 / 2000 [ 42%]  (Sampling)
## Chain 4: Iteration: 1050 / 2000 [ 52%]  (Sampling)
## Chain 4: Iteration: 1250 / 2000 [ 62%]  (Sampling)
## Chain 4: Iteration: 1450 / 2000 [ 72%]  (Sampling)
## Chain 4: Iteration: 1650 / 2000 [ 82%]  (Sampling)
## Chain 4: Iteration: 1850 / 2000 [ 92%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.043 seconds (Warm-up)
## Chain 4:                0.279 seconds (Sampling)
## Chain 4:                0.322 seconds (Total)
## Chain 4:
```

```
h8.11b <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
                prior(normal(0, 10), class = b),
                prior(cauchy(0, 1), class = sigma)),
      iter = 2000, warmup = 500)
```

```
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 1: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 1: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 1: Iteration: 1100 / 2000 [ 55%]  (Sampling)
```

```
## Chain 1: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 1: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 1: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 1: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.043 seconds (Warm-up)
## Chain 1:                0.139 seconds (Sampling)
## Chain 1:                0.182 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 2: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 2: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 2: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 2: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 2: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 2: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.043 seconds (Warm-up)
## Chain 2:                0.22 seconds (Sampling)
## Chain 2:                0.263 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 3: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 3: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 3: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 3: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 3: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 3: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 3: Iteration: 1900 / 2000 [ 95%]  (Sampling)
```

```
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.091 seconds (Warm-up)
## Chain 3:                0.266 seconds (Sampling)
## Chain 3:                0.357 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  501 / 2000 [ 25%]  (Sampling)
## Chain 4: Iteration:  700 / 2000 [ 35%]  (Sampling)
## Chain 4: Iteration:  900 / 2000 [ 45%]  (Sampling)
## Chain 4: Iteration: 1100 / 2000 [ 55%]  (Sampling)
## Chain 4: Iteration: 1300 / 2000 [ 65%]  (Sampling)
## Chain 4: Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 4: Iteration: 1700 / 2000 [ 85%]  (Sampling)
## Chain 4: Iteration: 1900 / 2000 [ 95%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.084 seconds (Warm-up)
## Chain 4:                0.222 seconds (Sampling)
## Chain 4:                0.306 seconds (Total)
## Chain 4:
```

```
h8.11c <-
  brm(data = dd, family = gaussian,
      log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa,
      prior = c(prior(normal(0, 100), class = Intercept),
                prior(normal(0, 10), class = b),
                prior(cauchy(0, 1), class = sigma)),
      iter = 2000, warmup = 1000)
```

```
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
```

```
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.081 seconds (Warm-up)
## Chain 1:                0.082 seconds (Sampling)
## Chain 1:                0.163 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.076 seconds (Warm-up)
## Chain 2:                0.077 seconds (Sampling)
## Chain 2:                0.153 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
```

```
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.078 seconds (Warm-up)
## Chain 3:                0.074 seconds (Sampling)
## Chain 3:                0.152 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'f154372ece3580b00a5c5bf4cdf8c171' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.083 seconds (Warm-up)
## Chain 4:                0.077 seconds (Sampling)
## Chain 4:                0.16 seconds (Total)
## Chain 4:
```

```
print(h8.9)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 25; thin = 1;
##          total post-warmup samples = 7900
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.23      0.15     8.96     9.51       3608 1.00
## rugged               -0.21      0.08    -0.36    -0.06       4004 1.00
## cont_africa          -1.96      0.23    -2.40    -1.51       2089 1.00
## rugged:cont_africa    0.40      0.13     0.14     0.65       2677 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.06     0.85     1.06       7933 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.10)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 50; thin = 1;
##          total post-warmup samples = 7800
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.22      0.14     8.95     9.50       3450 1.00
## rugged               -0.20      0.08    -0.36    -0.05       3906 1.00
## cont_africa          -1.95      0.23    -2.40    -1.49       1894 1.00
## rugged:cont_africa    0.39      0.13     0.13     0.66       2383 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.85     1.06       9843 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.11)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 100; thin = 1;
##          total post-warmup samples = 7600
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.23      0.14     8.95     9.50       3725 1.00
## rugged               -0.20      0.08    -0.36    -0.05       4149 1.00
## cont_africa          -1.95      0.23    -2.40    -1.49       2009 1.00
## rugged:cont_africa    0.40      0.13     0.14     0.65       2592 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.85     1.05      10280 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.11a)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 250; thin = 1;
```

```
##          total post-warmup samples = 7000
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.22      0.14     8.95     9.50       4658 1.00
## rugged               -0.20      0.08    -0.36    -0.05       4507 1.00
## cont_africa          -1.95      0.23    -2.39    -1.49       4305 1.00
## rugged:cont_africa    0.39      0.13     0.13     0.65       4097 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.85     1.05       6680 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.11b)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 500; thin = 1;
##          total post-warmup samples = 6000
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.23      0.14     8.94     9.51       3833 1.00
## rugged               -0.20      0.08    -0.36    -0.05       3791 1.00
## cont_africa          -1.95      0.24    -2.42    -1.49       3599 1.00
## rugged:cont_africa    0.39      0.14     0.13     0.66       3241 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.85     1.06       5862 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
print(h8.11c)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: log_gdp ~ 1 + rugged + cont_africa + rugged:cont_africa
##    Data: dd (Number of observations: 170)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                   Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept             9.22      0.14     8.95     9.49       2665 1.00
## rugged               -0.20      0.08    -0.35    -0.05       2660 1.00
## cont_africa          -1.94      0.23    -2.39    -1.48       2503 1.00
```

```
## rugged:cont_africa     0.39      0.13     0.12     0.65       2397 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     0.95      0.05     0.85     1.06       3948 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Estimating the terrain model at different levels of warmup (from 25 to 1000 across the six models estimated above) shows the importance of specifying enough warmup iterations for an adequate number of effective samples. The effect that warmup length has is most noticeable when looking across the model results at the parameter for cont_africa. Across the first three models (h8.9, h8.10, h8.11), the number of effective samples for this parameter is less than 2000, while the rest of the parameters in each of these models have much more than 2000 effective samples. Once the specification for warmup is 250 iterations in model h8.11a, the number of effective samples for cont_africa is much more consistent with the rest of the parameters, and furthermore, the effective samples for all parameters in this model is much higher than in any of the previous models. Interestingly, when the warmup is specified to be 1000 in the final model, h8.11c, the number of effective samples actually decreases, perhaps because too much of the time is being used for warmup rather than actual estimation. Based on these results, I think that a good rule of thumb is more than 200 warmup iterations.

Hard Problems

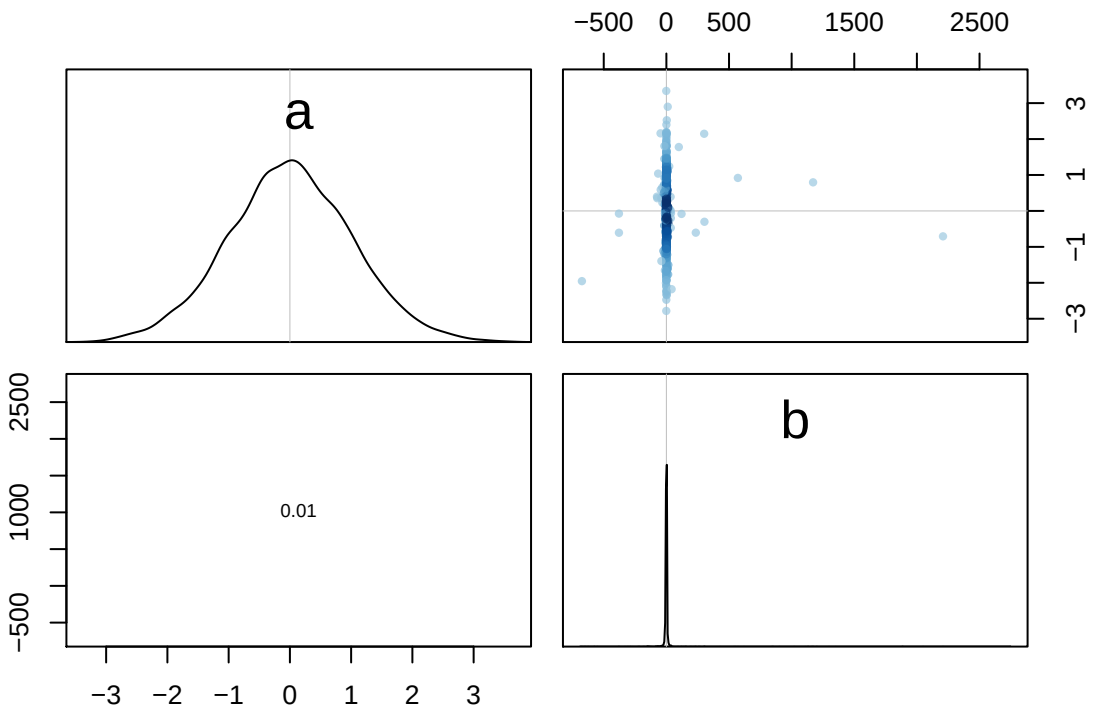
8H1

```
h8.12 <- map2stan(
            alist(
              a ~ dnorm(0, 1),
              b ~ dcauchy(0,1)
            ),
          data = list(y = 1),
          start = list( a = 0, b = 0),
          iter = 1e4, warmup = 100, WAIC = FALSE)
```

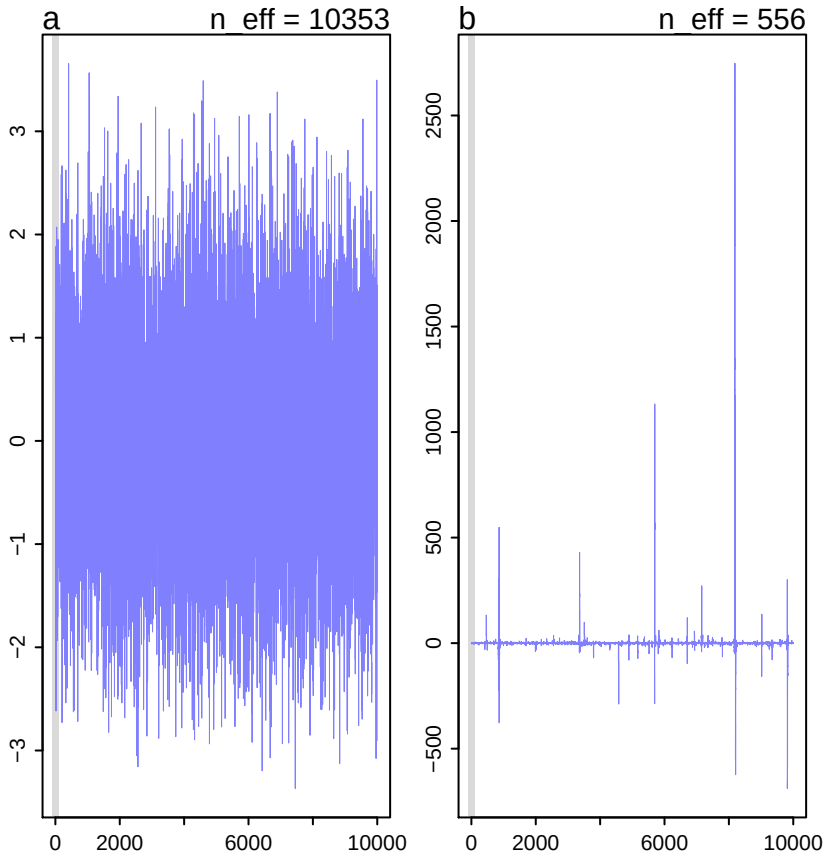
```
##
## SAMPLING FOR MODEL 'a ~ dnorm(0, 1)' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: WARNING: There aren't enough warmup iterations to fit the
## Chain 1:          three stages of adaptation as currently configured.
## Chain 1:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 1:          the given number of warmup iterations:
## Chain 1:            init_buffer = 15
## Chain 1:            adapt_window = 75
## Chain 1:            term_buffer = 10
## Chain 1:
## Chain 1: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 1: Iteration:  101 / 10000 [  1%]  (Sampling)
```

```
## Chain 1: Iteration: 1100 / 10000 [ 11%]  (Sampling)
## Chain 1: Iteration: 2100 / 10000 [ 21%]  (Sampling)
## Chain 1: Iteration: 3100 / 10000 [ 31%]  (Sampling)
## Chain 1: Iteration: 4100 / 10000 [ 41%]  (Sampling)
## Chain 1: Iteration: 5100 / 10000 [ 51%]  (Sampling)
## Chain 1: Iteration: 6100 / 10000 [ 61%]  (Sampling)
## Chain 1: Iteration: 7100 / 10000 [ 71%]  (Sampling)
## Chain 1: Iteration: 8100 / 10000 [ 81%]  (Sampling)
## Chain 1: Iteration: 9100 / 10000 [ 91%]  (Sampling)
## Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.001 seconds (Warm-up)
## Chain 1:                0.208 seconds (Sampling)
## Chain 1:                0.209 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'a ~ dnorm(0, 1)' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: WARNING: No variance estimation is
## Chain 1:          performed for num_warmup < 20
## Chain 1:
## Chain 1: Iteration: 1 / 1 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0 seconds (Warm-up)
## Chain 1:                0 seconds (Sampling)
## Chain 1:                0 seconds (Total)
## Chain 1:
```

```
post <- extract.samples(h8.12)

pairs(h8.12)
```




```
plot(h8.12)
```



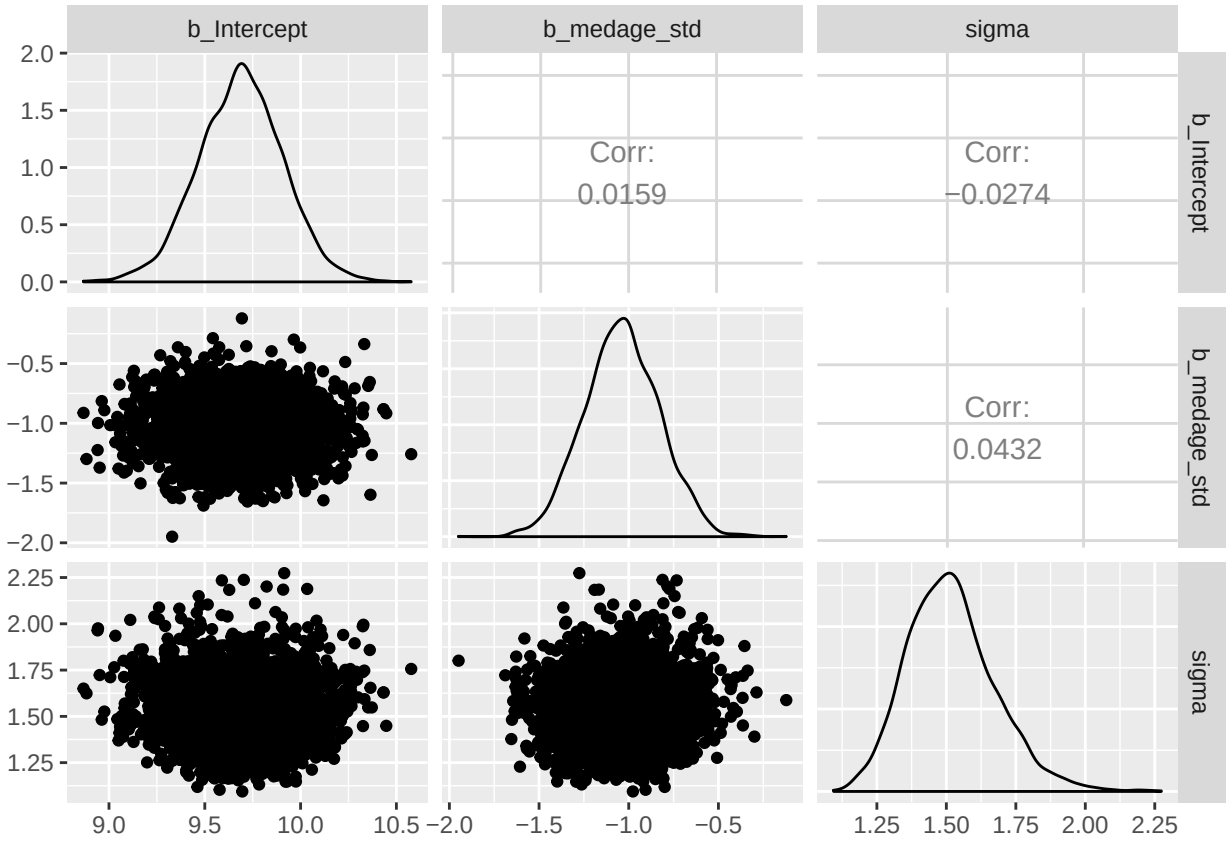
Since this model does not include the specification of an equation specifying mu as a function of the parameters, it appears that this model is essentially just sampling from the two distributions: normal with a mean of 0 and a standard deviation of 1 for a and cauchy with location 0 and scale 1 for b.

The trace plot and number of effective samples for a appears to be healthy, exhibiting both stationarity and mixing. However, the second trace plot of parameter b, which is sampling from the cauchy distribution has neither of those features. Because the cauchy distribution has very thick tails, it is likely that what is occuring is that extreme values are occasionally being sampled and making the chain get stuck around those values, resulting in the spikes seen in the trace plot. This additionally increases the autocorrelation and thus decreases the number of effective samples.

8H2

```
data(WaffleDivorce)
d <- WaffleDivorce

d <- d %>% mutate(medage_std = (MedianAgeMarriage - mean(MedianAgeMarriage))/ sd(MedianAgeMarriage),
                  marriage_std = (Marriage - mean(Marriage)) / sd(Marriage))

#m5.1
h8.13 <-  brm(data = d, family = gaussian,
      Divorce ~ 1 + medage_std,
      prior = c(prior(normal(10, 10), class = Intercept),
                prior(normal(0, 1), class = b),
                prior(uniform(0, 10), class = sigma)),
      iter = 2000, warmup = 1000, chains = 4, cores = 4)
```

```
#m5.2
h8.14 <- brm(data = d, family = gaussian,
             Divorce ~ 1 + marriage_std,
             prior = c(prior(normal(10, 10), class = Intercept),
                       prior(normal(0, 1), class = b),
                       prior(uniform(0, 10), class = sigma)),
             iter = 2000, warmup = 1000, chains = 4, cores = 4)

#m5.3
h8.15 <- brm(data = d, family = gaussian,
      Divorce ~ 1 + medage_std + marriage_std,
      prior = c(prior(normal(10, 10), class = Intercept),
                prior(normal(0, 1), class = b),
                prior(uniform(0, 10), class = sigma)),
      iter = 2000, warmup = 1000, chains = 4, cores = 4)

library(GGally)
post <- posterior_samples(h8.13)
post %>%
  select(b_Intercept:sigma) %>%
  ggpairs()
```

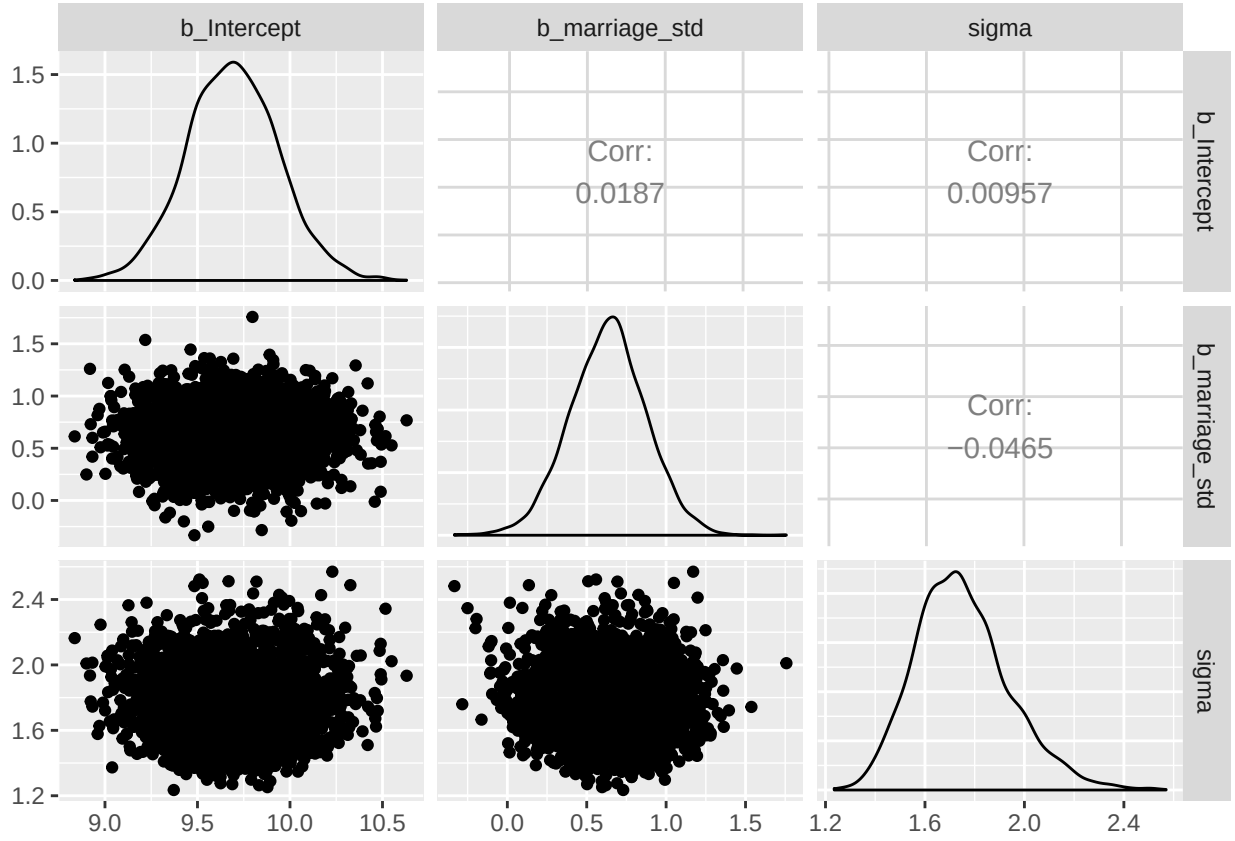



```
print(h8.13)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
```

```
## Formula: Divorce ~ 1 + medage_std
##    Data: d (Number of observations: 50)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##            Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept      9.69      0.22     9.25    10.12       3234 1.00
## medage_std    -1.04      0.21    -1.44    -0.63       3453 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     1.52      0.16     1.25     1.87       3306 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
post <- posterior_samples(h8.14)
post %>%
  select(b_Intercept:sigma) %>%
  ggpairs()
```

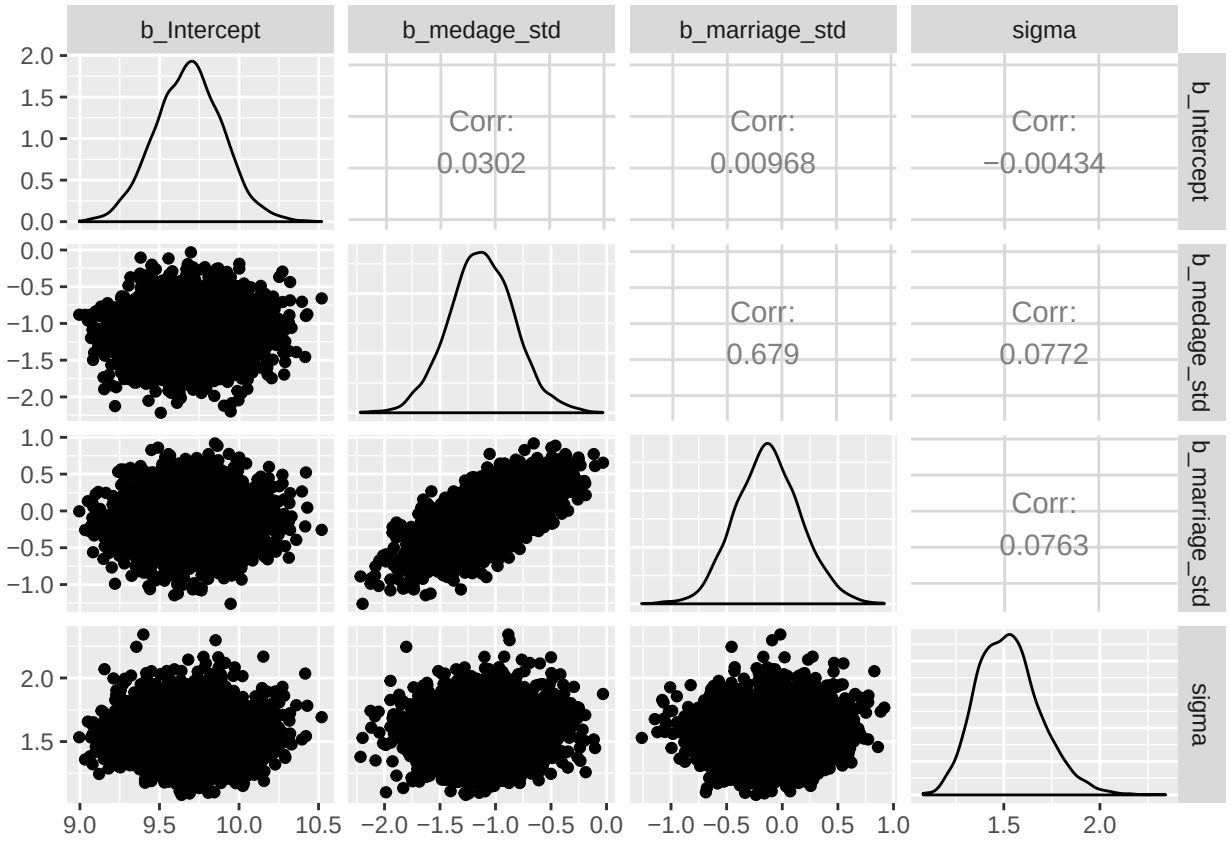



```
print(h8.14)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
```

```
## Formula: Divorce ~ 1 + marriage_std
##    Data: d (Number of observations: 50)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##              Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept        9.69      0.25     9.21    10.19       4209 1.00
## marriage_std     0.63      0.24     0.17     1.10       4264 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     1.75      0.19     1.42     2.17       4087 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
post <- posterior_samples(h8.15)
post %>%
  select(b_Intercept:sigma) %>%
  ggpairs()
```




```
print(h8.15)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
```

```
## Formula: Divorce ~ 1 + medage_std + marriage_std
##    Data: d (Number of observations: 50)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##              Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## Intercept        9.69      0.21     9.27    10.11       3943 1.00
## medage_std      -1.12      0.30    -1.71    -0.51       2993 1.00
## marriage_std    -0.13      0.29    -0.68     0.46       2999 1.00
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
## sigma     1.53      0.17     1.24     1.89       3413 1.00
##
## Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
## is a crude measure of effective sample size, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

These three models investigate the relationship between marriage rate, marriage age, and divorce rate, as we did in chapter 5, but this time using MCMC instead of MAP.

In the first model (h8.13), divorce is predicted by the median age of marriage, standardized. The estimate for this parameter is -1.04, with a 95% confidence interval that does not contain zero. This indicates that for every one unit increase in the standard deviation of the median age at which people are married, we expect a decrease in divorce rate of -1.04 percentage points.

In the second model (h8.14), divorce is predicted by the marriage rate, standardized. The estimate for this parameter is 0.64, with a 95% confidence interval that does not contain zero. This indicates that for every one unit increase in the standard deviation of the marriage rate, we expect an increase in divorce rate of 0.64 percentage points.

In the final model (h8.15), divorce is predicted by both the standardized marriage rate as well as the standardized median age of marriage. Interestingly, the coefficient for the parameter of marriage rate is now negative, -0.12, and the 95% confidence interval does contain 0, meaning there may be no effect of marriage rate on divorce rate at all. This indicates that the relationship between marriage rate and divorce rate we observed in h8.14 is spurious, and instead is explained through the relationship between median age at marriage and divorce rate. Because states that have high marriage rates also tend to have lower median ages at which people get married, the effect in model h8.14 disappears when median age is also included in the model. In fact, the coefficient for median age at marriage in this model is -1.12, which is even more negative than in h8.13, and the 95% confidence interval still does not contain 0. This estimate indicates that for every one unit increase in the standard deviation of the median age at which people are married, we expect a decrease in divorce rate of -1.12 percentage points.

```
waic(h8.13, h8.14, h8.15)
```

```
##                 WAIC    SE
## h8.13         186.15 12.50
## h8.14         199.92  9.73
## h8.15         188.10 12.64
## h8.13 - h8.14 -13.76  9.06
## h8.13 - h8.15  -1.95  0.82
## h8.14 - h8.15  11.81  9.37
```

To see which model performs best, we compare them to each other using WAIC. The initial model that only included the median age of marriage (h8.13) has the lowest WAIC score, even a little bit lower than the model with both of the parameters (h8.15). I think the reason that h8.13 has a lower WAIC than h8.15 is

because the marriage rate parameter is not adding enough information to counteract the penalty the model takes for adding another parameter. The fact that the model with only the marriage rate coefficient has the highest WAIC by far provides more weight to this conclusion because since the parameter doesn't add very much on its own, it similarly is not helping significantly in the combined model.

8H3

```
n <- 100
height <- rnorm(n, 10, 2)
leg_prop <- runif(n, 0.4, 0.5)
leg_left <- leg_prop*height + rnorm(n, 0, 0.02)
leg_right <- leg_prop*height + rnorm(n, 0, 0.02)
l <- data.frame(height, leg_left, leg_right)

h8.16 <- map2stan(
  alist(height ~ dnorm( mu , sigma ) ,
        mu <- a + bl*leg_left + br*leg_right ,
        a ~ dnorm( 10 , 100 ) ,
        bl ~ dnorm( 2 , 10 ) ,
        br ~ dnorm( 2 , 10 ) ,
        sigma ~ dcauchy( 0 , 1 ) ) ,
      data=l, chains=4, start=list(a=10,bl=0,br=0,sigma=1))
```

```
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 4.538 seconds (Warm-up)
## Chain 1:                6.122 seconds (Sampling)
## Chain 1:                10.66 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
```

```
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 3.958 seconds (Warm-up)
## Chain 2:                5.229 seconds (Sampling)
## Chain 2:                9.187 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 4.044 seconds (Warm-up)
## Chain 3:                5.118 seconds (Sampling)
## Chain 3:                9.162 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
```

```
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 4.249 seconds (Warm-up)
## Chain 4:                6.283 seconds (Sampling)
## Chain 4:                10.532 seconds (Total)
## Chain 4:
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: WARNING: No variance estimation is
## Chain 1:          performed for num_warmup < 20
## Chain 1:
## Chain 1: Iteration: 1 / 1 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0 seconds (Warm-up)
## Chain 1:                0 seconds (Sampling)
## Chain 1:                0 seconds (Total)
## Chain 1:
## [ 400 / 4000 ]
[ 800 / 4000 ]
[ 1200 / 4000 ]
[ 1600 / 4000 ]
[ 2000 / 4000 ]
[ 2400 / 4000 ]
[ 2800 / 4000 ]
[ 3200 / 4000 ]
[ 3600 / 4000 ]
[ 4000 / 4000 ]
```

```
h8.17 <- map2stan(
  alist(height ~ dnorm( mu , sigma) ,
        mu <- a + bl*leg_left + br*leg_right ,
        a ~ dnorm( 10 , 100 ) ,
        bl ~ dnorm( 2 , 10 ) ,
        br ~ dnorm( 2 , 10 ) & T[0,] ,
        sigma ~ dcauchy( 0 , 1 ) ) ,
      data=l, chains=4, start=list(a=10,bl=0,br=0,sigma=1))
```

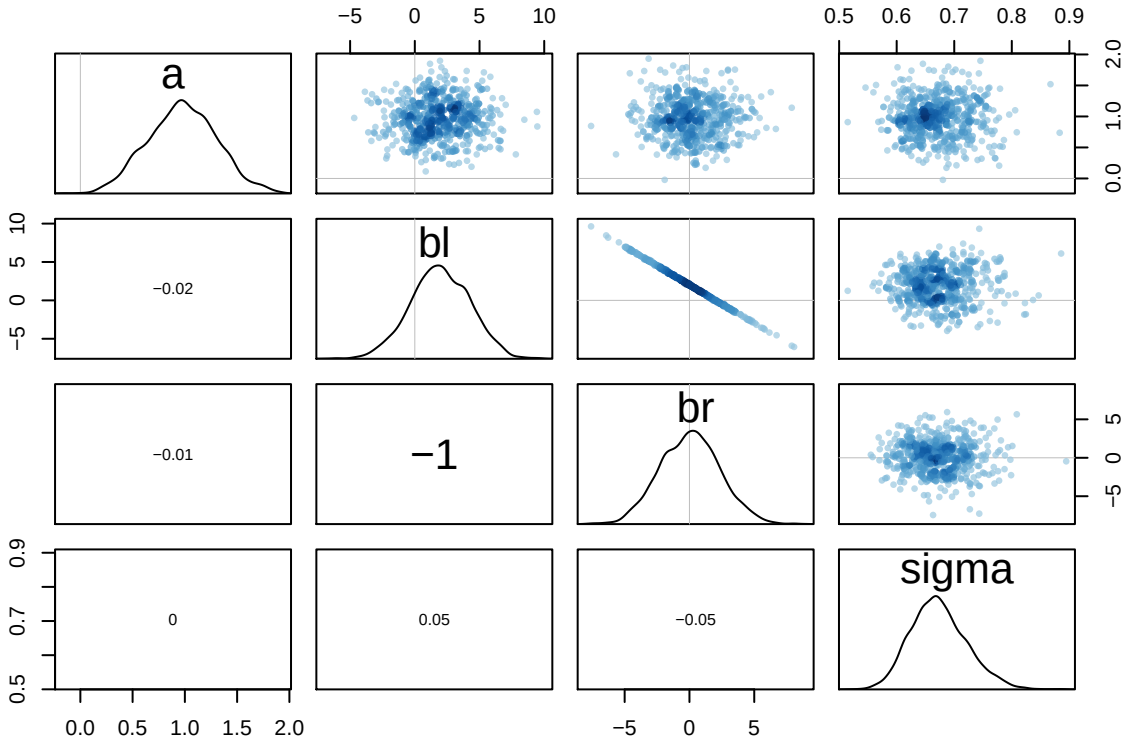
```
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 1).
## Chain 1:
```

```
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.911 seconds (Warm-up)
## Chain 1:                2.553 seconds (Sampling)
## Chain 1:                4.464 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 2.072 seconds (Warm-up)
## Chain 2:                1.939 seconds (Sampling)
## Chain 2:                4.011 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
```

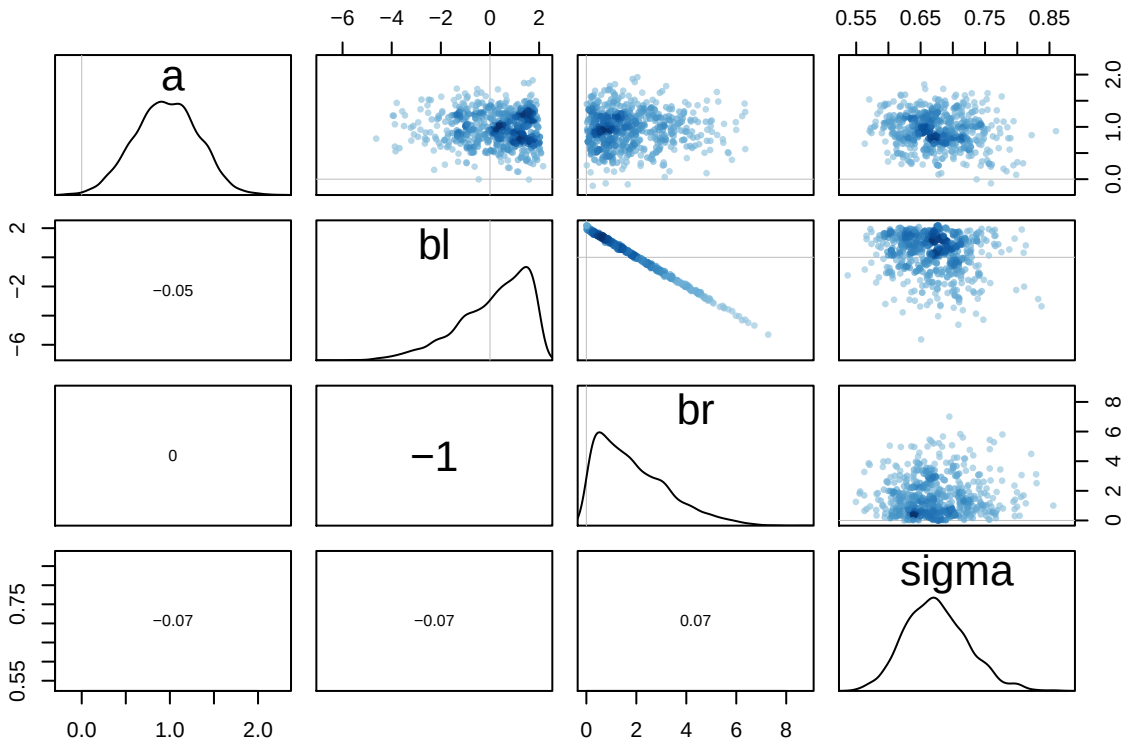
```
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 3.208 seconds (Warm-up)
## Chain 3:                1.913 seconds (Sampling)
## Chain 3:                5.121 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 2.079 seconds (Warm-up)
## Chain 4:                2.593 seconds (Sampling)
## Chain 4:                4.672 seconds (Total)
## Chain 4:
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: WARNING: No variance estimation is
## Chain 1:          performed for num_warmup < 20
## Chain 1:
```

```
## Chain 1: Iteration: 1 / 1 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0 seconds (Warm-up)
## Chain 1:                0 seconds (Sampling)
## Chain 1:                0 seconds (Total)
## Chain 1:
## [ 400 / 4000 ]
[ 800 / 4000 ]
[ 1200 / 4000 ]
[ 1600 / 4000 ]
[ 2000 / 4000 ]
[ 2400 / 4000 ]
[ 2800 / 4000 ]
[ 3200 / 4000 ]
[ 3600 / 4000 ]
[ 4000 / 4000 ]
```

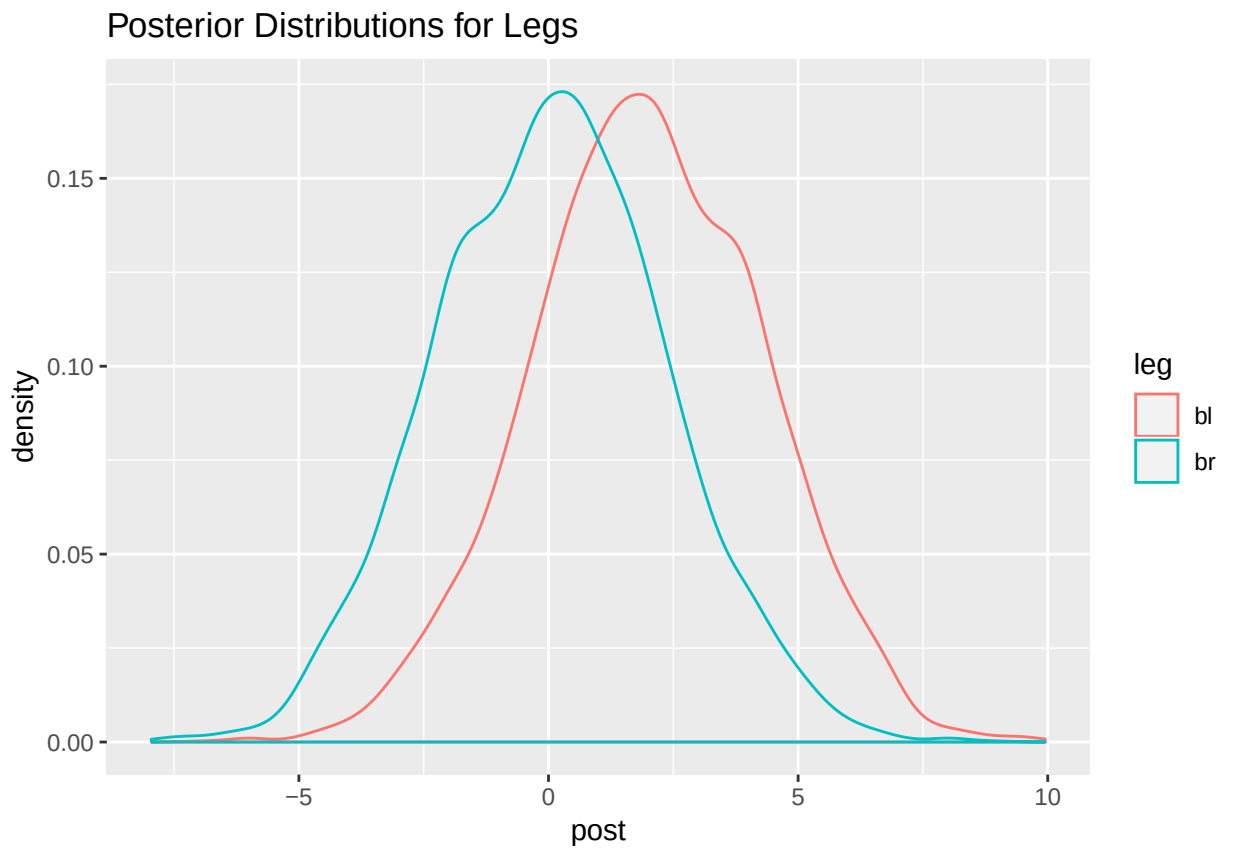
```
pairs(h8.16)
```



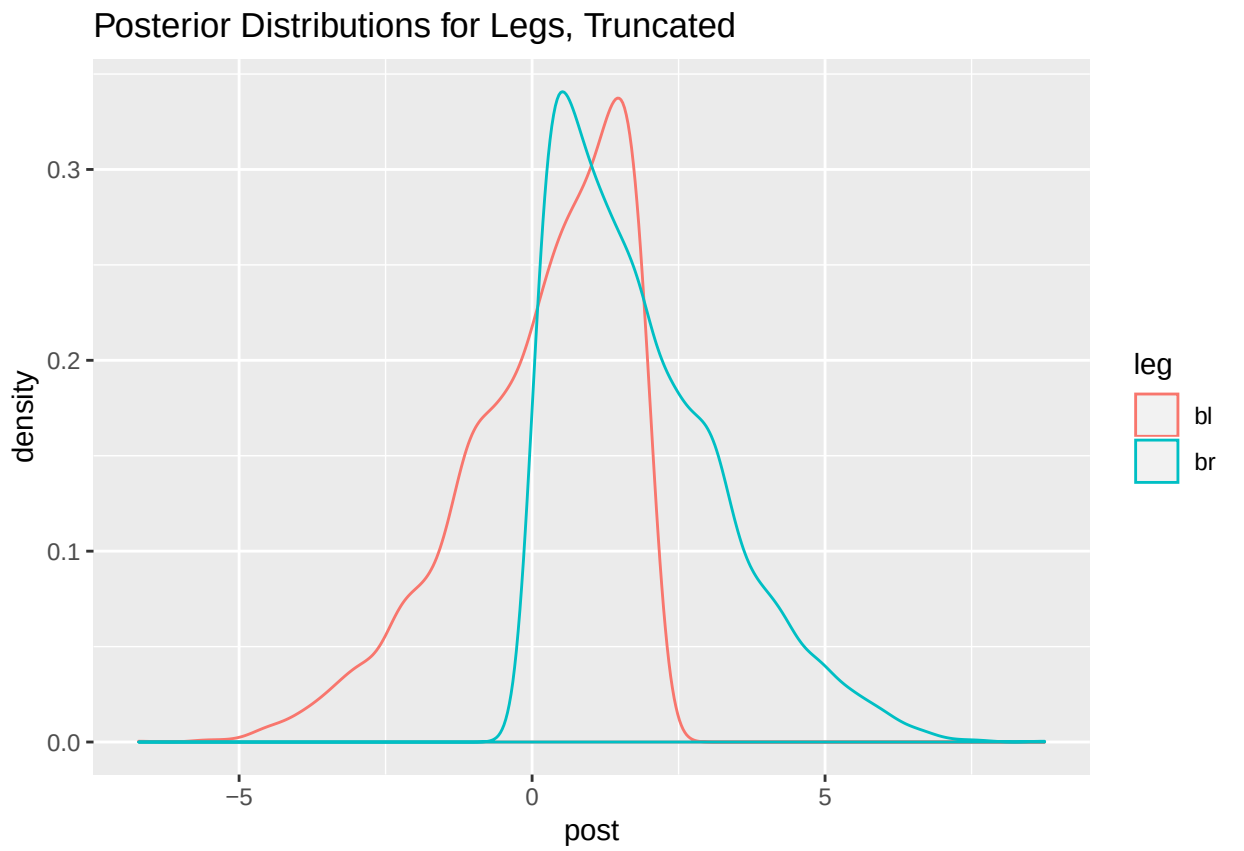

```
pairs(h8.17)
```




```
postcorr <- extract.samples(h8.16)
posttrunc <- extract.samples(h8.17)

postcorr <- data.frame(postcorr)
postcorr <- gather(postcorr, key = leg, value = post, bl:br)

nt <- ggplot(data = postcorr, aes(x = post, color = leg)) +
  geom_density() +
  ggtitle("Posterior Distributions for Legs")

posttrunc <- data.frame(posttrunc)
posttrunc <- gather(posttrunc, key = leg, value = post, bl:br)

t <- ggplot(data = posttrunc, aes(x = post, color = leg)) +
  geom_density() +
  ggtitle("Posterior Distributions for Legs, Truncated")

nt
```


Posterior Distributions for Legs
density
0.15
0.10
0.05
0.00
−5
0
5
10
post
leg
bl
br

```
t
```



Adding the information to the prior for the right leg such that it can only have positive probability above zero resulted in a separation of the posterior distributions of the left and the right leg. Comparing the posterior distribution plots for the model with no truncation and the model that truncated the right leg prior, the former shows essentially no difference in the posterior distributions while the second has the right leg truncated at 0 with only one tail on its right side in the positve range. In this plot, the left leg is the exact opposite with a tail extending into negative numbers and truncated at approximately 2.5.

Because the parameters for the right and left leg are highly correlated with each other, including the prior for the right leg influenced the parameter for the left leg as well, giving the posterior distributions the same shape, just flipped.

8H4

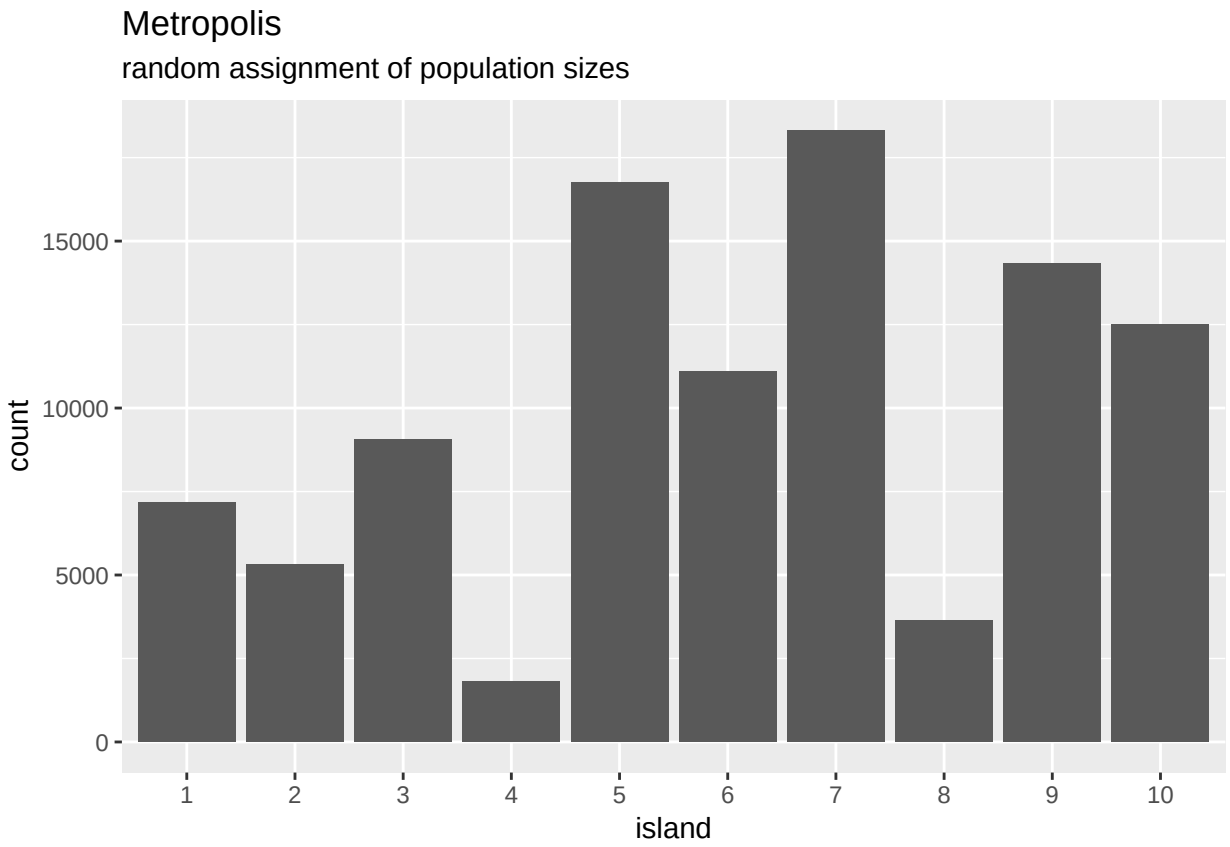
```
compare(h8.16, h8.17)
```

```
##        WAIC pWAIC dWAIC weight   SE  dSE
## h8.17 207.4   3.2     0    0.5 9.51   NA
## h8.16 207.4   3.5     0    0.5 9.49 1.52
```

The lower WAIC and higher weight assigned to the model with the truncated prior (h8.17) indicates that this model performs better than the model without the truncated prior (h8.16). The model with the truncated prior also has a lower number of effective parameters (pWAIC) than the model without the truncated prior. Because pWAIC is essentially the sum of the variance in the log-likelihood for all of the observations for each sample in the posterior distribution, truncating the left and right leg posterior distributions such that each is smaller than in the model with no truncation decreases this variance and therefore decreases the pWAIC value.

8H5

```
#set the number of weeks and make position placeholder
num_weeks <- 1e5
positions <- rep(0, num_weeks)
current <- 10
set.seed(100)

#make random distribution of population size
isl_num <- seq(1, 10, by = 1)
isl_pop <- sample(isl_num)

for(i in 1:num_weeks) {
  #record current position
  positions[i] <- current

  #flip coin to generate proposal
  proposal <- current + sample(c(-1, 1), size = 1)

  #deal with 1 -> 10 and 10 -> 1
  if(proposal < 1) proposal <- 10
  if(proposal > 10) proposal <- 1

  #CHOICE
  prob_move <- isl_pop[proposal]/isl_pop[current]
  current <- ifelse(runif(1) < prob_move, proposal, current)
}

tibble(week   = 1:1e5,
       island = positions) %>%
  mutate(island = factor(island)) %>%

  ggplot(aes(x = island)) +
  geom_bar() +
  labs(title = "Metropolis",
       subtitle = "random assignment of population sizes")
```

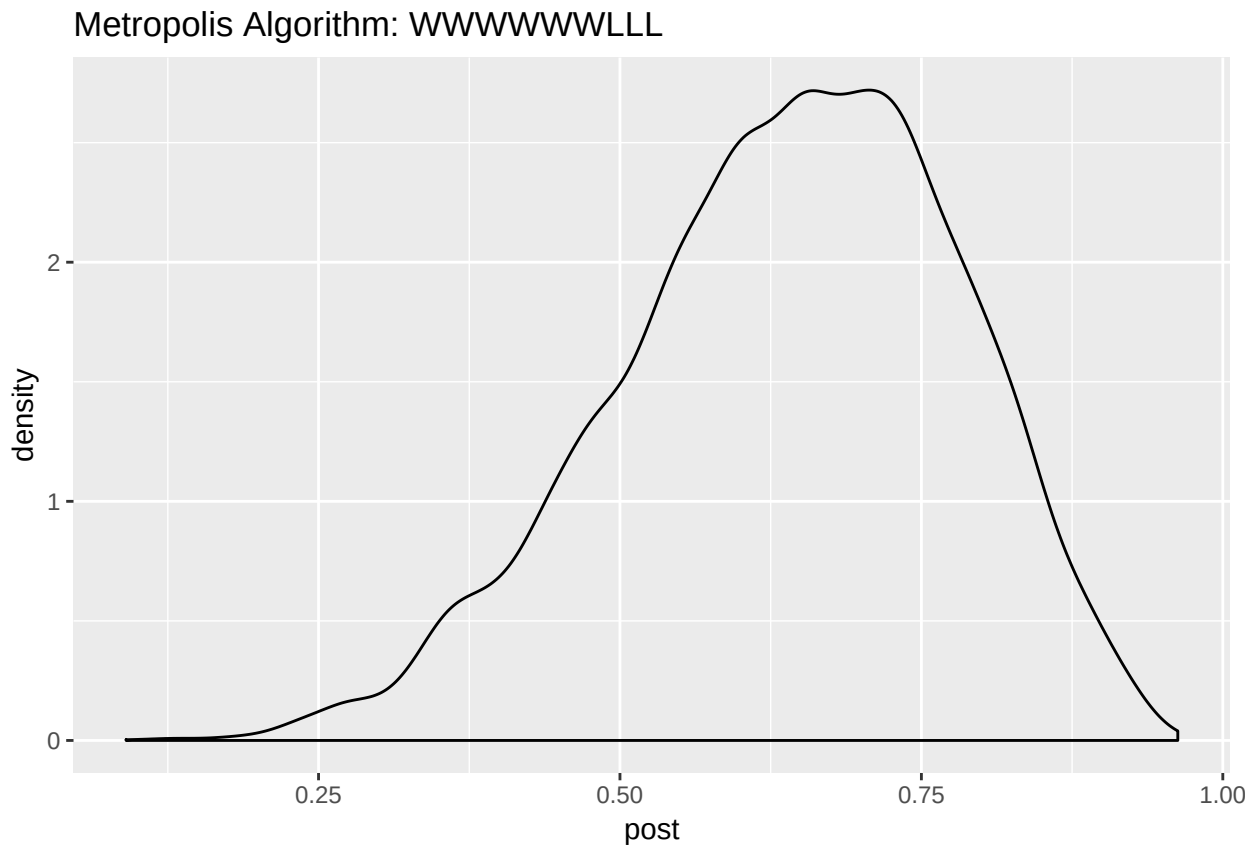


After randomly assigning the populations of the islands such that for islands 1-10, their respective populations are 4, 3, 5, 1, 9, 6, 10, 2, 8, 7, the modified algorithm results in the posterior distribution plotted above. The posterior distribution follows expectaions, in that the highest bar is island 7, which was assigned the population ranking of 10, and the lowest bar is at island 4, which was assigned the population ranking of 1. Furthermore, the other bars also follow the randomly assigned population ranking such that bar size matches relative population size.

H86

```
#set the number of weeks and make position placeholder
n <- 1e4
post <- rep(0, n)
current <- runif(1)

#data = WWWWWWLLL (6 water, 3 land)

for(i in 1:n) {
  #record current position
  post[i] <- current

  #get likelihood at current position
  l1 <- dbinom(6, 9, p = current)

  #flip coin to generate proposal
  proposal <- current + runif(1, min = -0.2, max = 0.2)
```

```
  #deal with <0 and >1
  if(proposal < 0) proposal <- 1 + proposal
  if(proposal > 1) proposal <- 0 + (proposal - 1)

  l2 <- dbinom(6, 9, p = proposal)

  #CHOICE
  prob_move <- l2/l1
  current <- ifelse(runif(1) < prob_move, proposal, current)
}

post <- as.data.frame(post)
ggplot(data = post, aes(x = post)) + geom_density() +
  ggtitle("Metropolis Algorithm: WWWWWWLLL")
```



Modifying the Metropolis algorithm to create a simple MCMC estimator for the globe tossing data in which we observed 6 water observations in 9 tosses results in a posterior distribution centered around 0.6386198, which is about 6/9. In this example, instead of having a population ranking to use to evaluate the likelihood of moving islands, the likelihood of the current position (corresponding to a possible proportion of water on Earth) was calculated by finding the probability that we would have observed 6 out of 9 tosses had the proportion of water been equal to the current position. After randomly selecting a new possible proportion a little bit higher or lower than the current one, the probability that we would have observed 6 out of 9 tosses had the proportion of water been equal to the proposal proportion was calculated. If the proposal probability was higher than the current probability, the proposal was selected and the process repeated. If the proposal probability was lower, then the probability the proposal is selected is equal to the proposal probability divided by the current probability. If the proposal is not selected, the process repeats with the current proportion.