Delayed effect of multiple human feedback messages

Nele Albers

12 August, 2024

Contents

Introduction **1**

Setup **1**

Data file **1**

Fit model for effort **2**

Fit model for return likelihood **8**

Introduction

Here we fit multilevel models to assess the delayed impact of sending multiple human feedback messages on the effort people spend on their preparatory activities and the likelihood that they would have returned to the next session if it was part of an unpaid intervention. Specifically, we include a factor that captures if a person has received feedback so far and a factor that captures how many times a person has received feedback.

Required files:

- Data/data_rl_samples.csv

Setup

First, we load the rethinking package, which we need to fit and sample from models. We also load formatR for formatting.

```
library(formatR)  # For formatting
library(rethinking)  # For Bayesian models
```

Also, we set the number of chains used for fitting the models.

```
NUM_CHAINS = 4  # our value: 4
```

Data file

We load the pre-processed data.

```
df = read.csv(file = "Data/data_rl_samples.csv")
df$Prev_Feedback_Binary = as.integer(df$Prev_Feedback_Count > 0)
```

Fit model for effort

First we fit a model for the effort.

```
# Create a data list to be used for the model
dat_list_effort <- list(
  effort = df$effort,
  humansupport = df$a,
  id = df$cons_id + 1,  # needs to start at 1
  prev_humansupport = df$Prev_Feedback_Binary,
  prev_humansupport_mult = df$Prev_Feedback_Count
)

set.seed(18)

ml.effort <- ulam(
  alist(
    effort ~ dstudent(v, mu, sigma),
    mu <- a_bar + z[id] * sigma_a + b_prevhs * prev_humansupport + b_prevhsm * prev_humansupport_mult,
    v ~ gamma(2,0.1),
    z[id] ~ dnorm(0, 1),
    sigma_a ~ dexp(1),
    a_bar ~ dnorm(5, 10),
    sigma ~ dexp(1),
    b_prevhs ~ dnorm(0, 10),
    b_prevhsm ~ dnorm(0, 10)
  ), data = dat_list_effort, chains=NUM_CHAINS, log_lik = TRUE, cores=NUM_CHAINS, iter = 3000
)
```

```
## Running MCMC with 4 parallel chains, with 1 thread(s) per chain...
##
## Chain 1 Iteration:    1 / 3000 [  0%]  (Warmup)

## Chain 1 Informational Message: The current Metropolis proposal is about to be rejected because of the

## Chain 1 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9y

## Chain 1 If this warning occurs sporadically, such as for highly constrained variable types like cova

## Chain 1 but if this warning occurs often then your model may be either severely ill-conditioned or m

## Chain 1

## Chain 1 Informational Message: The current Metropolis proposal is about to be rejected because of the

## Chain 1 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9y

## Chain 1 If this warning occurs sporadically, such as for highly constrained variable types like cova

## Chain 1 but if this warning occurs often then your model may be either severely ill-conditioned or m

## Chain 1

## Chain 1 Informational Message: The current Metropolis proposal is about to be rejected because of the

## Chain 1 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9y

## Chain 1 If this warning occurs sporadically, such as for highly constrained variable types like cova

## Chain 1 but if this warning occurs often then your model may be either severely ill-conditioned or m

## Chain 1
```

```
## Chain 2 Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 2 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 2 Exception: gamma_lpdf: Random variable is inf, but must be positive finite! (in '/tmp/Rtmp6N9
## Chain 2 If this warning occurs sporadically, such as for highly constrained variable types like covar
## Chain 2 but if this warning occurs often then your model may be either severely ill-conditioned or mi
## Chain 2
## Chain 2 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 2 Exception: gamma_lpdf: Random variable is inf, but must be positive finite! (in '/tmp/Rtmp6N9
## Chain 2 If this warning occurs sporadically, such as for highly constrained variable types like covar
## Chain 2 but if this warning occurs often then your model may be either severely ill-conditioned or mi
## Chain 2
## Chain 2 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 2 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9yb
## Chain 2 If this warning occurs sporadically, such as for highly constrained variable types like covar
## Chain 2 but if this warning occurs often then your model may be either severely ill-conditioned or mi
## Chain 2
## Chain 2 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 2 Exception: student_t_lpdf: Scale parameter is inf, but must be positive finite! (in '/tmp/Rtm
## Chain 2 If this warning occurs sporadically, such as for highly constrained variable types like covar
## Chain 2 but if this warning occurs often then your model may be either severely ill-conditioned or mi
## Chain 2
## Chain 3 Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 3 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 3 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9yb
## Chain 3 If this warning occurs sporadically, such as for highly constrained variable types like covar
## Chain 3 but if this warning occurs often then your model may be either severely ill-conditioned or mi
## Chain 3
## Chain 3 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 3 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9yb
## Chain 3 If this warning occurs sporadically, such as for highly constrained variable types like covar
## Chain 3 but if this warning occurs often then your model may be either severely ill-conditioned or mi
## Chain 3
## Chain 3 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 3 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9yb
## Chain 3 If this warning occurs sporadically, such as for highly constrained variable types like covar
## Chain 3 but if this warning occurs often then your model may be either severely ill-conditioned or mi
```

```
## Chain 3
## Chain 3 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 3 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9y
## Chain 3 If this warning occurs sporadically, such as for highly constrained variable types like cova
## Chain 3 but if this warning occurs often then your model may be either severely ill-conditioned or m
## Chain 3
## Chain 3 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 3 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9y
## Chain 3 If this warning occurs sporadically, such as for highly constrained variable types like cova
## Chain 3 but if this warning occurs often then your model may be either severely ill-conditioned or m
## Chain 3
## Chain 4 Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 4 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 4 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9y
## Chain 4 If this warning occurs sporadically, such as for highly constrained variable types like cova
## Chain 4 but if this warning occurs often then your model may be either severely ill-conditioned or m
## Chain 4
## Chain 4 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 4 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9y
## Chain 4 If this warning occurs sporadically, such as for highly constrained variable types like cova
## Chain 4 but if this warning occurs often then your model may be either severely ill-conditioned or m
## Chain 4
## Chain 4 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 4 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9y
## Chain 4 If this warning occurs sporadically, such as for highly constrained variable types like cova
## Chain 4 but if this warning occurs often then your model may be either severely ill-conditioned or m
## Chain 4
## Chain 4 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 4 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9y
## Chain 4 If this warning occurs sporadically, such as for highly constrained variable types like cova
## Chain 4 but if this warning occurs often then your model may be either severely ill-conditioned or m
## Chain 4
## Chain 4 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 4 Exception: gamma_lpdf: Random variable is 0, but must be positive finite! (in '/tmp/Rtmp6N9y
## Chain 4 If this warning occurs sporadically, such as for highly constrained variable types like cova
## Chain 4 but if this warning occurs often then your model may be either severely ill-conditioned or m
```

```
## Chain 4
## Chain 1 Iteration:  100 / 3000 [  3%]  (Warmup)
## Chain 4 Iteration:  100 / 3000 [  3%]  (Warmup)
## Chain 3 Iteration:  100 / 3000 [  3%]  (Warmup)
## Chain 1 Iteration:  200 / 3000 [  6%]  (Warmup)
## Chain 2 Iteration:  100 / 3000 [  3%]  (Warmup)
## Chain 4 Iteration:  200 / 3000 [  6%]  (Warmup)
## Chain 1 Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 3 Iteration:  200 / 3000 [  6%]  (Warmup)
## Chain 2 Iteration:  200 / 3000 [  6%]  (Warmup)
## Chain 4 Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 1 Iteration:  400 / 3000 [ 13%]  (Warmup)
## Chain 3 Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 2 Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 4 Iteration:  400 / 3000 [ 13%]  (Warmup)
## Chain 1 Iteration:  500 / 3000 [ 16%]  (Warmup)
## Chain 3 Iteration:  400 / 3000 [ 13%]  (Warmup)
## Chain 2 Iteration:  400 / 3000 [ 13%]  (Warmup)
## Chain 4 Iteration:  500 / 3000 [ 16%]  (Warmup)
## Chain 1 Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 3 Iteration:  500 / 3000 [ 16%]  (Warmup)
## Chain 1 Iteration:  700 / 3000 [ 23%]  (Warmup)
## Chain 2 Iteration:  500 / 3000 [ 16%]  (Warmup)
## Chain 4 Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 3 Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 1 Iteration:  800 / 3000 [ 26%]  (Warmup)
## Chain 2 Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 4 Iteration:  700 / 3000 [ 23%]  (Warmup)
## Chain 3 Iteration:  700 / 3000 [ 23%]  (Warmup)
## Chain 1 Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 4 Iteration:  800 / 3000 [ 26%]  (Warmup)
## Chain 2 Iteration:  700 / 3000 [ 23%]  (Warmup)
## Chain 3 Iteration:  800 / 3000 [ 26%]  (Warmup)
## Chain 1 Iteration: 1000 / 3000 [ 33%]  (Warmup)
## Chain 4 Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 2 Iteration:  800 / 3000 [ 26%]  (Warmup)
## Chain 3 Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 4 Iteration: 1000 / 3000 [ 33%]  (Warmup)
## Chain 1 Iteration: 1100 / 3000 [ 36%]  (Warmup)
## Chain 2 Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 3 Iteration: 1000 / 3000 [ 33%]  (Warmup)
## Chain 1 Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 4 Iteration: 1100 / 3000 [ 36%]  (Warmup)
## Chain 2 Iteration: 1000 / 3000 [ 33%]  (Warmup)
## Chain 3 Iteration: 1100 / 3000 [ 36%]  (Warmup)
## Chain 1 Iteration: 1300 / 3000 [ 43%]  (Warmup)
## Chain 4 Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 2 Iteration: 1100 / 3000 [ 36%]  (Warmup)
## Chain 3 Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 1 Iteration: 1400 / 3000 [ 46%]  (Warmup)
## Chain 4 Iteration: 1300 / 3000 [ 43%]  (Warmup)
## Chain 2 Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 3 Iteration: 1300 / 3000 [ 43%]  (Warmup)
```

```
## Chain 1 Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 1 Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 4 Iteration: 1400 / 3000 [ 46%]  (Warmup)
## Chain 2 Iteration: 1300 / 3000 [ 43%]  (Warmup)
## Chain 3 Iteration: 1400 / 3000 [ 46%]  (Warmup)
## Chain 1 Iteration: 1600 / 3000 [ 53%]  (Sampling)
## Chain 2 Iteration: 1400 / 3000 [ 46%]  (Warmup)
## Chain 4 Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 4 Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 3 Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 3 Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 1 Iteration: 1700 / 3000 [ 56%]  (Sampling)
## Chain 2 Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 2 Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 4 Iteration: 1600 / 3000 [ 53%]  (Sampling)
## Chain 3 Iteration: 1600 / 3000 [ 53%]  (Sampling)
## Chain 1 Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 2 Iteration: 1600 / 3000 [ 53%]  (Sampling)
## Chain 4 Iteration: 1700 / 3000 [ 56%]  (Sampling)
## Chain 3 Iteration: 1700 / 3000 [ 56%]  (Sampling)
## Chain 1 Iteration: 1900 / 3000 [ 63%]  (Sampling)
## Chain 2 Iteration: 1700 / 3000 [ 56%]  (Sampling)
## Chain 4 Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 3 Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 1 Iteration: 2000 / 3000 [ 66%]  (Sampling)
## Chain 2 Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 4 Iteration: 1900 / 3000 [ 63%]  (Sampling)
## Chain 3 Iteration: 1900 / 3000 [ 63%]  (Sampling)
## Chain 1 Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 2 Iteration: 1900 / 3000 [ 63%]  (Sampling)
## Chain 4 Iteration: 2000 / 3000 [ 66%]  (Sampling)
## Chain 3 Iteration: 2000 / 3000 [ 66%]  (Sampling)
## Chain 1 Iteration: 2200 / 3000 [ 73%]  (Sampling)
## Chain 2 Iteration: 2000 / 3000 [ 66%]  (Sampling)
## Chain 4 Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 3 Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 1 Iteration: 2300 / 3000 [ 76%]  (Sampling)
## Chain 2 Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 4 Iteration: 2200 / 3000 [ 73%]  (Sampling)
## Chain 3 Iteration: 2200 / 3000 [ 73%]  (Sampling)
## Chain 1 Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 2 Iteration: 2200 / 3000 [ 73%]  (Sampling)
## Chain 4 Iteration: 2300 / 3000 [ 76%]  (Sampling)
## Chain 3 Iteration: 2300 / 3000 [ 76%]  (Sampling)
## Chain 1 Iteration: 2500 / 3000 [ 83%]  (Sampling)
## Chain 2 Iteration: 2300 / 3000 [ 76%]  (Sampling)
## Chain 4 Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 3 Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 1 Iteration: 2600 / 3000 [ 86%]  (Sampling)
## Chain 2 Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 4 Iteration: 2500 / 3000 [ 83%]  (Sampling)
## Chain 3 Iteration: 2500 / 3000 [ 83%]  (Sampling)
## Chain 1 Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 2 Iteration: 2500 / 3000 [ 83%]  (Sampling)
```

```
## Chain 4 Iteration: 2600 / 3000 [ 86%]  (Sampling)
## Chain 3 Iteration: 2600 / 3000 [ 86%]  (Sampling)
## Chain 1 Iteration: 2800 / 3000 [ 93%]  (Sampling)
## Chain 4 Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 2 Iteration: 2600 / 3000 [ 86%]  (Sampling)
## Chain 1 Iteration: 2900 / 3000 [ 96%]  (Sampling)
## Chain 3 Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 4 Iteration: 2800 / 3000 [ 93%]  (Sampling)
## Chain 2 Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 1 Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 1 finished in 34.4 seconds.
## Chain 3 Iteration: 2800 / 3000 [ 93%]  (Sampling)
## Chain 4 Iteration: 2900 / 3000 [ 96%]  (Sampling)
## Chain 2 Iteration: 2800 / 3000 [ 93%]  (Sampling)
## Chain 3 Iteration: 2900 / 3000 [ 96%]  (Sampling)
## Chain 4 Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 4 finished in 35.5 seconds.
## Chain 2 Iteration: 2900 / 3000 [ 96%]  (Sampling)
## Chain 3 Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 3 finished in 36.1 seconds.
## Chain 2 Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 2 finished in 36.3 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 35.6 seconds.
## Total execution time: 36.5 seconds.
```

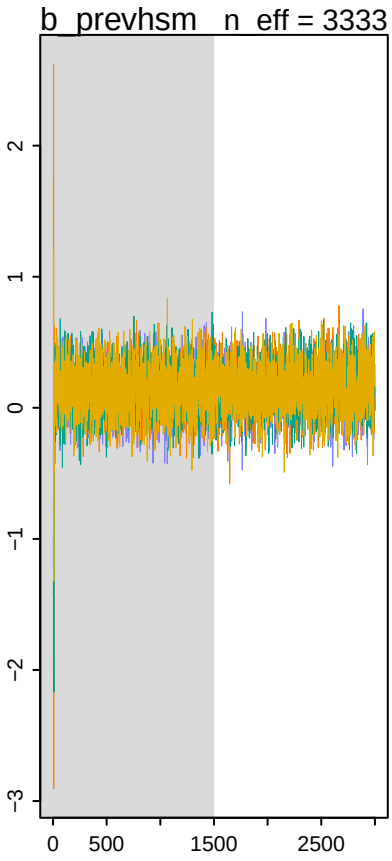
```
output_effort = precis(ml.effort, prob = 0.95)
```

```
## 679 vector or matrix parameters hidden. Use depth=2 to show them.
```

```
output_effort
```

```
##                mean         sd       2.5%     97.5%      rhat ess_bulk
## v         4.1515900 0.40929274  3.4198325 5.0388605 0.9998222 3372.047
## sigma_a   1.9834494 0.07037274  1.8496150 2.1306603 1.0012782 1752.579
## a_bar     5.6749766 0.09999863  5.4758398 5.8649170 1.0042487 1501.550
## sigma     1.5602106 0.05189254  1.4611780 1.6622812 1.0001640 2641.097
## b_prevhs  0.2302264 0.23510199 -0.2361153 0.6885817 1.0004848 3141.675
## b_prevhsm 0.1441044 0.17700566 -0.2057524 0.4960004 1.0002102 3333.044
```

```
traceplot(ml.effort, pars = c("b_prevhsm"))
```



Let's compute the posterior probability that `b_prevhsm` is greater than 0.

```
set.seed(18)  # For reproducibility
samples.ml.effort <- extract.samples(ml.effort)
Heffort_post <- samples.ml.effort$b_prevhsm[which(samples.ml.effort$b_prevhsm >
    0)]
Heffort_post_p <- round(length(Heffort_post)/length(samples.ml.effort$b_prevhsm),
    5)
Heffort_post_p
```

```
## [1] 0.7955
```

And let's also compute an effect size.

```
b_prevhsm = output_effort$mean[6]
sd = output_effort$mean[4]
effect_size = b_prevhsm/sd
round(effect_size, 2)
```

```
## [1] 0.09
```

```
rm(ml.effort, output_effort, samples.ml.effort)
```

Fit model for return likelihood

And we also fit a model for the return likelihood.

```
# Create a data list to be used for the model
dat_list_dropout <- list(
```

```
  dropout = df$dropout_response,
  humansupport = df$a,
  id = df$cons_id + 1,  # needs to start at 1
  prev_humansupport = df$Prev_Feedback_Binary,
  prev_humansupport_mult = df$Prev_Feedback_Count
)

set.seed(18)

ml.dropout <- ulam(
  alist(
    dropout ~ dnorm(mu, sigma),
    mu <- a_bar + z[id] + b_prevhs * prev_humansupport + b_prevhsm * prev_humansupport_mult,
    z[id] ~ dnorm(0, sigma_a),
    sigma_a ~ dexp(1),
    a_bar ~ dnorm(0, 10),
    sigma ~ dexp(1),
    b_prevhs ~ dnorm(0, 10),
    b_prevhsm ~ dnorm(0, 10)
  ), data = dat_list_dropout, chains=NUM_CHAINS, log_lik = TRUE, cores=NUM_CHAINS, iter = 3000, control=
)
```

```
## Running MCMC with 4 parallel chains, with 1 thread(s) per chain...
##
## Chain 1 Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 1 Informational Message: The current Metropolis proposal is about to be rejected because of the
## Chain 1 Exception: normal_lpdf: Scale parameter is 0, but must be positive! (in '/tmp/Rtmp6N9yDH/model
## Chain 1 If this warning occurs sporadically, such as for highly constrained variable types like covar
## Chain 1 but if this warning occurs often then your model may be either severely ill-conditioned or mi
## Chain 1
## Chain 2 Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 3 Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 4 Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 1 Iteration:  100 / 3000 [  3%]  (Warmup)
## Chain 3 Iteration:  100 / 3000 [  3%]  (Warmup)
## Chain 4 Iteration:  100 / 3000 [  3%]  (Warmup)
## Chain 2 Iteration:  100 / 3000 [  3%]  (Warmup)
## Chain 2 Iteration:  200 / 3000 [  6%]  (Warmup)
## Chain 1 Iteration:  200 / 3000 [  6%]  (Warmup)
## Chain 4 Iteration:  200 / 3000 [  6%]  (Warmup)
## Chain 2 Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 4 Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 2 Iteration:  400 / 3000 [ 13%]  (Warmup)
## Chain 1 Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 3 Iteration:  200 / 3000 [  6%]  (Warmup)
## Chain 4 Iteration:  400 / 3000 [ 13%]  (Warmup)
## Chain 1 Iteration:  400 / 3000 [ 13%]  (Warmup)
## Chain 2 Iteration:  500 / 3000 [ 16%]  (Warmup)
## Chain 1 Iteration:  500 / 3000 [ 16%]  (Warmup)
## Chain 2 Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 4 Iteration:  500 / 3000 [ 16%]  (Warmup)
```

```
## Chain 3 Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 1 Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 4 Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 2 Iteration:  700 / 3000 [ 23%]  (Warmup)
## Chain 1 Iteration:  700 / 3000 [ 23%]  (Warmup)
## Chain 3 Iteration:  400 / 3000 [ 13%]  (Warmup)
## Chain 4 Iteration:  700 / 3000 [ 23%]  (Warmup)
## Chain 1 Iteration:  800 / 3000 [ 26%]  (Warmup)
## Chain 2 Iteration:  800 / 3000 [ 26%]  (Warmup)
## Chain 3 Iteration:  500 / 3000 [ 16%]  (Warmup)
## Chain 4 Iteration:  800 / 3000 [ 26%]  (Warmup)
## Chain 3 Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 1 Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 2 Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 4 Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 3 Iteration:  700 / 3000 [ 23%]  (Warmup)
## Chain 4 Iteration: 1000 / 3000 [ 33%]  (Warmup)
## Chain 1 Iteration: 1000 / 3000 [ 33%]  (Warmup)
## Chain 2 Iteration: 1000 / 3000 [ 33%]  (Warmup)
## Chain 3 Iteration:  800 / 3000 [ 26%]  (Warmup)
## Chain 4 Iteration: 1100 / 3000 [ 36%]  (Warmup)
## Chain 2 Iteration: 1100 / 3000 [ 36%]  (Warmup)
## Chain 1 Iteration: 1100 / 3000 [ 36%]  (Warmup)
## Chain 3 Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 4 Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 2 Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 1 Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 3 Iteration: 1000 / 3000 [ 33%]  (Warmup)
## Chain 4 Iteration: 1300 / 3000 [ 43%]  (Warmup)
## Chain 2 Iteration: 1300 / 3000 [ 43%]  (Warmup)
## Chain 1 Iteration: 1300 / 3000 [ 43%]  (Warmup)
## Chain 3 Iteration: 1100 / 3000 [ 36%]  (Warmup)
## Chain 4 Iteration: 1400 / 3000 [ 46%]  (Warmup)
## Chain 2 Iteration: 1400 / 3000 [ 46%]  (Warmup)
## Chain 1 Iteration: 1400 / 3000 [ 46%]  (Warmup)
## Chain 3 Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 4 Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 4 Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 2 Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 2 Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 3 Iteration: 1300 / 3000 [ 43%]  (Warmup)
## Chain 4 Iteration: 1600 / 3000 [ 53%]  (Sampling)
## Chain 1 Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 1 Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 2 Iteration: 1600 / 3000 [ 53%]  (Sampling)
## Chain 4 Iteration: 1700 / 3000 [ 56%]  (Sampling)
## Chain 3 Iteration: 1400 / 3000 [ 46%]  (Warmup)
## Chain 1 Iteration: 1600 / 3000 [ 53%]  (Sampling)
## Chain 2 Iteration: 1700 / 3000 [ 56%]  (Sampling)
## Chain 4 Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 3 Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 3 Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 1 Iteration: 1700 / 3000 [ 56%]  (Sampling)
## Chain 3 Iteration: 1600 / 3000 [ 53%]  (Sampling)
```

```
## Chain 4 Iteration: 1900 / 3000 [ 63%]  (Sampling)
## Chain 2 Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 3 Iteration: 1700 / 3000 [ 56%]  (Sampling)
## Chain 1 Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 4 Iteration: 2000 / 3000 [ 66%]  (Sampling)
## Chain 3 Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 2 Iteration: 1900 / 3000 [ 63%]  (Sampling)
## Chain 3 Iteration: 1900 / 3000 [ 63%]  (Sampling)
## Chain 1 Iteration: 1900 / 3000 [ 63%]  (Sampling)
## Chain 4 Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 2 Iteration: 2000 / 3000 [ 66%]  (Sampling)
## Chain 3 Iteration: 2000 / 3000 [ 66%]  (Sampling)
## Chain 3 Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 4 Iteration: 2200 / 3000 [ 73%]  (Sampling)
## Chain 1 Iteration: 2000 / 3000 [ 66%]  (Sampling)
## Chain 2 Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 3 Iteration: 2200 / 3000 [ 73%]  (Sampling)
## Chain 4 Iteration: 2300 / 3000 [ 76%]  (Sampling)
## Chain 1 Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 3 Iteration: 2300 / 3000 [ 76%]  (Sampling)
## Chain 2 Iteration: 2200 / 3000 [ 73%]  (Sampling)
## Chain 4 Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 3 Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 1 Iteration: 2200 / 3000 [ 73%]  (Sampling)
## Chain 3 Iteration: 2500 / 3000 [ 83%]  (Sampling)
## Chain 2 Iteration: 2300 / 3000 [ 76%]  (Sampling)
## Chain 4 Iteration: 2500 / 3000 [ 83%]  (Sampling)
## Chain 3 Iteration: 2600 / 3000 [ 86%]  (Sampling)
## Chain 1 Iteration: 2300 / 3000 [ 76%]  (Sampling)
## Chain 4 Iteration: 2600 / 3000 [ 86%]  (Sampling)
## Chain 3 Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 2 Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 3 Iteration: 2800 / 3000 [ 93%]  (Sampling)
## Chain 1 Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 4 Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 2 Iteration: 2500 / 3000 [ 83%]  (Sampling)
## Chain 3 Iteration: 2900 / 3000 [ 96%]  (Sampling)
## Chain 4 Iteration: 2800 / 3000 [ 93%]  (Sampling)
## Chain 3 Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 3 finished in 77.7 seconds.
## Chain 1 Iteration: 2500 / 3000 [ 83%]  (Sampling)
## Chain 2 Iteration: 2600 / 3000 [ 86%]  (Sampling)
## Chain 4 Iteration: 2900 / 3000 [ 96%]  (Sampling)
## Chain 1 Iteration: 2600 / 3000 [ 86%]  (Sampling)
## Chain 2 Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 4 Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 4 finished in 81.5 seconds.
## Chain 1 Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 2 Iteration: 2800 / 3000 [ 93%]  (Sampling)
## Chain 1 Iteration: 2800 / 3000 [ 93%]  (Sampling)
## Chain 2 Iteration: 2900 / 3000 [ 96%]  (Sampling)
## Chain 1 Iteration: 2900 / 3000 [ 96%]  (Sampling)
## Chain 2 Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 2 finished in 87.1 seconds.
```

```
## Chain 1 Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 1 finished in 88.2 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 83.6 seconds.
## Total execution time: 88.4 seconds.
```

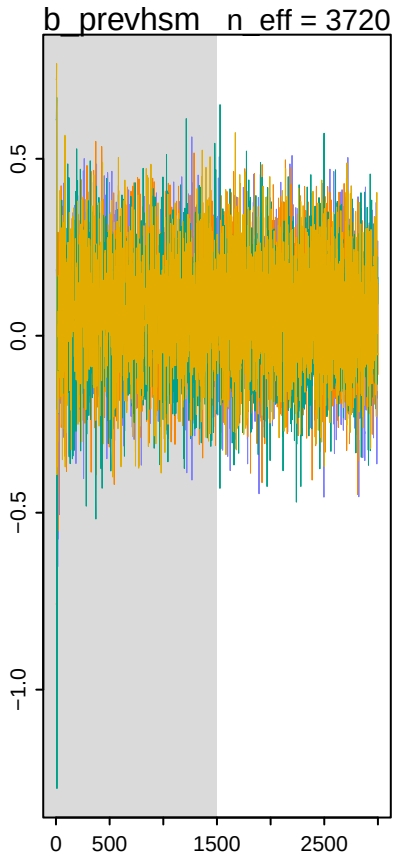
```
output_dropout = precis(ml.dropout, prob = 0.95)
```

```
## 679 vector or matrix parameters hidden. Use depth=2 to show them.
```

```
output_dropout
```

```
##                  mean         sd       2.5%     97.5%     rhat ess_bulk
## sigma_a   2.19609999 0.07106392  2.0636968 2.3410802 1.000021 6156.060
## a_bar     1.65864078 0.09917930  1.4673092 1.8522795 1.001561 1278.129
## sigma     1.59157832 0.02756146  1.5391598 1.6475207 1.000448 4900.747
## b_prevhs  0.21745681 0.20235133 -0.1739553 0.6168928 1.000718 3541.077
## b_prevhsm 0.06156302 0.15563550 -0.2434849 0.3661082 1.000374 3720.357
```

```
traceplot(ml.dropout, pars = c("b_prevhsm"))
```



Let's compute the posterior probability that `b_prevhsm` is greater than 0.

```
set.seed(18)  # For reproducibility
samples.ml.dropout <- extract.samples(ml.dropout)
print(paste0("Number of samples extracted: ", length(samples.ml.dropout$b_prevhsm)))
```

```
## [1] "Number of samples extracted: 6000"
```

```
Hdrop_post <- samples.ml.dropout$b_prevhsm[which(samples.ml.dropout$b_prevhsm >
    0)]
Hdrop_post_p <- round(length(Hdrop_post)/length(samples.ml.dropout$b_prevhsm),
    5)
Hdrop_post_p
```

```
## [1] 0.656
```

And let's also compute an effect size.

```
b_prevhsm = output_dropout$mean[5]
sd = output_dropout$mean[3]
effect_size = b_prevhsm/sd
round(effect_size, 2)
```

```
## [1] 0.04
```