Week 7 Exercises

Exercise 1 `brm()`

Reproduce the familiar negative binomial regression below with `brm()`.

Hints:

1. See `?brms::brmsfamily` for the distributions available in `brm()`.
2. Use `#| results: hide` in your Quarto code chunks to hide the Stan junk output. It's pesky.

```
# load packages
library(glmmTMB)

# load data
holland <- crdata::holland2015

# formula corresponds to model 1 for each city in holland (2015) table 2
f <- operations ~ lower + vendors + budget + population

# fit poisson regression model for Santiago
fit <- glmmTMB(
  f,
  family = nbinom2,
  data = holland,
  subset = city == "santiago"
  )

# summary
summary(fit)
```

Solution

```
# load packages
library(brms)

# fit poisson regression model for Santiago
fit_brm <- brm(
  f,
  family = negbinomial,
  data = subset(holland, city == "santiago")  # brm() doesn't have subset
  )
```

```
# summary
summary(fit_brm)
```

```
 Family: negbinomial
  Links: mu = log
Formula: operations ~ lower + vendors + budget + population
   Data: subset(holland, city == "santiago") (Number of observations: 34)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Regression Coefficients:
           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept      2.79      2.11    -1.06     7.22 1.00     2308     2059
lower         -0.05      0.03    -0.12     0.00 1.00     2285     2053
vendors       -0.23      0.32    -0.86     0.42 1.00     1918     1985
budget         0.00      0.00    -0.00     0.01 1.00     2537     2164
population     0.04      0.05    -0.04     0.15 1.00     1847     1939

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
shape     0.31      0.12     0.14     0.59 1.00     1829     1721

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

Exercise 2 Convergence

Using the output above, argue that `warmup` and `iter` are sufficiently large (or not).

Solution.

- **Setup recap**
  4 chains; `iter = 2000`; `warmup = 1000`; *post-warmup = 1000 per chain* $\rightarrow$ **4000 total draws**.
- $\hat{R} \rightarrow$ **all parameters near 1.00**
  Why it matters: $\hat{R} \approx 1.00$ indicates chains mix to the same target after adaptation.
  Evidence: Intercept, `lower`, `vendors`, `budget`, `population`, and `shape` all show `Rhat = 1.00`, consistent with well-tuned warmup and sufficient iterations for convergence.
- **Effective sample sizes $\rightarrow$ large**
  Why it matters: `Bulk_ESS` gauges precision of means; `Tail_ESS` gauges interval tails. Rules of thumb: ESS 2000 per parameter. *Evidence:* `Bulk_ESS` and `Tail_ESS` are about 1900 to 2800 across parameters (e.g., `lower`: `Bulk_ESS` 2743; `shape`: 2240),

---

Exercise 3 `brm()` arguments

Explain the following arguments to `brm()`. Explain what the argument controls and recommend reasonable defaults.

- `chains`
- `iter`
- `warmup`
- `cores`
- `backend`

Solution.

- **`chains`**
  What it controls: Number of independent MCMC chains to run.
  Why it matters: Multiple chains start from different initial values, helping check convergence (e.g., via $\hat{R}$).
  Reasonable default: `chains = 4`. Fewer if slow, more if diagnostics are uncertain.

- **`iter`**
  What it controls: Total iterations per chain, including warmup.
  Why it matters: More iterations yield more effective posterior draws and reduce Monte Carlo error.
  Reasonable default: `iter = 2000` (with half used for warmup). Use 4000–8000 for complex models.

- **`warmup`**
  What it controls: Number of iterations per chain used for tuning (e.g., step size, mass matrix) and discarded afterward.
  Why it matters: Proper adaptation improves sampling efficiency and stability.
  Reasonable default: `warmup = 1000` (half of `iter`).

- **`cores`**
  What it controls: Number of CPU cores used to run chains in parallel.
  Why it matters: Parallel chains shorten total runtime.
  Reasonable default: `cores = chains`, often with `options(mc.cores = parallel::detectCores())`.

- **`backend`**
  What it controls: The Stan interface used for sampling.
  Options: `"cmdstanr"` or `"rstan"`.
  Why it matters: Determines compilation speed, diagnostics, and toolchain.
  Reasonable default: `"cmdstanr"` (if installed); otherwise defaults to `"rstan"`.

Example

```
fit <- brm(
  y ~ x1 + x2,
  data = df,
  family = gaussian(),
  chains = 4,
  iter = 4000,
  warmup = 2000,
  cores = 4,
  backend = "cmdstanr"
)
```

Exercise 4 {rstan} or {cmdstanr}

Reproduce the fits from Exercise 1 using a `.stan` model via {rstan} or {cmdstanr}.

Hint: You might want to use `neg_binomial_2_log`–see the Stan docs.

Solution

First, we can create the data objects and list that {rstan} needs. You could also use {cmdstanr}–the logic is similar.

```
# load pacakges
library(rstan)

# create data object for stan
holland <- crdata::holland2015
sant <- subset(holland, city == "santiago")

# formula to create design matrix
operations ~ lower + vendors + budget + population

# data for stan
mf <- model.frame(f, data = sant)
X <- model.matrix(f, data = mf)
y <- model.response(mf)
N <- nrow(X)
K <- ncol(X)

# combine data for stan into single list
stan_data <- list(N = N, K = K, X = X, y = y)
```

Now we create the `.stan` file. You can write this in a separate file, but I'm including it as a character string in this document.

```
stan_code <- "
data {
  int<lower=1> N;
  int<lower=0> y[N];
  int<lower=1> K;
  matrix[N, K] X;
```

```
}
parameters {
  vector[K] beta;           // regression coefficients (includes intercept)
  real<lower=0> theta;        // shape/overdispersion
}
model {
  // weakly informative priors
  beta ~ normal(0, 5);
  theta  ~ exponential(1);    // favors moderate overdispersion, >0

  // likelihood with log link
  y ~ neg_binomial_2_log(X * beta, theta);
}
"
```

```
fit <- stan(model_code = stan_code,
            data = stan_data,
            chains = 10,
            cores = 10,
            warmup = 2000,
            iter = 4000)
```

```
print(fit)
```

```
Inference for Stan model: anon_model.
10 chains, each with iter=4000; warmup=2000; thin=1;
post-warmup draws per chain=2000, total post-warmup draws=20000.

          mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
beta[1]   2.37    0.02 1.94  -1.34   1.11   2.31   3.60   6.35  6982    1
beta[2]  -0.05    0.00 0.03  -0.11  -0.07  -0.05  -0.03   0.01  8040    1
beta[3]  -0.24    0.00 0.33  -0.89  -0.44  -0.24  -0.04   0.42  7290    1
beta[4]   0.00    0.00 0.00   0.00   0.00   0.00   0.00   0.01  8082    1
beta[5]   0.04    0.00 0.05  -0.04   0.01   0.04   0.07   0.16  7028    1
theta     0.30    0.00 0.12   0.13   0.22   0.28   0.37   0.59  7864    1
lp__    -66.08    0.03 2.14 -71.33 -67.23 -65.70 -64.50 -63.11  4913    1

Samples were drawn using NUTS(diag_e) at Wed Oct  8 21:23:27 2025.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

Exercise 5 Experimenting with difficult posterior

The posterior below is somewhat difficult. The samples generate slowly and the effective sample size is relatively low.

Stan is a excellent with default settings, but there are ways to improve efficiency even further for difficult problems.

Here's *one way* to think about efficiency: you want your effective *samples per second* (of your time) to be relatively large. Experiment with the several strategies to increase this measure of efficiency.

1. Rescale the predictors to have mean of zero and SD of one or 0.5. Or perhaps have a range from zero to one.
2. Increase the number of chains run in parallel.
3. Increase the number of iterations (and the warmp).
4. Increase the `adapt_delta` parameter. Search `?brm` for "adapt_delta" for more.

These models take a long time to fit, so just experiment and write about what happens. Don't try to fit five different difficult Stan models in a single Quarto document (like I did in the solutions).

```
# load packages
library(brms)
library(posterior)

# load data
hks <- crdata::hks2013

# fit negative binomial model
f <- osvAll ~ troopLag + policeLag + militaryobserversLag +
  brv_AllLag + osvAllLagDum + incomp + epduration +
  lntpop
fit_mcmc <- brm(
  f,
  data = hks,
  family = negbinomial,
  warmup = 5000,
  iter = 10000,
  chains = 4,
  cores = 4)
```

```
# print fit
fit_mcmc
```

```
 Family: negbinomial
  Links: mu = log
Formula: osvAll ~ troopLag + policeLag + militaryobserversLag + brv_AllLag + osvAllLagDum + incomp + epduration +
   Data: hks (Number of observations: 3746)
  Draws: 4 chains, each with iter = 10000; warmup = 5000; thin = 1;
         total post-warmup draws = 20000

Regression Coefficients:
                    Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
Intercept              -9.24      0.90   -11.01    -7.48 1.00    15859
troopLag               -0.53      0.07    -0.65    -0.38 1.00    16305
policeLag              -9.92      1.03   -11.91    -7.87 1.00    19495
militaryobserversLag   21.88      1.39    19.32    24.78 1.00    17140
brv_AllLag              0.00      0.00    -0.00     0.00 1.00    22348
osvAllLagDum            2.18      0.18     1.84     2.54 1.00    19066
incomp                  2.37      0.18     2.02     2.73 1.00    17325
epduration             -0.00      0.00    -0.00     0.00 1.00    26620
lntpop                  0.71      0.08     0.55     0.86 1.00    16803
                    Tail_ESS
Intercept              15196
troopLag               13494
policeLag              13354
militaryobserversLag   13133
brv_AllLag             12765
osvAllLagDum           14991
incomp                 14748
epduration             15954
lntpop                 15186

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
shape     0.06      0.00     0.05     0.06 1.00    23981    13413

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

How long it took.

We can grab the time (in seconds) it took to obtain the samples from the `fit` component of `fit_mcmc`.

```
# time (in seconds) it took to samples for each chain
times <- get_elapsed_time(fit_mcmc$fit)
times
```

```
        warmup sample
chain:1 24.911 16.318
chain:2 24.049 16.093
chain:3 25.260 14.937
chain:4 23.277 14.928
```

```
# find max of warmup + samples
t <- max(apply(times, 1, sum))
t
```

```
[1] 41.229
```

How many (effective) samples we generated.

We can compute the ESS using the `effective_samples()` function in {bayestestR} package. We need a way to think about the ESS overall, so I just averaged the bulk ESS across the parameters.

```
# load package; some helpful things here!
library(bayestestR)

# compute ess for each parameter
ess <- effective_sample(fit_mcmc)
ess
```

```
               Parameter   ESS ESS_tail
1            b_Intercept 15859    15196
2             b_troopLag 16305    13494
3            b_policeLag 19495    13354
4 b_militaryobserversLag 17140    13133
5           b_brv_AllLag 22348    12765
6         b_osvAllLagDum 19066    14991
7              b_incomp 17325    14748
8           b_epduration 26620    15954
9              b_lntpop 16803    15186
```

```
# compute the average ess for a single number
avg_ess <- mean(ess$ESS)
avg_ess
```

```
[1] 18995.67
```

Effective samples per second.

And then we can compute the ratio–effective samples per second.

```
avg_ess/t
```

```
[1] 460.7356
```

To make this easy, we can make a function to compute the effective samples per second.

```
sps <- function(fit) {
  times <- get_elapsed_time(fit$fit)
  t <- max(apply(times, 1, sum))
  ess <- effective_sample(fit)
  avg_ess <- mean(ess$ESS)
  list("sps" = avg_ess/t,
       "ess" = ess,
       "times" = times)
}
```

Solution.

Strategy 1: Rescale predictors.

```
rs <- \(x) arm::rescale(x) # alias

f_rs <- osvAll ~ rs(troopLag) + rs(policeLag) + rs(militaryobserversLag) +
  rs(brv_AllLag) + rs(osvAllLagDum) + rs(incomp) + rs(epduration) +
  rs(lntpop)

fit_mcmc_1 <- brm(
  f_rs,
  data = hks,
  family = negbinomial,
  warmup = 5000,
  iter = 10000,
  chains = 4,
  cores = 4)
```

```
print(fit_mcmc_1)
```

```
 Family: negbinomial
  Links: mu = log
Formula: osvAll ~ rs(troopLag) + rs(policeLag) + rs(militaryobserversLag) + rs(brv_AllLag) +
   Data: hks (Number of observations: 3746)
  Draws: 4 chains, each with iter = 10000; warmup = 5000; thin = 1;
         total post-warmup draws = 20000

Regression Coefficients:
                       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
Intercept                  1.85      0.07     1.71     1.99 1.00    28039
rstroopLag                -3.00      0.39    -3.72    -2.21 1.00    17783
rspoliceLag               -2.81      0.29    -3.37    -2.25 1.00    22015
rsmilitaryobserversLag     4.63      0.29     4.09     5.23 1.00    18673
rsbrv_AllLag               0.24      0.20    -0.08     0.69 1.00    24885
rsosvAllLagDum             2.18      0.18     1.84     2.54 1.00    22628
rsincomp                   2.37      0.18     2.01     2.73 1.00    19971
rsepduration              -0.06      0.16    -0.38     0.25 1.00    24508
rslntpop                   1.70      0.19     1.34     2.08 1.00    18671
                       Tail_ESS
Intercept                 15326
rstroopLag                14222
rspoliceLag               15343
rsmilitaryobserversLag    14388
rsbrv_AllLag              12599
rsosvAllLagDum            15126
rsincomp                  15961
rsepduration              15069
rslntpop                  16226

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
shape     0.06      0.00     0.05     0.06 1.00    30409    13525

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

```
sps(fit_mcmc_1)
```

```
$sps
```

```
[1] 731.5384

$ess
                  Parameter   ESS ESS_tail
1               b_Intercept 28039    15326
2             b_rstroopLag 17783    14222
3            b_rspoliceLag 22015    15343
4 b_rsmilitaryobserversLag 18673    14388
5           b_rsbrv_AllLag 24885    12599
6         b_rsosvAllLagDum 22628    15126
7               b_rsincomp 19971    15961
8           b_rsepduration 24508    15069
9               b_rslntpop 18671    16226

$times
        warmup sample
chain:1 13.817 15.459
chain:2 13.814 15.382
chain:3 14.073 15.875
chain:4 13.920 14.975
```

Strategy 2: Increase number of chains.

I have 12 cores on my computer, so I can easily run 10 chains in parallel rather than just 4.

```
fit_mcmc_2 <- brm(
  f,
  data = hks,
  family = negbinomial,
  warmup = 5000,
  iter = 10000,
  chains = 10,
  cores = 10)
```

```
print(fit_mcmc_2)
```

```
 Family: negbinomial
  Links: mu = log
Formula: osvAll ~ troopLag + policeLag + militaryobserversLag + brv_AllLag + osvAllLagDum + 
   Data: hks (Number of observations: 3746)
  Draws: 10 chains, each with iter = 10000; warmup = 5000; thin = 1;
         total post-warmup draws = 50000
```

```
Regression Coefficients:
                     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
Intercept               -9.25      0.90   -11.00    -7.49 1.00    38731
troopLag                -0.52      0.07    -0.65    -0.39 1.00    38221
policeLag               -9.91      1.03   -11.92    -7.87 1.00    48866
militaryobserversLag    21.86      1.39    19.29    24.76 1.00    39779
brv_AllLag               0.00      0.00    -0.00     0.00 1.00    55803
osvAllLagDum             2.18      0.18     1.84     2.54 1.00    49898
incomp                   2.37      0.18     2.01     2.73 1.00    43268
epduration              -0.00      0.00    -0.00     0.00 1.00    67050
lntpop                   0.71      0.08     0.55     0.86 1.00    41010
                     Tail_ESS
Intercept               37004
troopLag                33545
policeLag               36044
militaryobserversLag    32436
brv_AllLag              32597
osvAllLagDum            37203
incomp                  38170
epduration              38535
lntpop                  38777

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
shape     0.06      0.00     0.05     0.06 1.00    61988    34517

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

```
sps(fit_mcmc_2)
```

```
$sps
[1] 773.9212

$ess
               Parameter   ESS ESS_tail
1            b_Intercept 38731    37004
2             b_troopLag 38221    33545
3            b_policeLag 48866    36044
4 b_militaryobserversLag 39779    32436
```

```
5           b_brv_AllLag 55803    32597
6         b_osvAllLagDum 49898    37203
7              b_incomp 43268    38170
8           b_epduration 67050    38535
9              b_lntpop 41010    38777

$times
         warmup sample
chain:1  34.531 18.802
chain:2  27.380 18.519
chain:3  32.864 18.162
chain:4  30.934 21.190
chain:5  33.360 19.183
chain:6  37.766 18.798
chain:7  31.525 20.034
chain:8  45.077 15.599
chain:9  28.715 20.044
chain:10 33.433 19.729
```

Strategy 3: Increase number iterations.

```r
fit_mcmc_3 <- brm(
  f,
  data = hks,
  family = negbinomial,
  warmup = 25000, # up from 5000
  iter = 50000, # up from 10000
  chains = 4,
  cores = 4)
```

```r
print(fit_mcmc_3)
```

```
 Family: negbinomial
  Links: mu = log
Formula: osvAll ~ troopLag + policeLag + militaryobserversLag + brv_AllLag + osvAllLagDum + 
   Data: hks (Number of observations: 3746)
  Draws: 4 chains, each with iter = 50000; warmup = 25000; thin = 1;
         total post-warmup draws = 100000

Regression Coefficients:
                    Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
Intercept              -9.24      0.89   -11.00    -7.48 1.00    91071
```

```
troopLag                -0.52      0.07    -0.65    -0.38 1.00    92192
policeLag               -9.92      1.03   -11.92    -7.88 1.00   108164
militaryobserversLag    21.86      1.39    19.27    24.74 1.00    96274
brv_AllLag               0.00      0.00    -0.00     0.00 1.00   125038
osvAllLagDum             2.18      0.18     1.84     2.54 1.00   114213
incomp                   2.37      0.18     2.02     2.73 1.00   102982
epduration              -0.00      0.00    -0.00     0.00 1.00   127937
lntpop                   0.71      0.08     0.55     0.86 1.00    92940
                     Tail_ESS
Intercept               75765
troopLag                72243
policeLag               75024
militaryobserversLag    70195
brv_AllLag              62178
osvAllLagDum            76294
incomp                  80545
epduration              78556
lntpop                  80919

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
shape     0.06      0.00     0.05     0.06 1.00   141358    68553

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

```
sps(fit_mcmc_3)
```

```
$sps
[1] 686.4343

$ess
               Parameter    ESS ESS_tail
1            b_Intercept  91071    75765
2             b_troopLag  92192    72243
3            b_policeLag 108164    75024
4 b_militaryobserversLag  96274    70195
5           b_brv_AllLag 125038    62178
6         b_osvAllLagDum 114213    76294
7              b_incomp 102982    80545
8           b_epduration 127937    78556
```

```
9               b_lntpop  92940    80919

$times
        warmup sample
chain:1 77.060 72.825
chain:2 78.650 75.255
chain:3 74.990 77.939
chain:4 75.004 74.534
```

Strategy 4a: Increase `adapt_delta`.

```
fit_mcmc_4a <- brm(
  f,
  data = hks,
  family = negbinomial,
  warmup = 5000,
  iter = 10000,
  control = list(adapt_delta = 0.95),  # from default of 0.8
  chains = 4,
  cores = 4)
```

```
print(fit_mcmc_4a)
```

```
Warning: Parts of the model have not converged (some Rhats are > 1.05). Be
careful when analysing the results! We recommend running more iterations and/or
setting stronger priors.

Warning: There were 142 divergent transitions after warmup. Increasing
adapt_delta above 0.95 may help. See
http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup

 Family: negbinomial
  Links: mu = log
Formula: osvAll ~ troopLag + policeLag + militaryobserversLag + brv_AllLag + osvAllLagDum +
   Data: hks (Number of observations: 3746)
  Draws: 4 chains, each with iter = 10000; warmup = 5000; thin = 1;
         total post-warmup draws = 20000

Regression Coefficients:
                    Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
Intercept              -3.35     16.81   -10.89    74.40 1.59        7
```

```
troopLag                -0.29      0.45    -0.64     0.78 1.59        7
policeLag               -7.84      3.71   -11.79    -1.56 1.57        7
militaryobserversLag    16.23      9.86    -0.87    24.49 1.59        7
brv_AllLag               0.01      0.04    -0.00     0.19 1.17       16
osvAllLagDum             2.04      0.30     1.58     2.51 1.53        7
incomp                   2.09      0.52     1.09     2.71 1.59        7
epduration              -0.05      0.23    -1.18     0.00 1.59        7
lntpop                   0.46      0.43    -0.35     0.85 1.59        7
                     Tail_ESS
Intercept                  11
troopLag                   11
policeLag                  17
militaryobserversLag       11
brv_AllLag                 11
osvAllLagDum               12
incomp                     11
epduration                 11
lntpop                     12

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
shape     0.53      1.35     0.06     5.06 1.59        7       11

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

```
sps(fit_mcmc_4a)
```

```
$sps
[1] 0.01249941

$ess
               Parameter ESS ESS_tail
1            b_Intercept   7       11
2             b_troopLag   7       11
3            b_policeLag   7       17
4 b_militaryobserversLag   7       11
5           b_brv_AllLag  16       11
6         b_osvAllLagDum   7       12
7              b_incomp   7       11
8           b_epduration   7       11
```

```
9               b_lntpop   7       12

$times
        warmup  sample
chain:1 37.905  17.967
chain:2 38.103  17.417
chain:3 49.754 590.276
chain:4 36.801  21.631
```

Strategy 4b: Increase `adapt_delta` even more.

```r
fit_mcmc_4b <- brm(
  f,
  data = hks,
  family = negbinomial,
  warmup = 5000,
  iter = 10000,
  control = list(adapt_delta = 0.99),  # from default of 0.8
  chains = 4,
  cores = 4)
```

```r
print(fit_mcmc_4b)
```

```
 Family: negbinomial
  Links: mu = log
Formula: osvAll ~ troopLag + policeLag + militaryobserversLag + brv_AllLag + osvAllLagDum + i
   Data: hks (Number of observations: 3746)
  Draws: 4 chains, each with iter = 10000; warmup = 5000; thin = 1;
         total post-warmup draws = 20000

Regression Coefficients:
                    Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
Intercept              -9.23      0.90   -11.01    -7.47 1.00    14424
troopLag               -0.52      0.07    -0.65    -0.39 1.00    14702
policeLag              -9.92      1.03   -11.94    -7.89 1.00    17536
militaryobserversLag   21.86      1.39    19.30    24.72 1.00    14834
brv_AllLag              0.00      0.00    -0.00     0.00 1.00    23535
osvAllLagDum            2.18      0.18     1.84     2.54 1.00    17075
incomp                  2.37      0.18     2.01     2.73 1.00    15979
epduration             -0.00      0.00    -0.00     0.00 1.00    24789
lntpop                  0.70      0.08     0.55     0.86 1.00    15241
                    Tail_ESS
```

```
Intercept               14051
troopLag                13096
policeLag               14737
militaryobserversLag    14047
brv_AllLag              12601
osvAllLagDum            15297
incomp                  14870
epduration              14797
lntpop                  14638

Further Distributional Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
shape     0.06      0.00     0.05     0.06 1.00    21713    13665

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

```
sps(fit_mcmc_4b)
```

```
$sps
[1] 196.7845

$ess
                Parameter   ESS ESS_tail
1             b_Intercept 14424    14051
2              b_troopLag 14702    13096
3             b_policeLag 17536    14737
4 b_militaryobserversLag 14834    14047
5            b_brv_AllLag 23535    12601
6          b_osvAllLagDum 17075    15297
7                b_incomp 15979    14870
8            b_epduration 24789    14797
9                b_lntpop 15241    14638

$times
        warmup sample
chain:1 57.148 30.738
chain:2 56.674 30.355
chain:3 57.012 32.265
chain:4 53.174 30.492
```