Solutions for assignment 7

Guy Roberts

2023-03-08

Solution to 1.1

In order to answer this question, we used trial and error of varying values of α, this is because we know that:

$$Pr(N \leq 2) = \pi(2) + \pi(1) + \pi(0)$$

We find that our solution is approximately $\alpha \approx 0.4349$ from using our function defined earlier.

$$\int_1^\infty \beta^{-2} d\beta = \left[-\beta^{-1}\right]_1^\infty = 0 + 1 = 1$$

Hence, our prior is a probability density function.

Solution to 1.2

The mode for an Inverse-Gamma distribution with parameters α and β is defined as $\frac{\beta}{\alpha+1}$. So for σ^2 the prior mode will be $\frac{1}{11}$.

And for the prior mode of β, we do not have a prior mode, since the function is undefined at $\beta = 0$.

Solution to 1.3

```
data {
  int<lower=0> N; // num obs
  vector[N] x;    // obs explanatory
  vector[N] y;    // obs response
}

parameters {
  real beta;           // gradient
  real<lower=0> sigma; // error std.
}

model {
  // Priors
  target += -2 * log(beta); // prior for beta
```

```
  target += inv_gamma_lpdf(sigma | 0.1, 0.1); // prior on sigma^2

  // Likelihood
  target += normal_lpdf(y | beta * x, sigma);
}
```

Solution to 1.4

```
library(rstan)

our_data = list(
  N = 8,
  x = c(2.13, 4.32, 3.60, 0.19, 5.62, 2.86, 4.50, 1.95),
  y = c(4.02, 8.73, 7.33, 0.51, 12.09, 5.99, 8.91, 4.02)
)

setwd('C:/Users/guyro/OneDrive/Y3/bayesian-modelling/assignments/assignment7/')

model = stan_model('model1.stan')
fit = sampling(model, our_data, chains=4, iter=10000, warmup=500, thin=1)
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1: Rejecting initial value:
## Chain 1:   Log probability evaluates to log(0), i.e. negative infinity.
## Chain 1:   Stan can't start sampling from this initial value.
## Chain 1: Rejecting initial value:
## Chain 1:   Log probability evaluates to log(0), i.e. negative infinity.
## Chain 1:   Stan can't start sampling from this initial value.
## Chain 1: Rejecting initial value:
## Chain 1:   Log probability evaluates to log(0), i.e. negative infinity.
## Chain 1:   Stan can't start sampling from this initial value.
## Chain 1:
## Chain 1: Gradient evaluation took 4.2e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.42 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 1: Iteration:  501 / 10000 [  5%]  (Sampling)
## Chain 1: Iteration: 1500 / 10000 [ 15%]  (Sampling)
## Chain 1: Iteration: 2500 / 10000 [ 25%]  (Sampling)
## Chain 1: Iteration: 3500 / 10000 [ 35%]  (Sampling)
## Chain 1: Iteration: 4500 / 10000 [ 45%]  (Sampling)
## Chain 1: Iteration: 5500 / 10000 [ 55%]  (Sampling)
## Chain 1: Iteration: 6500 / 10000 [ 65%]  (Sampling)
## Chain 1: Iteration: 7500 / 10000 [ 75%]  (Sampling)
## Chain 1: Iteration: 8500 / 10000 [ 85%]  (Sampling)
## Chain 1: Iteration: 9500 / 10000 [ 95%]  (Sampling)
## Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1:
```

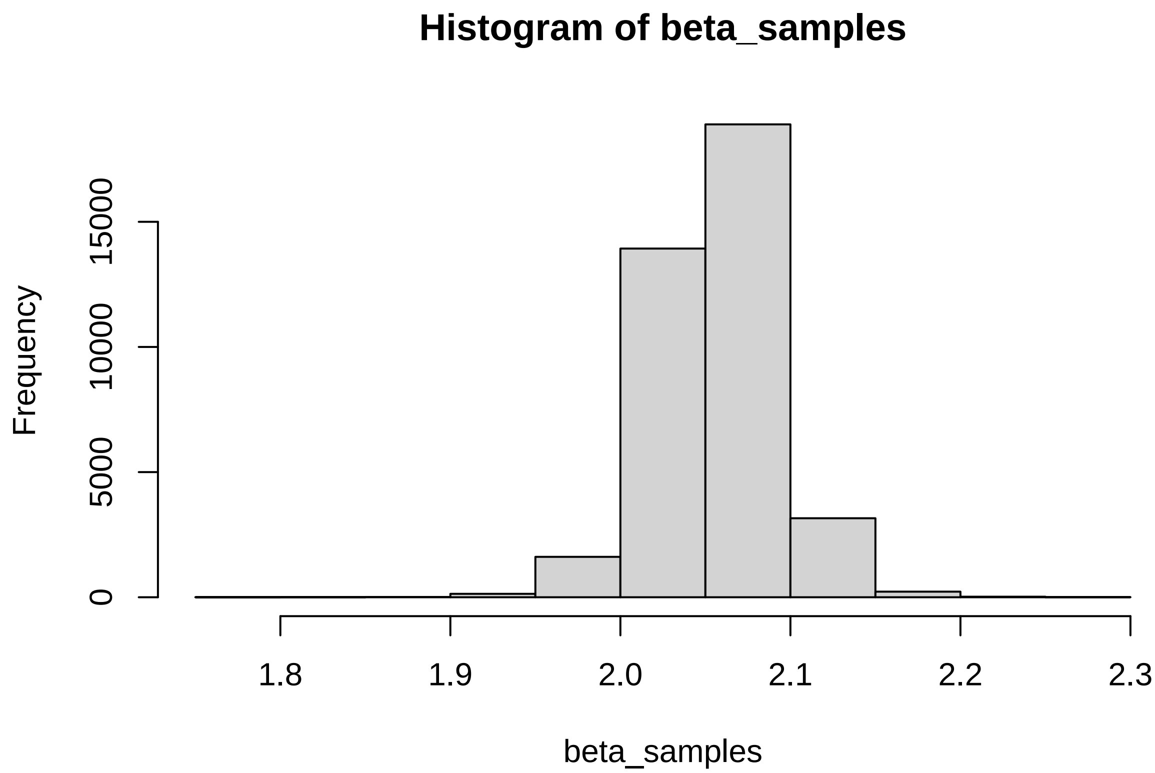
```
## Chain 1:  Elapsed Time: 0.029 seconds (Warm-up)
## Chain 1:                0.285 seconds (Sampling)
## Chain 1:                0.314 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2: Rejecting initial value:
## Chain 2:   Log probability evaluates to log(0), i.e. negative infinity.
## Chain 2:   Stan can't start sampling from this initial value.
## Chain 2:
## Chain 2: Gradient evaluation took 7e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 2: Iteration:  501 / 10000 [  5%]  (Sampling)
## Chain 2: Iteration: 1500 / 10000 [ 15%]  (Sampling)
## Chain 2: Iteration: 2500 / 10000 [ 25%]  (Sampling)
## Chain 2: Iteration: 3500 / 10000 [ 35%]  (Sampling)
## Chain 2: Iteration: 4500 / 10000 [ 45%]  (Sampling)
## Chain 2: Iteration: 5500 / 10000 [ 55%]  (Sampling)
## Chain 2: Iteration: 6500 / 10000 [ 65%]  (Sampling)
## Chain 2: Iteration: 7500 / 10000 [ 75%]  (Sampling)
## Chain 2: Iteration: 8500 / 10000 [ 85%]  (Sampling)
## Chain 2: Iteration: 9500 / 10000 [ 95%]  (Sampling)
## Chain 2: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.015 seconds (Warm-up)
## Chain 2:                0.272 seconds (Sampling)
## Chain 2:                0.287 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 7e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 3: Iteration:  501 / 10000 [  5%]  (Sampling)
## Chain 3: Iteration: 1500 / 10000 [ 15%]  (Sampling)
## Chain 3: Iteration: 2500 / 10000 [ 25%]  (Sampling)
## Chain 3: Iteration: 3500 / 10000 [ 35%]  (Sampling)
## Chain 3: Iteration: 4500 / 10000 [ 45%]  (Sampling)
## Chain 3: Iteration: 5500 / 10000 [ 55%]  (Sampling)
## Chain 3: Iteration: 6500 / 10000 [ 65%]  (Sampling)
## Chain 3: Iteration: 7500 / 10000 [ 75%]  (Sampling)
## Chain 3: Iteration: 8500 / 10000 [ 85%]  (Sampling)
## Chain 3: Iteration: 9500 / 10000 [ 95%]  (Sampling)
## Chain 3: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.018 seconds (Warm-up)
```

```
## Chain 3:                0.28 seconds (Sampling)
## Chain 3:                0.298 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4: Rejecting initial value:
## Chain 4:   Log probability evaluates to log(0), i.e. negative infinity.
## Chain 4:   Stan can't start sampling from this initial value.
## Chain 4:
## Chain 4: Gradient evaluation took 7e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.07 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 4: Iteration:  501 / 10000 [  5%]  (Sampling)
## Chain 4: Iteration: 1500 / 10000 [ 15%]  (Sampling)
## Chain 4: Iteration: 2500 / 10000 [ 25%]  (Sampling)
## Chain 4: Iteration: 3500 / 10000 [ 35%]  (Sampling)
## Chain 4: Iteration: 4500 / 10000 [ 45%]  (Sampling)
## Chain 4: Iteration: 5500 / 10000 [ 55%]  (Sampling)
## Chain 4: Iteration: 6500 / 10000 [ 65%]  (Sampling)
## Chain 4: Iteration: 7500 / 10000 [ 75%]  (Sampling)
## Chain 4: Iteration: 8500 / 10000 [ 85%]  (Sampling)
## Chain 4: Iteration: 9500 / 10000 [ 95%]  (Sampling)
## Chain 4: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.021 seconds (Warm-up)
## Chain 4:                0.255 seconds (Sampling)
## Chain 4:                0.276 seconds (Total)
## Chain 4:
```

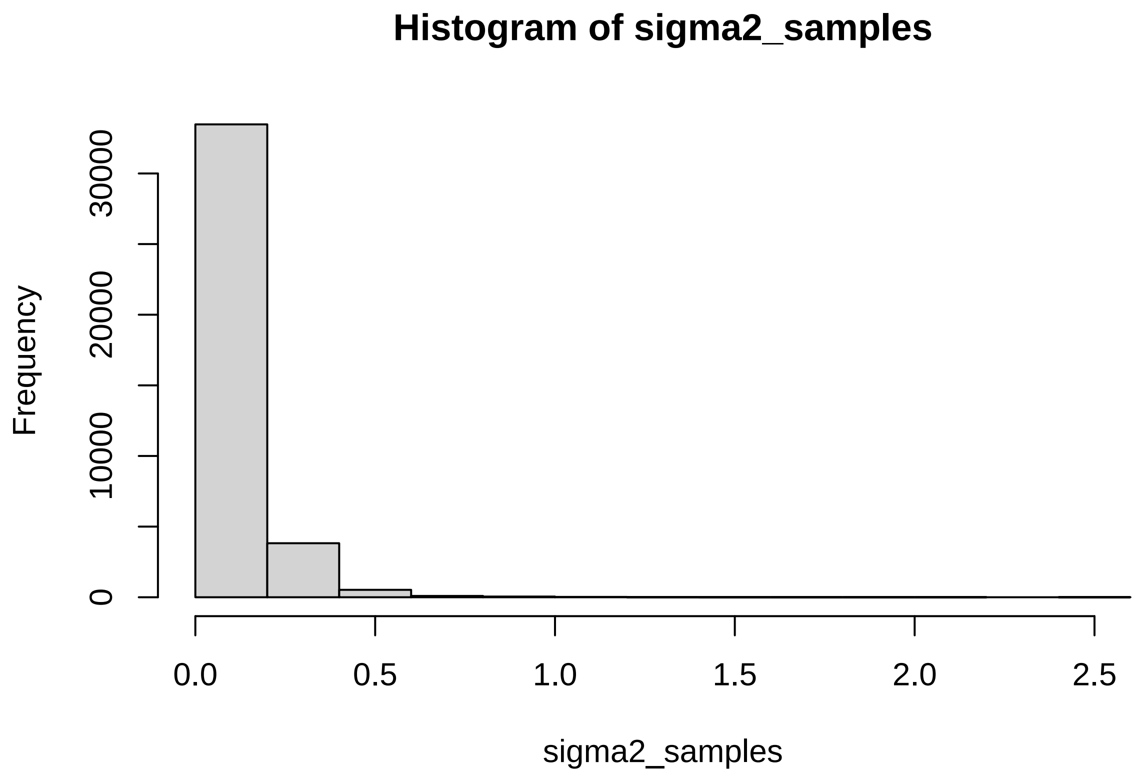
```
print(fit, pars=c("beta", "sigma"))
```

```
## Inference for Stan model: anon_model.
## 4 chains, each with iter=10000; warmup=500; thin=1;
## post-warmup draws per chain=9500, total post-warmup draws=38000.
##
##       mean se_mean   sd 2.5%  25%  50%  75% 97.5% n_eff Rhat
## beta  2.06       0 0.03 1.99 2.04 2.06 2.08  2.12 16061    1
## sigma 0.33       0 0.11 0.20 0.26 0.31 0.38  0.60 14939    1
##
## Samples were drawn using NUTS(diag_e) at Wed Mar  8 20:08:58 2023.
## 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).
```

```
# Extract the posterior samples for beta and sigma
beta_samples = extract(fit)$beta
sigma_samples = extract(fit)$sigma
sigma2_samples = sigma_samples^2

hist(beta_samples)
```




```
hist(sigma2_samples)
```



Solution to 2.1

```
data {
  int<lower=0> N;
  int<lower=0> x[N];
}

parameters {
  real<lower=0> lambda;
}

model {
  // Prior
  lambda ~ frechet(2, 4);

  // Likelihood (Comment out to sample from preposterior)
  //for (i in 1:N) {
  //  x[i] ~ poisson(lambda[i]);
  //}
}
```

Solution to 2.2

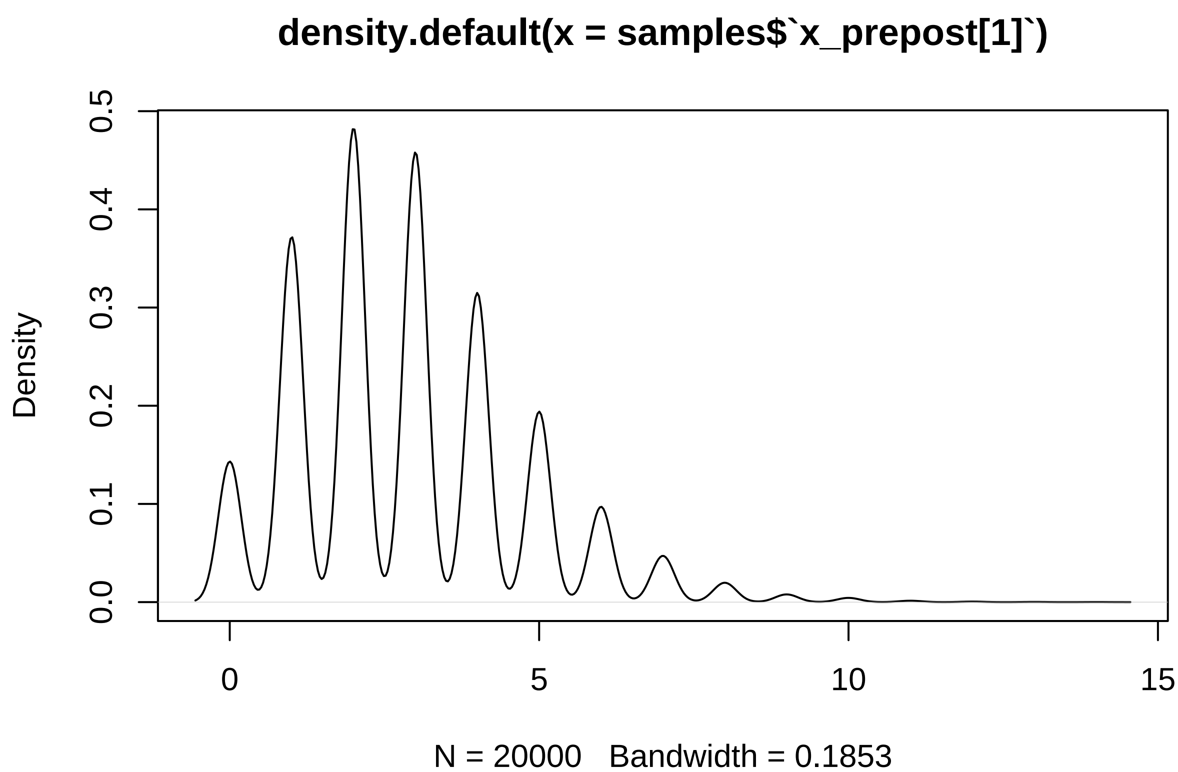
```
library(rstan)
N = 3 # no. data points
x = c(1, 3, 2) # some random count data
our_data = list(N = N, x = x)

model2 = stan_model('model2.stan')
fit = sampling(model2, our_data, iter=10000)
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.5e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 1: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 1: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 1: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 1: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 1: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 1: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 1: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 1: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 1: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 1: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.072 seconds (Warm-up)
## Chain 1:                0.073 seconds (Sampling)
## Chain 1:                0.145 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 5e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 2: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 2: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 2: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 2: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 2: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 2: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 2: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 2: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 2: Iteration: 8000 / 10000 [ 80%]  (Sampling)
```

```
## Chain 2: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 2: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.07 seconds (Warm-up)
## Chain 2:                0.071 seconds (Sampling)
## Chain 2:                0.141 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 4e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.04 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 3: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 3: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 3: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 3: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 3: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 3: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 3: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 3: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 3: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 3: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 3: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.084 seconds (Warm-up)
## Chain 3:                0.11 seconds (Sampling)
## Chain 3:                0.194 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 4e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.04 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 4: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 4: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 4: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 4: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 4: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 4: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 4: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 4: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 4: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 4: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 4: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.071 seconds (Warm-up)
```

```
## Chain 4:                0.069 seconds (Sampling)
## Chain 4:                0.14 seconds (Total)
## Chain 4:
```

```
samples = extract(fit, pars=c('x_prepost[1]', 'x_prepost[2]', 'x_prepost[3]'))
plot(density(samples$`x_prepost[1]`))
```



Solution to 2.3

One solution would be to add some constraints to the model. From the lectures in Chapter 10, we could truncate our model, in order to force the generated samples to be greater than 4. Essentially adding a lower bound. However, this would mean that no samples would be less than 4, which isn't exactly what we want.

Alternatively, we could modify our prior distribution for *lambda* to "shift" the density further from 4. For example, we use the location parameter, m of the Frechet distribution. Though I am not entirely sure how to implement this in Stan, since the distribution only takes in the scale and shape parameters, not the location parameter, m.