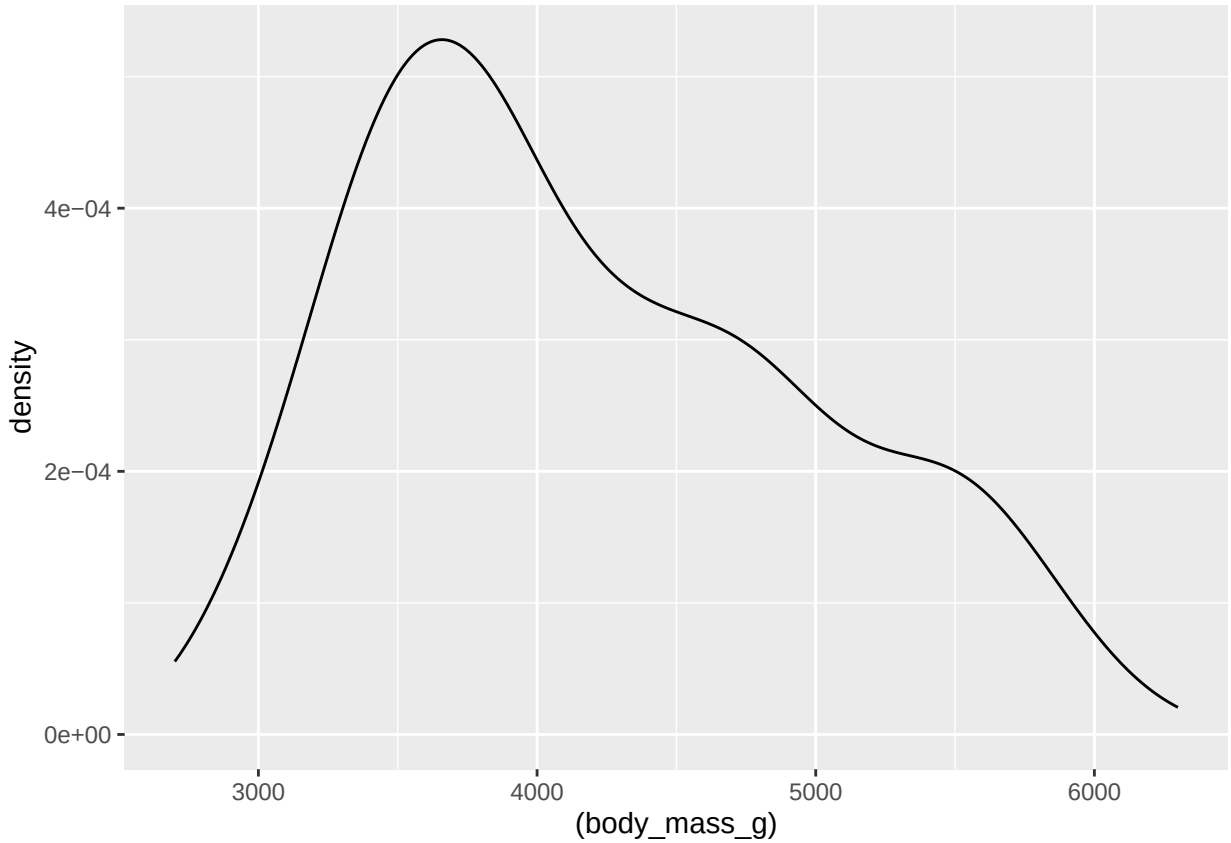
STAT 450: Bayesian Statistics - Homework 9

Problem 1: Penguins' Body Mass

```
library(bayesrules)
data("penguins_bayes")   # load dataset

penguins_bayes <- penguins_bayes %>% filter(!is.na(sex))   # remove missing entries
start_val <- list(
              list(".RNG.name" = "base::Wichmann-Hill", ".RNG.seed" = 45091),
              list(".RNG.name" = "base::Wichmann-Hill", ".RNG.seed" = 45092)
              )

ggplot(data = penguins_bayes) + geom_density(aes(x = (body_mass_g)))
```



The normal density plot seemed to be right-skewed. I would expect that a log transformation would make the situation better, but I'm honestly not confident with the calculations and syntax associated with a lognormal model outside of sampling, especially in making graphs. I looked into both lognormal and normal models, and found that despite the skewness, a normal model was still good enough. So I just proceeded with that as normal.

Model A

```
mtext <- "
model{
  for(i in 1:n){
    y[i] ~ dnorm(beta0 + beta1*x[i], tau)
  }
  beta0 ~ dnorm(0, 1/(100^2))
  beta1 ~ dnorm(0, 1/(100^2))
  tau ~ dgamma(1,1)

  mu = beta0 + beta1*200

  ytilde ~ dnorm(beta0 + beta1*200, tau)
}
"
dat <- list(y = penguins_bayes$body_mass_g,
            x = penguins_bayes$flipper_length_mm,
            n = length(penguins_bayes$body_mass_g))

outA <- run.jags(model = mtext,
                 monitor = c("beta0", "beta1", "tau","mu","ytilde"),
                 data = dat,
                 n.chains = 2,
                 inits = start_val,
                 sample = 10000,
                 thin = 10)
```

```
## Loading required namespace: rjags
```

```
## Compiling rjags model...
## Calling the simulation using the rjags method...
## Note: the model did not require adaptation
## Burning in the model for 4000 iterations...
## Running the model for 100000 iterations...
## Simulation complete
## Calculating summary statistics...
## Calculating the Gelman-Rubin statistic for 5 variables....
## Note: Unable to calculate the multivariate psrf
## Finished running the simulation
```

```
outA
```

```
##
## JAGS model summary statistics from 20000 samples (thin = 10; chains = 2; adapt+burnin = 5000):
##
##           Lower95     Median    Upper95       Mean         SD Mode     MCerr
## beta0     -489.38     -295.7    -99.503     -295.6     99.646   --    1.1032
## beta1      21.541     22.538     23.557     22.538    0.51689   -- 0.0057345
## tau    2.7917e-06 3.2931e-06 3.8093e-06 3.3016e-06 2.6204e-07   -- 1.932e-09
## mu           4152     4212.1     4269.4     4212.1     30.104   --   0.21287
## ytilde     3148.5     4217.1     5281.2     4213.3     547.57   --    3.8719
##
##        MC%ofSD SSeff      AC.100    psrf
## beta0      1.1  8159  -0.0013786  1.0002
## beta1      1.1  8125 -0.00037309  1.0003
```

```
## tau         0.7 18395    0.013508       1
## mu          0.7 20000   0.0020191       1
## ytilde      0.7 20000  -0.0093742 0.99999
##
## Total time taken: 3.8 seconds
```

Model B

```
penguins_bayes$sex <- factor(penguins_bayes$sex)

mtext <- "
model{
  for(i in 1:n){
   y[i] ~ dnorm(beta0 + beta1[x[i]], tau)
  }
  beta0 ~ dnorm(0, 1/(1000^2))
  beta1[1] = 0
  for(j in 2:numlevels){
    beta1[j] ~ dnorm(0, 1/(1000^2))
  }
  tau ~ dgamma(1,1)

  mu_Female = beta0
  mu_Male = beta0 + beta1[2]

  ytilde_F ~ dnorm(mu_Female, tau)
  ytilde_M ~ dnorm(mu_Male, tau)

}
"
dat <- list(y = penguins_bayes$body_mass_g, x=penguins_bayes$sex,
             n = length(penguins_bayes$body_mass_g),
             numlevels = n_distinct(penguins_bayes$sex))

outB <- run.jags(model = mtext,
                 monitor = c("beta0", "beta1", "tau","ytilde_F","ytilde_M"),
                 data = dat,
                 n.chains = 2,
                 inits = start_val,
                 sample = 10000,
                 thin = 10)
```

```
## Compiling rjags model...
## Calling the simulation using the rjags method...
## Note: the model did not require adaptation
## Burning in the model for 4000 iterations...
## Running the model for 100000 iterations...
## Simulation complete
## Calculating summary statistics...
## Note: The monitored variable 'beta1[1]' appears to be non-stochastic;
## it will not be included in the convergence diagnostic
## Calculating the Gelman-Rubin statistic for 6 variables....
## Finished running the simulation
```

Model C

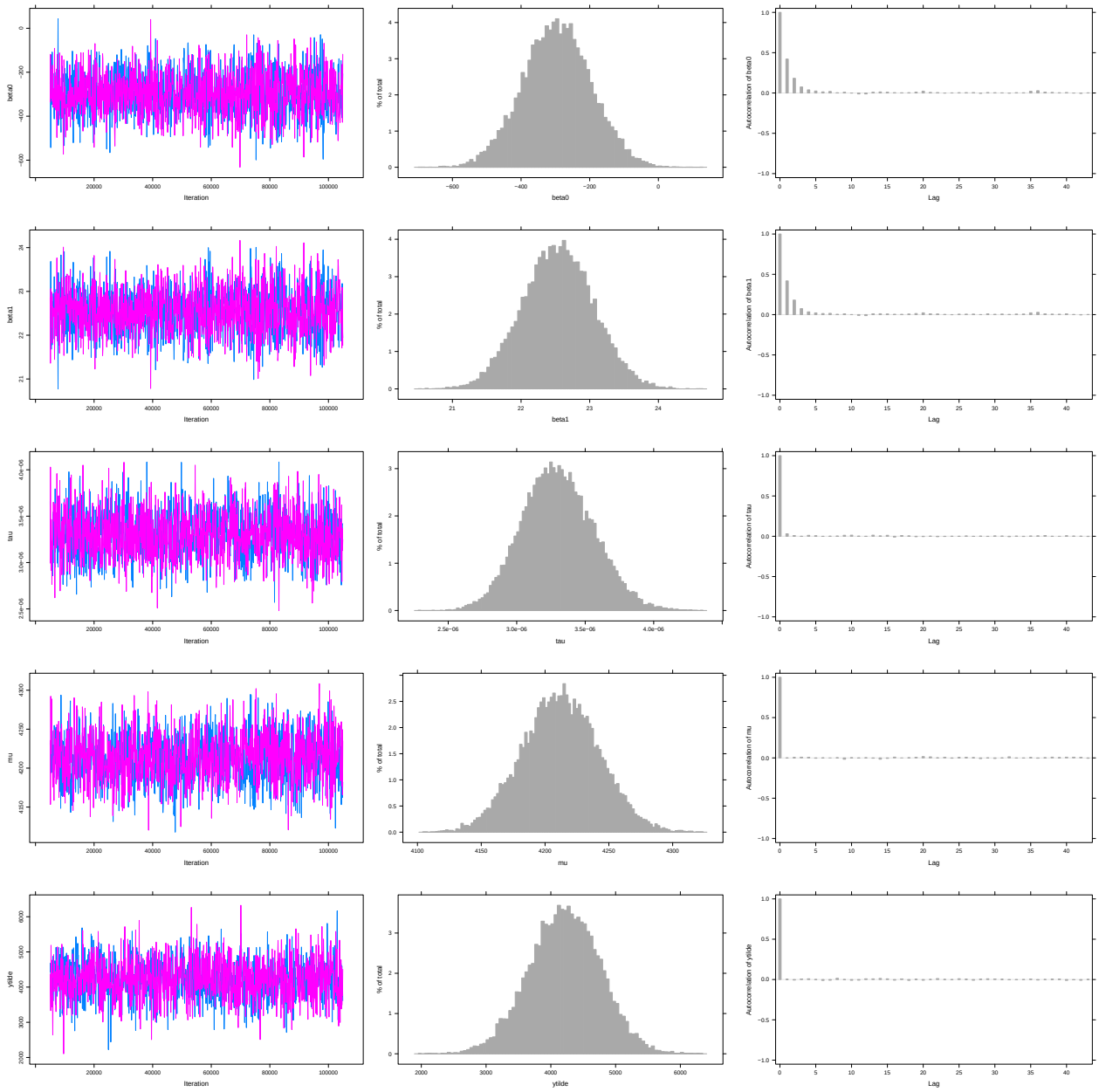
```
mtext <- "
model{
  for(i in 1:n){
    y[i] ~ dnorm(beta0 + beta1*x1[i] + beta2[x2[i]], tau)
  }
  beta0 ~ dnorm(0, 1/(1000^2))
  beta1 ~ dnorm(0, 1/(1000^2))
  beta2[1] = 0
  for(j in 2:numlevels){
    beta2[j] ~ dnorm(0, 1/(1000)^2)
  }
  tau ~ dgamma(1,1)
}
"

dat <- list(y = penguins_bayes$body_mass_g,
            n = length(penguins_bayes$body_mass_g),
            numlevels = n_distinct(penguins_bayes$sex),
            x1 = penguins_bayes$flipper_length_mm,
            x2 = penguins_bayes$sex)

outC <- run.jags(model = mtext,
                 monitor = c("beta0", "beta1", "beta2", "tau"),
                 data = dat,
                 n.chains = 2,
                 inits = start_val,
                 sample = 10000,
                 thin = 10)
```

```
## Compiling rjags model...
## Calling the simulation using the rjags method...
## Note: the model did not require adaptation
## Burning in the model for 4000 iterations...
## Running the model for 100000 iterations...
## Simulation complete
## Calculating summary statistics...
## Note: The monitored variable 'beta2[1]' appears to be non-stochastic;
## it will not be included in the convergence diagnostic
## Calculating the Gelman-Rubin statistic for 5 variables....
## Finished running the simulation
```

(a)

```
plot(outA, plot.type = c("trace", "histogram", "autocorr"), layout = c(5,3))
```

```
## Generating plots...
```



Everything looks great. I think the model is okay!

Median Absolute Error:

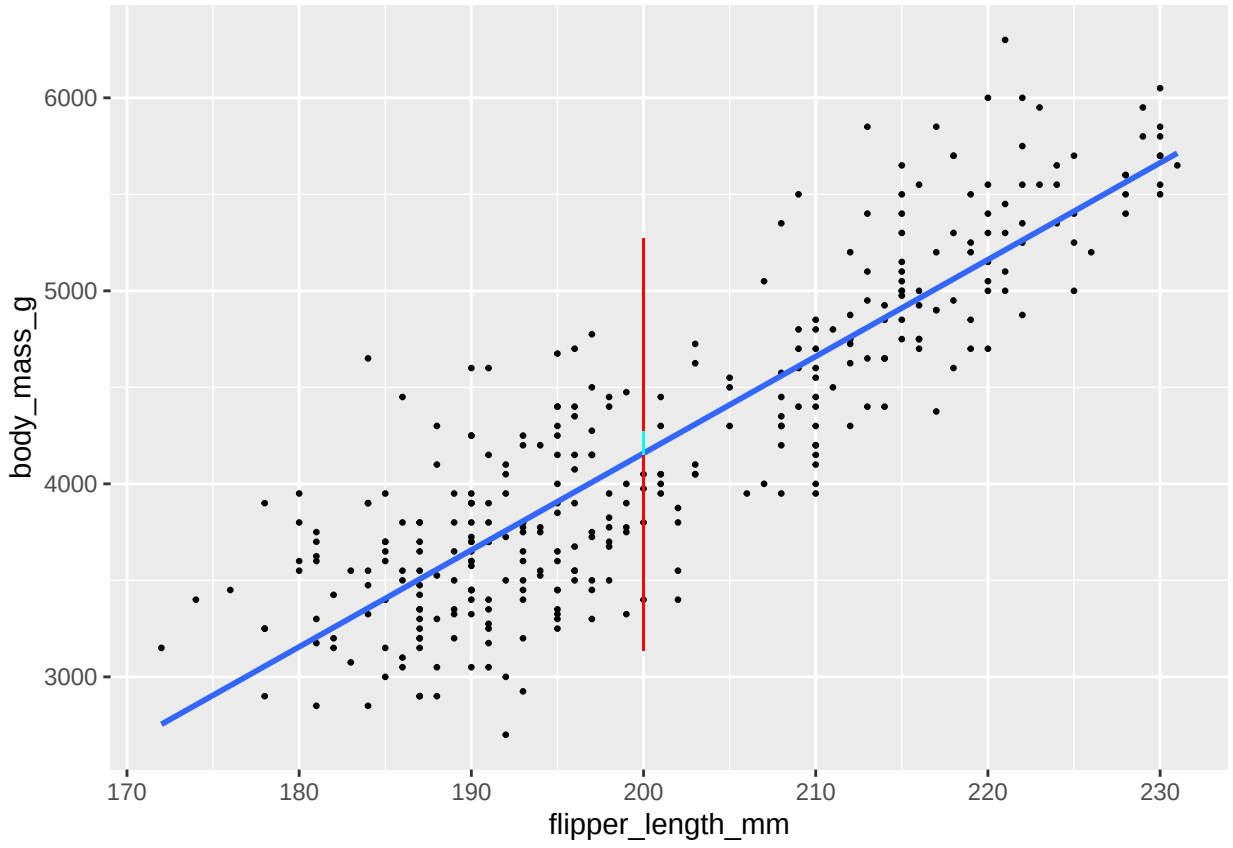
```
parameters <- rbind(outA$mcmc[[1]], outA$mcmc[[2]])

beta0 <- parameters[,1]
beta1 <- parameters[,2]
tau <- parameters[,3]
mu <- parameters[,4]
ytilde <- parameters[,5]

repdatasets <- matrix(nrow=100, ncol = nrow(penguins_bayes))

for(i in 1:nrow(repdatasets))
{
```

```
  for(j in 1:ncol(repdatasets))
  {
    repdatasets[i,j] = rnorm(1,
                             mean = beta0[i] + beta1[i]*penguins_bayes$flipper_length_mm[i],
                             sd = sqrt(1/tau[i]))
  }
}

post_pred_medians <- apply(repdatasets, 2, median)

median(abs(penguins_bayes$body_mass_g-post_pred_medians))
```

```
## [1] 553.367
```

```
ggplot(penguins_bayes, aes(x = flipper_length_mm, y = body_mass_g)) +
  geom_point(size = 0.5) +
  geom_smooth(method = "lm", se = FALSE) +
  geom_linerange(aes(x = 200,
                     ymin = quantile(ytilde, c(0.025, 0.975))[1],
                     ymax = quantile(ytilde, c(0.025, 0.975))[2]),
                 color = "red") +
  geom_linerange(aes(x = 200,
                     ymin = quantile(mu, c(0.025, 0.975))[1],
                     ymax = quantile(mu, c(0.025, 0.975))[2]),
                 color = "cyan")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



(b)

Model A: 95% credible intervals for the mean body mass of penguins with a flipper length of 200 mm.

```
quantile(mu, c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 4153.351 4270.761
```

```
quantile(ytilde, c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 3135.319 5271.975
```

95% credible intervals for the body mass of an individual penguin with a flipper length of 200 mm.

(c)

The CI for beta1_2, which is the difference in the body mass of male penguins compared to female penguins.

```
beta1_2 <- parameters[,3]
options(scipen = 999)
quantile(beta1_2, c(0.025,0.975))
```

```
##           2.5%          97.5%
## 0.000002812431 0.000003835759
```

The CI is entirely above 0, so male penguins, on average, have higher body masses than female penguins.

```
parameters <- rbind(outB$mcmc[[1]], outB$mcmc[[2]])

ytildeM <- parameters[,6]
ytildeF <- parameters[,5]
```

The average body mass of Male penguins, Female penguins, and the difference, respectively:

```
mean(ytildeM)
```

```
## [1] 4553.769
```

```
mean(ytildeF)
```

```
## [1] 3845.902
```

```
mean(ytildeM - ytildeF)
```

```
## [1] 707.867
```
