STA465 Homework 1

1.1 $Pr(\mu|Y) = \frac{Pr(Y,\mu)}{Pr(Y)} = \frac{p(Y|\mu,\sigma)\,p(\mu)}{P(Y)} \propto p(Y|\mu,\sigma)\,p(\mu)$

likelihood: $P(Y|\mu,\sigma) = \prod_{i=1}^{T} P(Y_i|\mu,\sigma)$

$$= (2\pi\sigma^2)^{-T/2} \exp\left[-\frac{1}{2\sigma^2}\sum_{i=1}^{T}(Y_i-\mu)^2\right]$$

$$\propto \sigma^{-T} \exp\left[\frac{-1}{2\sigma^2}\sum_{i=1}^{T}(Y_i-\mu)^2\right]$$

prior: $P(\mu) = (2\pi v^2)^{-1/2} \exp\left[\frac{-1}{2v^2}(\mu-s)^2\right]$

Posterior: $P(\mu|Y) = (2\pi\sigma^2)^{-T/2}(2\pi v^2)^{-1/2} \exp\left[-\frac{1}{2\sigma^2}\sum_{i=1}^{T}(Y_i-\mu)^2\right]\exp\left[\frac{-1}{2v^2}(\mu-s)^2\right]$

$$= (2\pi\sigma^2)^{-T/2}(2\pi v^2)^{-1/2} \exp\left[-\frac{1}{2}\left(\frac{1}{\sigma^2}\sum_{i=1}^{T}(Y_i-\mu)^2 + \frac{1}{v^2}(\mu-s)^2\right)\right]$$

$$= (2\pi\sigma^2)^{-T/2}(2\pi v^2)^{-1/2} \exp\left[-\frac{1}{2v_T^2}(\mu-s_T)^2\right]$$ by (2.11, 2.12) in Bayesian Data Analysis 3rd ed.

where $s_T = \frac{\frac{1}{v^2}s + \frac{T}{\sigma^2}\bar{Y}}{\frac{1}{v^2} + \frac{T}{\sigma^2}}$

$v_T = \left(\frac{1}{v^2} + \frac{T}{\sigma^2}\right)^{-1}$ by slide 69, Lec1 and (2.12) in BDA. 3rd ed.

so $\mu|Y \sim N(s_T, v_T)$

1.2 $Y_i \sim \text{Binomial}(N, p)$

$p \sim \text{Beta}(\alpha, \beta)$

$$P(Y|p) = \prod_{i=1}^{T} \text{Bin}(Y_i|N,p) = \prod_{i=1}^{T} \binom{N}{Y_i} p^{Y_i}(1-p)^{N-Y_i}$$

$$P(p) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1}(1-p)^{\beta-1}$$

$$P(p|Y) = \frac{P(Y|p)P(p)}{P(Y)} \propto P(Y|P)P(p)$$

$$= \left[\prod_{i=1}^{T} \binom{N}{Y_i} p^{Y_i}(1-P)^{N-Y_i}\right]\left[\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} P^{\alpha-1}(1-P)^{\beta-1}\right]$$

$$\propto P^{T\bar{Y}}(1-P)^{T(N-\bar{Y})} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} P^{\alpha-1}(1-P)^{\beta-1}$$

$$\propto p^{T\bar{Y}}(1-p)^{T(N-\bar{Y})} \; p^{\alpha-1}(1-p)^{\beta-1}$$

$$= p^{T\bar{Y}+\alpha-1}(1-p)^{T(N-\bar{Y})+\beta-1}$$

- $\binom{N}{Y_i}$ does not depend on p and can thus be treated as a constant
- similarly, $\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}$

so $p|y \sim \text{Beta}(\alpha^*, \beta^*)$ where $\alpha^* = T\bar{Y}+\alpha$

$\beta^* = T(N-\bar{Y})+\beta$

Homework 1 STA465

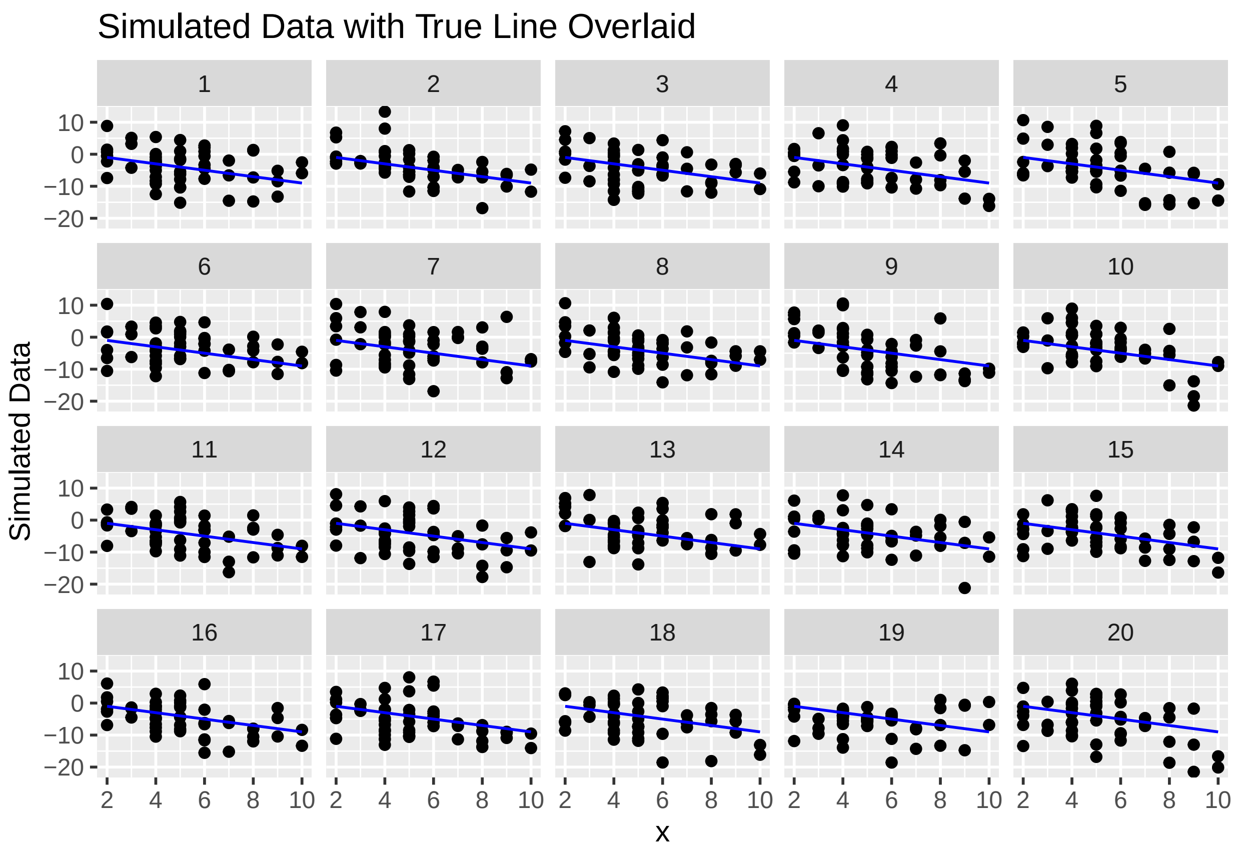
Tianyi Zhang

30/01/2022

Question 2.1

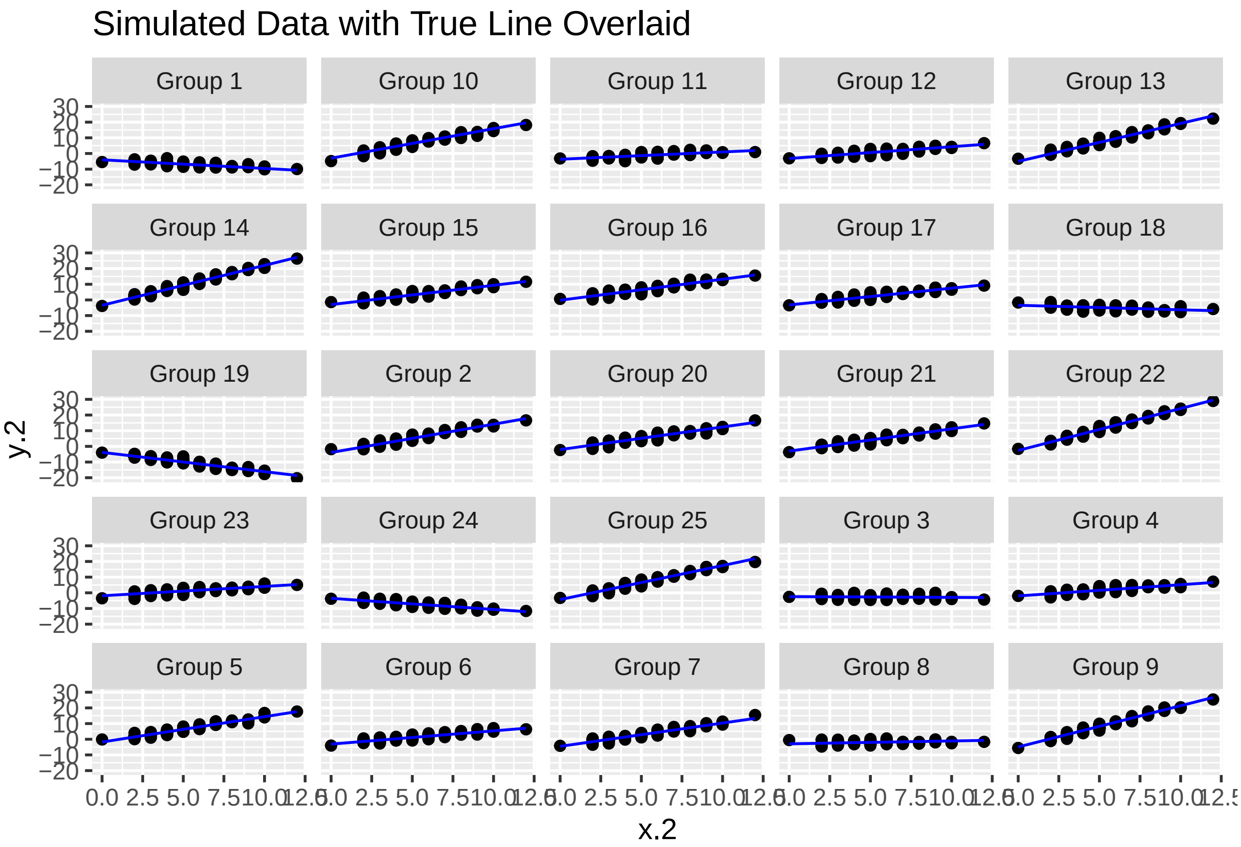
```
#Question 2.1
beta0 <- 1
beta1 <- -1
sigma <- 5

set.seed(465)

x <- rpois(n = 50, lambda = 5)
y.mean <- beta0 + beta1*x
y <- rnorm(n = 50, mean = y.mean, sd = sigma)
```

```
#The following code uses Code from "Simulation: Linear Regression Model + Priors" as reference
sim.data20 <- tibble(x = rep(x, 20),
                     y.mean = rep(y.mean, 20),
                     y.sim = rnorm(n = 20*50,
                                   mean = y.mean,
                                   sd = sigma),
                     group = rep(1:20, each = 50))

#Plot the Simulated Data with the true line overlaid
ggplot(data = sim.data20, aes(x,y.sim)) + geom_point() +
  geom_line(aes(x,y.mean), col="blue") +
  ylab("Simulated Data") +
  ggtitle("Simulated Data with True Line Overlaid") +
  facet_wrap(~group)
```



Question 2.2

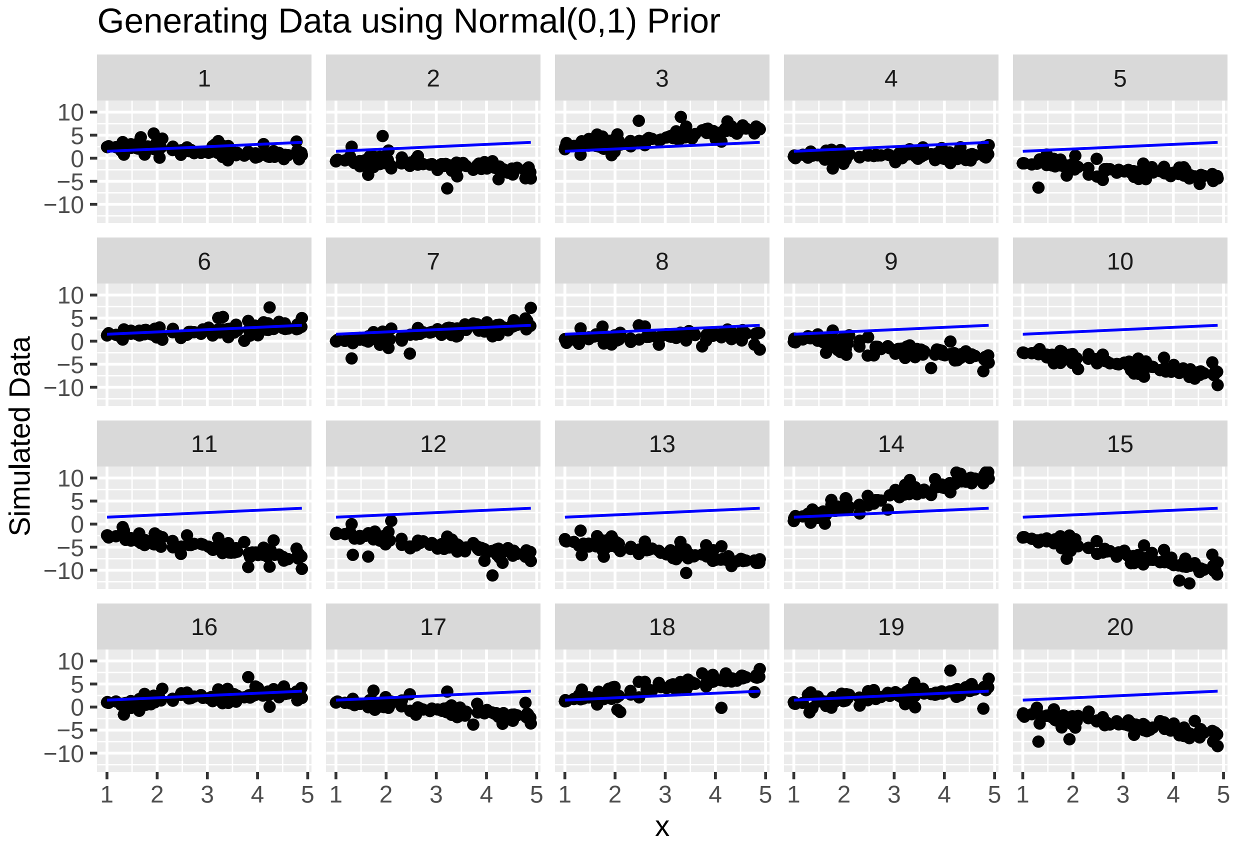
```
#Question 2.2
#The following code uses code from "Simulating From All Sorts of Linear Models" as reference
set.seed(465)
x.2 <- rpois(n = 100, lambda = 5)
beta0.2 <- rnorm(n=25, mean=-3, sd=1)
beta1.2 <- rnorm(n=25, mean=1, sd=1)
sigma.2 <- 1

y.mean.2 <- rep(beta0.2, each=100) +
            rep(beta1.2, each=100) *
            rep(x.2, 25)
y.2 <- rnorm(n = 25*100, mean = y.mean.2, sd = sigma.2)

sim.data.2 <- tibble(x.2 = rep(x.2, 25),
                   y.2,
                   y.mean.2,
                   group.2 = paste("Group",
                                  rep(1:25, each = 100)))

ggplot(sim.data.2, aes(x.2,y.2)) + geom_point() +
         geom_line(aes(x.2, y.mean.2), col="blue") +
         ggtitle("Simulated Data with True Line Overlaid") +
         facet_wrap(~group.2)
```



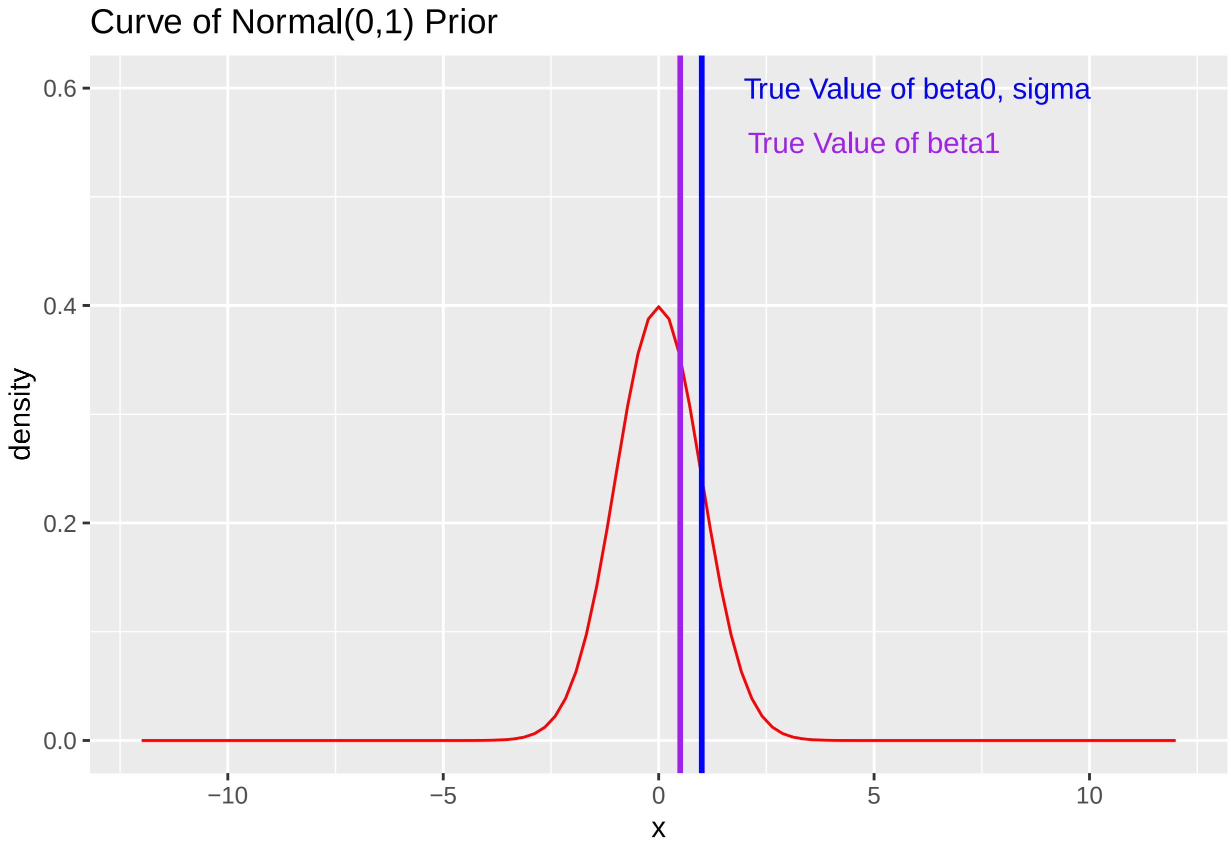
Question 3.1: Linear Regression

```
#Question 3.1
#Simulated Data from Handout
beta0 <- 1
beta1 <- 0.5
sigma <- 1
set.seed(17)
x <- runif(n = 100, min = 1, max=5)
y.mean <- beta0 + beta1*x
y <- rnorm(n = 100,
mean = y.mean,
sd = sigma)
sim.data <- tibble(x,y, y.mean)

#--------------------------------------------------------------------------------
#First Prior Distribution
#--------------------------------------------------------------------------------

beta0.prior1 <- rnorm(n = 20, mean=0, sd=1)
```

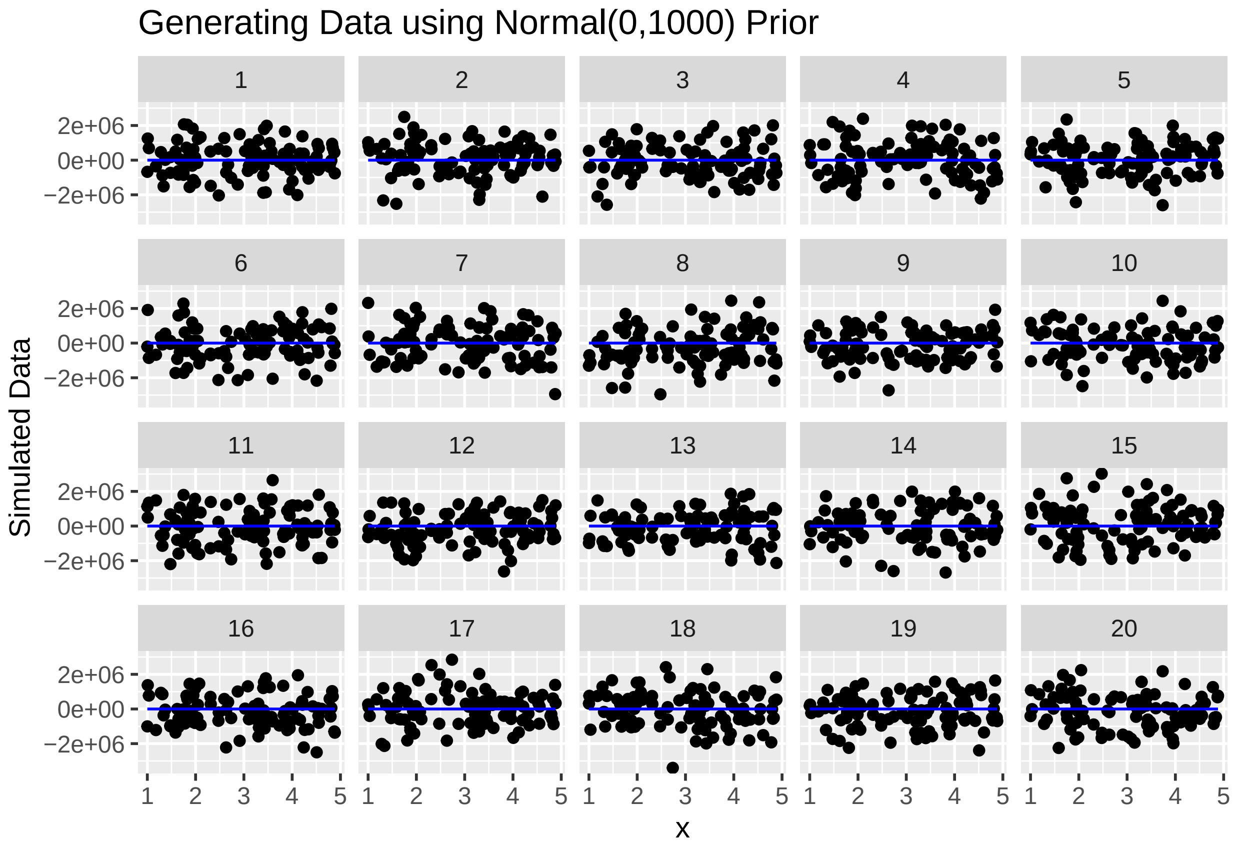
```
beta1.prior1 <- rnorm(n = 20, mean=0, sd=1)
sigma.prior1 <- rgamma(n = 20, shape=1, scale=1)

sim.data.p1 <- data.frame(x = rep(x, 20),
                          y.mean.p1 = rep(NA, 20*100),
                          y.p1 = rep(NA, 20*100),
                          group = rep(1:20, each = 100))

for(j in 1:20){
  y.mean.p1.sim <- beta0.prior1[j] + beta1.prior1[j]*x
  y.p1.sim <- rnorm(n = length(x),
                    mean = y.mean.p1.sim,
                    sd=sigma.prior1)
  sim.data.p1$y.mean.p1[sim.data.p1$group == j] <- y.mean.p1.sim
  sim.data.p1$y.p1[sim.data.p1$group == j] <- y.p1.sim
}

ggplot(data = sim.data.p1, aes(x,y.p1)) + geom_point() +
  geom_line(data=sim.data, aes(x,y.mean), col="blue") +
  ylab("Simulated Data") +
  facet_wrap(~group) +
  ggtitle("Generating Data using Normal(0,1) Prior")
```



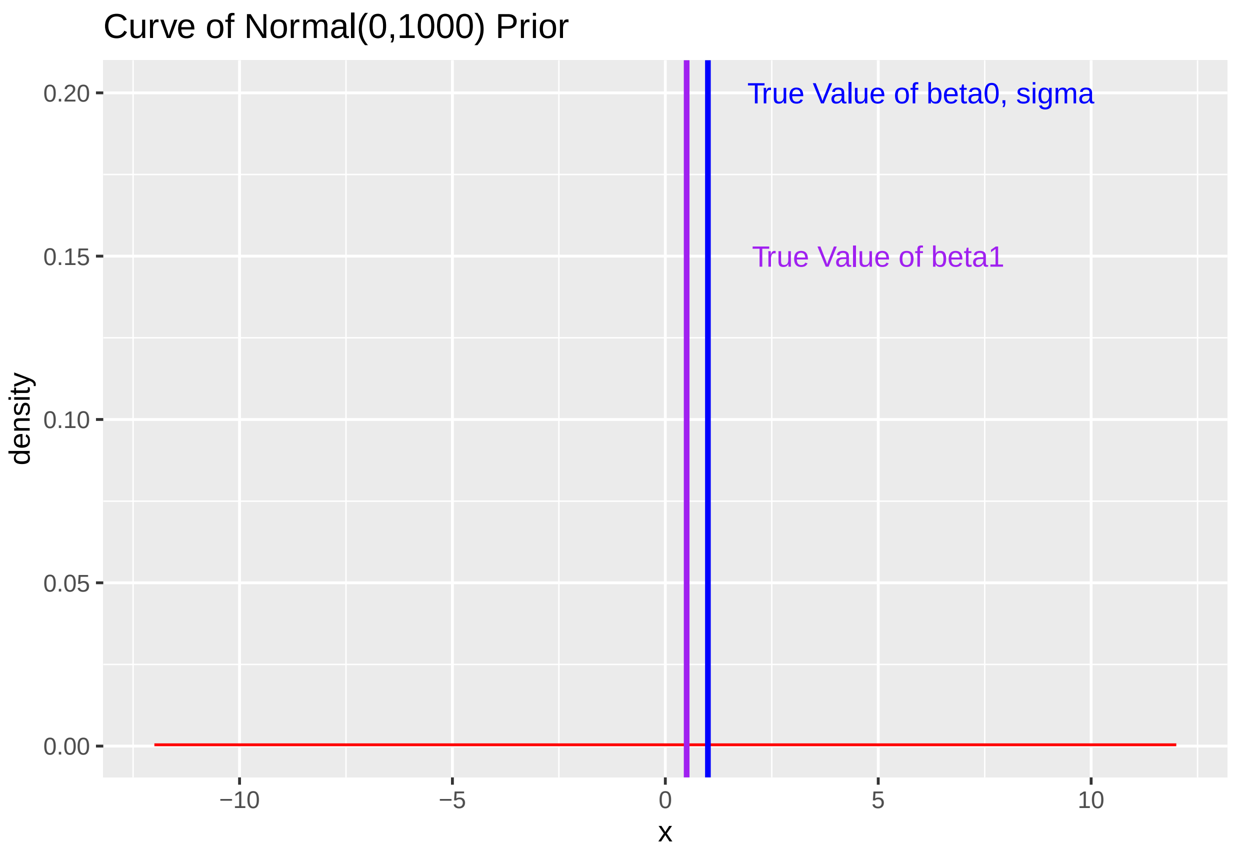
```
#Curve of first prior distribution
ggplot(data.frame(bx = seq(-12, 12, length.out = 100)), aes(bx)) +
  stat_function(fun = dnorm, colour="red") +
  geom_vline(xintercept = beta0, color="blue", size=1) +
  geom_vline(xintercept = beta1, color="purple", size=1) +
  annotate("text", x=6,y=0.6, label="True Value of beta0, sigma", col="blue") +
  annotate("text", x=5,y=0.55, label="True Value of beta1", col="purple") +
  ylab("density") +
  xlab("x") +
  ggtitle("Curve of Normal(0,1) Prior")
```



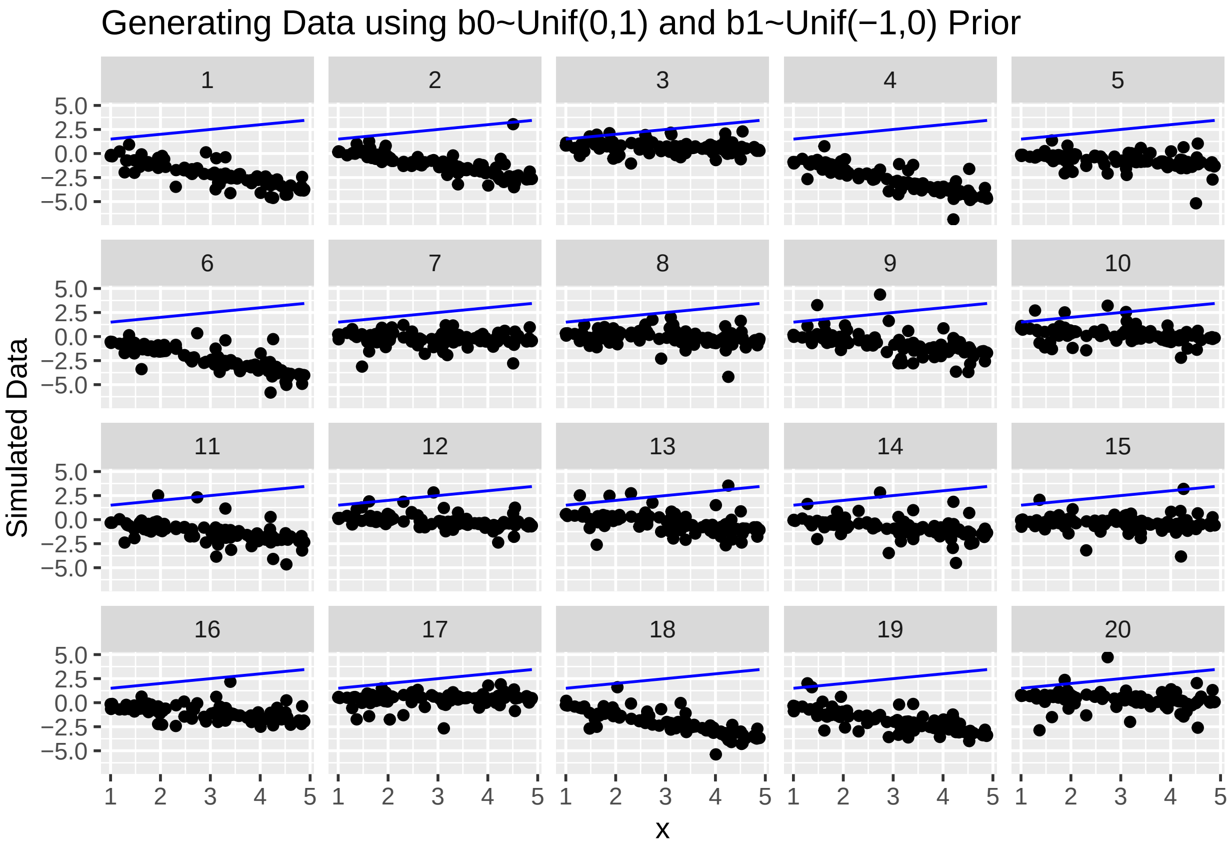
```
#-----------------------------------------------------------------------------
#Second Prior Distribution
#-----------------------------------------------------------------------------

beta0.prior2 <- rnorm(n = 20, mean=0, sd=1000)
beta1.prior2 <- rnorm(n = 20, mean=0, sd=1000)
sigma.prior2 <- rgamma(n = 20, shape=1000, scale=1000)

sim.data.p2 <- data.frame(x = rep(x, 20),
                          y.mean.p2 = rep(NA, 20*100),
                          y.p2 = rep(NA, 20*100),
                          group = rep(1:20, each = 100))

for(j in 1:20){
  y.mean.p2.sim <- beta0.prior2[j] + beta1.prior2[j]*x
  y.p2.sim <- rnorm(n = length(x),
                    mean = y.mean.p2.sim,
                    sd=sigma.prior2)
  sim.data.p2$y.mean.p2[sim.data.p2$group == j] <- y.mean.p2.sim
  sim.data.p2$y.p2[sim.data.p2$group == j] <- y.p2.sim
}

ggplot(data = sim.data.p2, aes(x,y.p2)) + geom_point() +
  geom_line(data=sim.data, aes(x,y.mean), col="blue") +
  ylab("Simulated Data") +
  facet_wrap(~group) +
  ggtitle("Generating Data using Normal(0,1000) Prior")
```



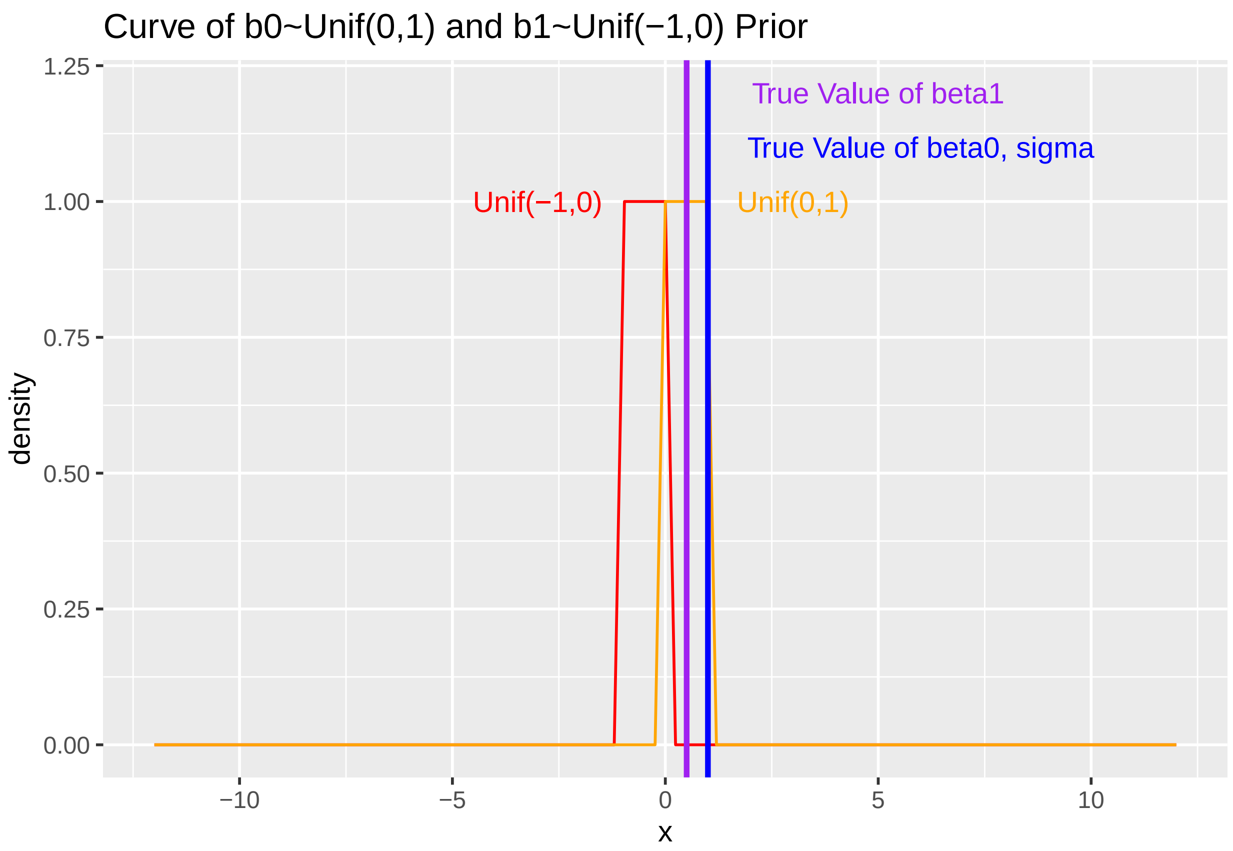
```
#Curve of Second prior distribution
ggplot(data.frame(bx = seq(-12, 12, length.out = 100)), aes(bx)) +
  stat_function(fun = dnorm, args = list(mean=0,sd=1000), colour="red") +
  geom_vline(xintercept = beta0, color="blue", size=1) +
  geom_vline(xintercept = beta1, color="purple", size=1) +
  annotate("text", x=6,y=0.2, label="True Value of beta0, sigma", col="blue") +
  annotate("text", x=5,y=0.15, label="True Value of beta1", col="purple") +
  ylab("density") +
  xlab("x") +
  ggtitle("Curve of Normal(0,1000) Prior")
```



```
#-------------------------------------------------------------------------------
#Third Prior Distribution
#-------------------------------------------------------------------------------

beta0.prior3 <- runif(n = 20, min=0, max=1)
beta1.prior3 <- runif(n = 20, min=-1, max=0)
sigma.prior3 <- rexp(n = 20, rate=1)

sim.data.p3 <- data.frame(x = rep(x, 20),
                          y.mean.p3 = rep(NA, 20*100),
                          y.p3 = rep(NA, 20*100),
                          group = rep(1:20, each = 100))

for(j in 1:20){
  y.mean.p3.sim <- beta0.prior3[j] + beta1.prior3[j]*x
  y.p3.sim <- rnorm(n = length(x),
                    mean = y.mean.p3.sim,
                    sd=sigma.prior3)
  sim.data.p3$y.mean.p3[sim.data.p3$group == j] <- y.mean.p3.sim
  sim.data.p3$y.p3[sim.data.p3$group == j] <- y.p3.sim
}

ggplot(data = sim.data.p3, aes(x,y.p3)) + geom_point() +
  geom_line(data=sim.data, aes(x,y.mean), col="blue") +
  ylab("Simulated Data") +
  facet_wrap(~group) +
  ggtitle("Generating Data using b0~Unif(0,1) and b1~Unif(-1,0) Prior")
```



```
#Curve of Third prior distribution
ggplot(data.frame(bx = seq(-12, 12, length.out = 100)), aes(bx)) +
  stat_function(fun = dunif, args = list(min=-1,max=0), colour="red") +
  stat_function(fun = dunif, args = list(min=0,max=1), colour="orange") +
  annotate("text", x=-3,y=1, label="Unif(-1,0)", col="red") +
  annotate("text", x=3,y=1, label="Unif(0,1)", col="orange") +
  geom_vline(xintercept = beta0, color="blue", size=1) +
  geom_vline(xintercept = beta1, color="purple", size=1) +
  annotate("text", x=6,y=1.1, label="True Value of beta0, sigma", col="blue") +
  annotate("text", x=5,y=1.2, label="True Value of beta1", col="purple") +
  ylab("density") +
  xlab("x") +
  ggtitle("Curve of b0~Unif(0,1) and b1~Unif(-1,0) Prior")
```

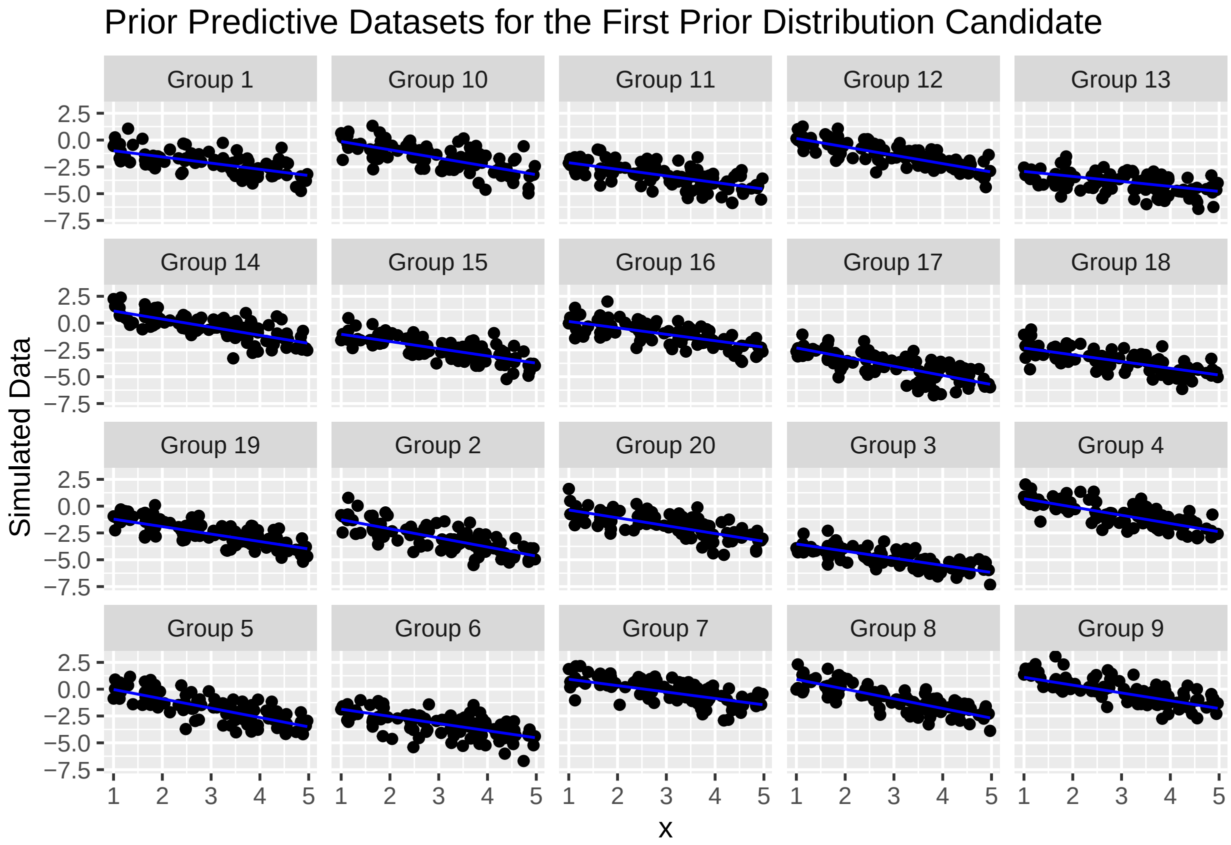


From the simulation, we can see that data generation is extremely dependent on our choice of prior distribution. Choosing a noninformative prior such as Normal(0,1000) results in data that seems evenly spread along mean 0 with huge variations and no discernible patterns, meaning that it does not provide any specific information about the unknown parameters. Choosing a more informative prior such as Normal(0,1) results in data that has more structure and, in some cases, resembles the data sample from the true parameter values. In some simulated data sets, though, the direction of the dataset mirrors the true line of the parameter values, meaning that a Normal(0,1) cannot tell us whether the true parameter values are positive or negative. Choosing a prior distribution such as Unif(-1,0) for β_1 led to data that often opposed the direction of the true line (negative slope vs. positive slope), which is expected since the true value of $\beta_1 = 0.5$ is not included between -1 and 0. This shows that we should have a good idea of the where the true value of the parameter is to ensure it is included in the prior distribution when we want to choose strict priors such as Unif(-1,0).

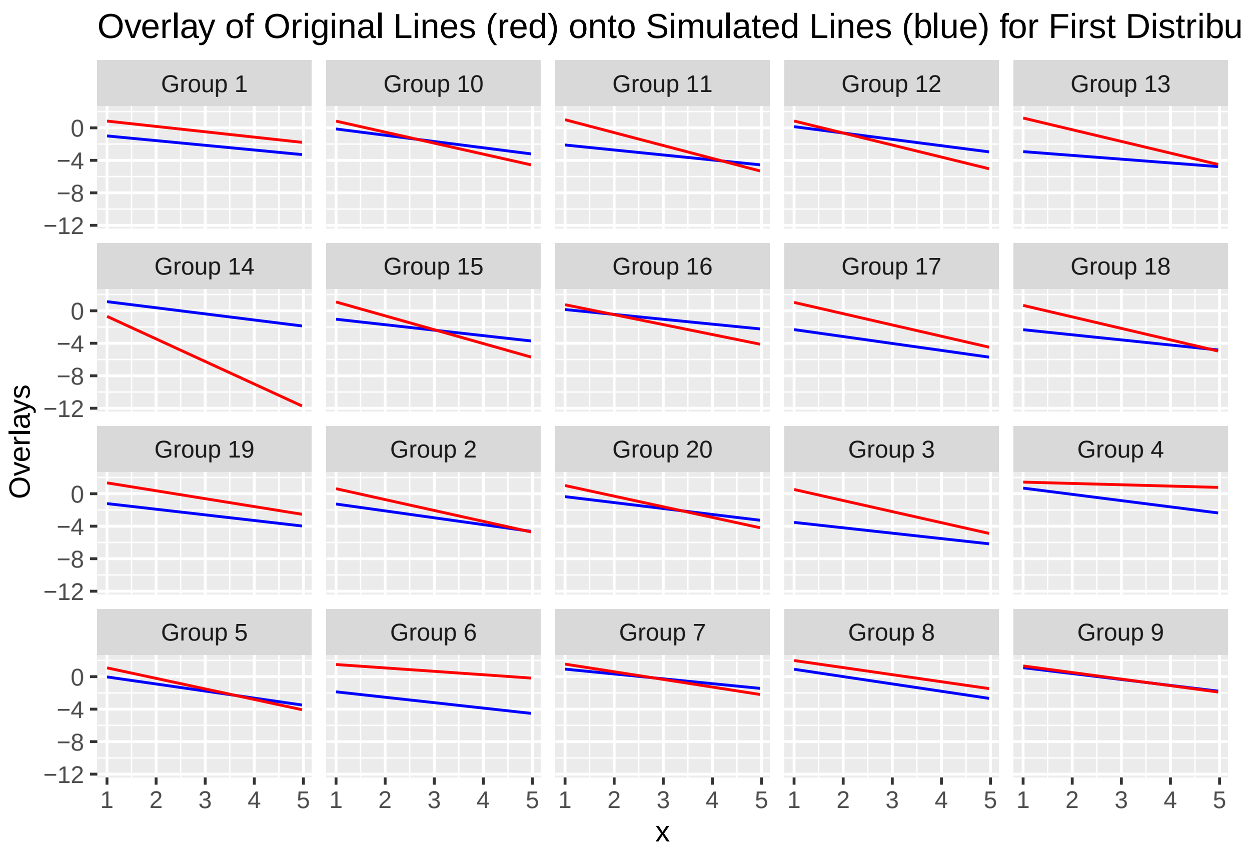
Question 3.2: Multilevel/Hierarchical Model

```
#Question 3.2
#Simulated Data from Handout
set.seed(17)

nu.mu <- 2
tau.mu <- 0.5
nu.beta <- -1
tau.beta <- 0.5
mu.hm <- rnorm(n=20, mean = nu.mu, sd=tau.mu)
```

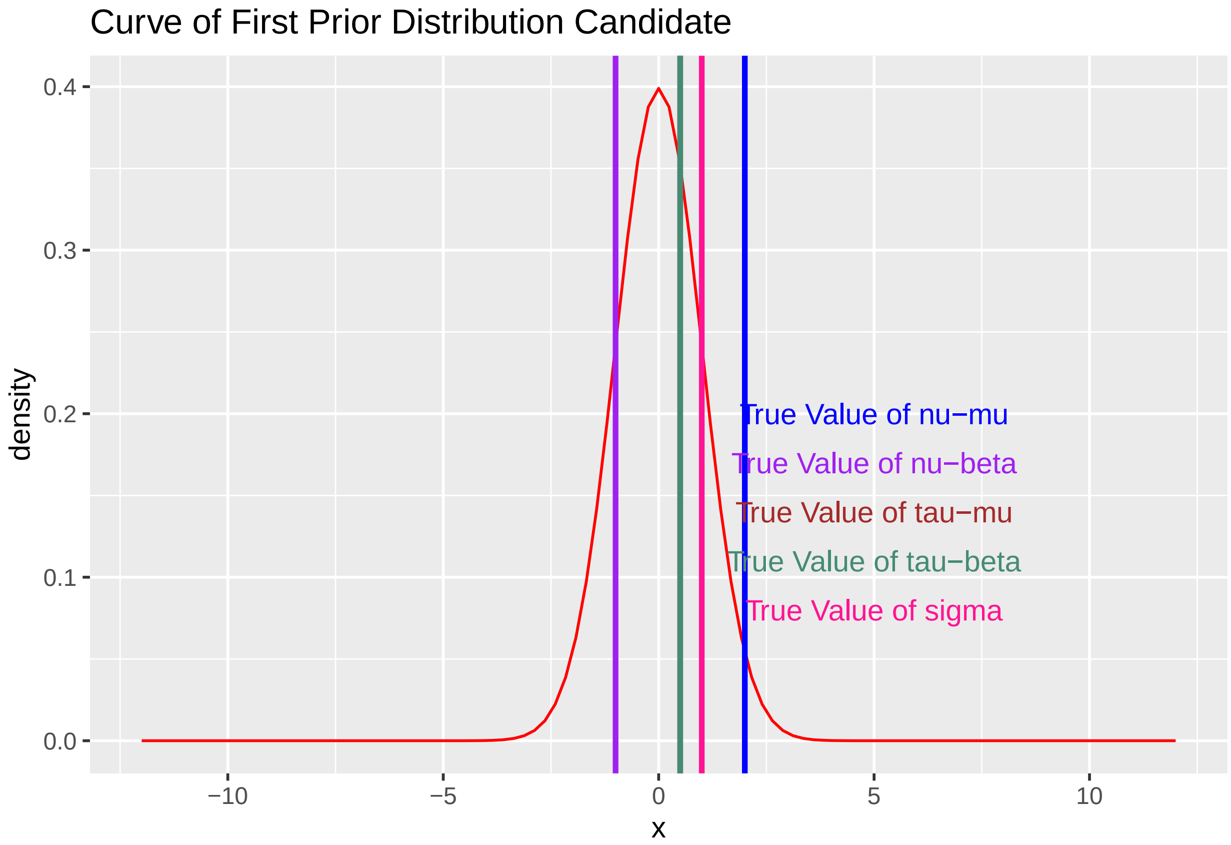
```
beta.hm <- rnorm(n=20, mean = nu.beta, sd= tau.beta)
sigma <- 1

x.hm <- runif(n = 100, min = 1, max=5)
y.mean.hier <- c(rep(mu.hm, each = 100) +
                  rep(beta.hm, each = 100)*
                  rep(x.hm, 20))
y.hier <- rnorm(n = 20*100, mean = y.mean.hier, sigma)
sim.data.hier <- tibble(x = rep(x.hm, 20), y.hier,
                        y.mean.hier,
                        group = paste("Group",
                        rep(1:20, each = 100)))
#--------------------------------------------------------------------------------
#First Prior Distribution
#--------------------------------------------------------------------------------
nu.mu.p1 <- rnorm(1, mean=0,sd=1)
tau.mu.p1 <- rgamma(1, shape=1,scale=1)
nu.beta.p1 <- rnorm(1, mean=0,sd=1)
tau.beta.p1 <- rgamma(1, shape=1,scale=1)
sigma.p1 <- rgamma(1, shape=1,scale=1)
mu.hm.p1 <- rnorm(n=20, mean = nu.mu.p1, sd=tau.mu.p1)
beta.hm.p1 <- rnorm(n=20, mean = nu.beta.p1, sd= tau.beta.p1)

y.mean.hier.p1 <- c(rep(mu.hm.p1, each = 100) +
                  rep(beta.hm.p1, each = 100)*
                  rep(x.hm, 20))
y.hier.p1 <- rnorm(n = 20*100, mean = y.mean.hier.p1, sigma.p1)
sim.data.hier.p1 <- tibble(x = rep(x.hm, 20), y.hier.p1,
                          y.mean.hier.p1,
                          group = paste("Group",
                          rep(1:20, each = 100)))

#Graph of Prior Predictive Datasets for First Prior Distribution
ggplot(data=sim.data.hier.p1, aes(x,y.hier.p1)) + geom_point() +
  geom_line(aes(x,y.mean.hier.p1), col="blue") +
  ylab("Simulated Data") +
  ggtitle("Prior Predictive Datasets for the First Prior Distribution Candidate") +
  facet_wrap(~group)
```



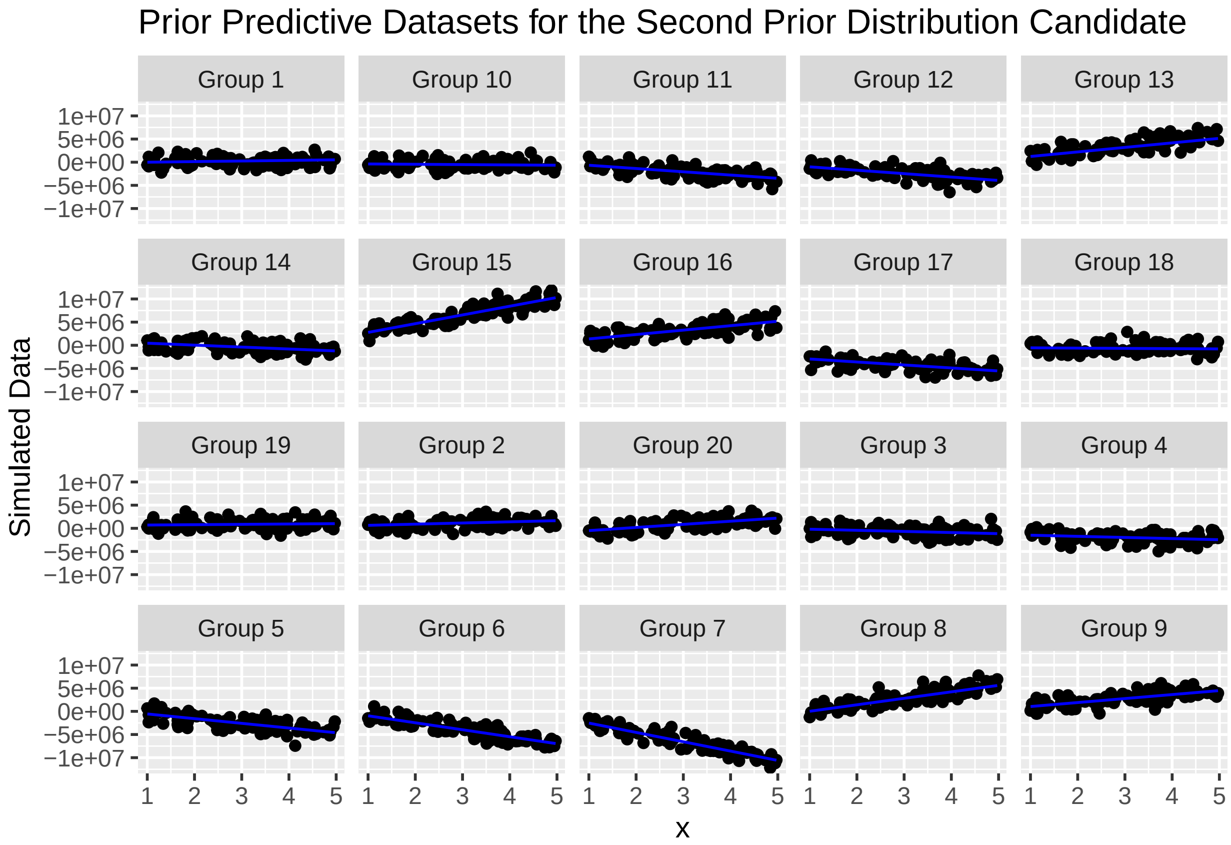
```r
#Overlay of Original Lines Onto Simulated Lines for First Prior Distribution
ggplot(data=sim.data.hier.p1, aes(x,y.hier.p1)) +
  geom_line(aes(x,y.mean.hier.p1), col="blue") +
  geom_line(aes(x,y.mean.hier), col="red") +
  ylab("Overlays") +
  ggtitle("Overlay of Original Lines (red) onto Simulated Lines (blue) for First Distribution") +
  facet_wrap(~group)
```



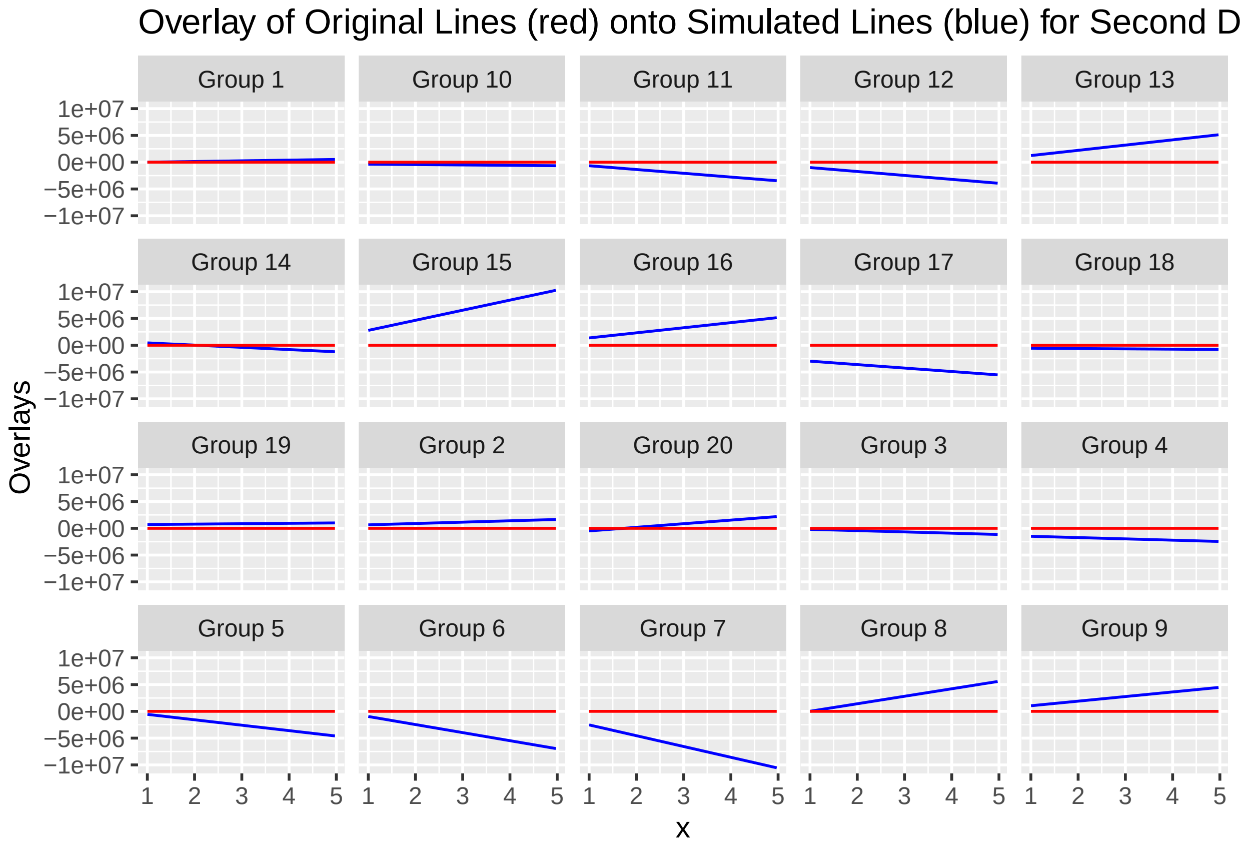
```
#Curve of First Prior Distribution
ggplot(data.frame(bx = seq(-12, 12, length.out = 100)), aes(bx)) +
  stat_function(fun = dnorm, args = list(mean=0,sd=1), colour="red") +
  geom_vline(xintercept = nu.mu, color="blue", size=1) +
  geom_vline(xintercept = nu.beta, color="purple", size=1) +
  geom_vline(xintercept = tau.mu, color="brown", size=1) +
  geom_vline(xintercept = tau.beta, color="aquamarine4", size=1) +
  geom_vline(xintercept = sigma, color="deeppink", size=1) +
  annotate("text", x=5,y=0.2, label="True Value of nu-mu", col="blue") +
  annotate("text", x=5,y=0.17, label="True Value of nu-beta", col="purple") +
  annotate("text", x=5,y=0.14, label="True Value of tau-mu", col="brown") +
  annotate("text", x=5,y=0.11, label="True Value of tau-beta", col="aquamarine4") +
  annotate("text", x=5,y=0.08, label="True Value of sigma", col="deeppink") +
  ylab("density") +
  xlab("x") +
  ggtitle("Curve of First Prior Distribution Candidate")
```



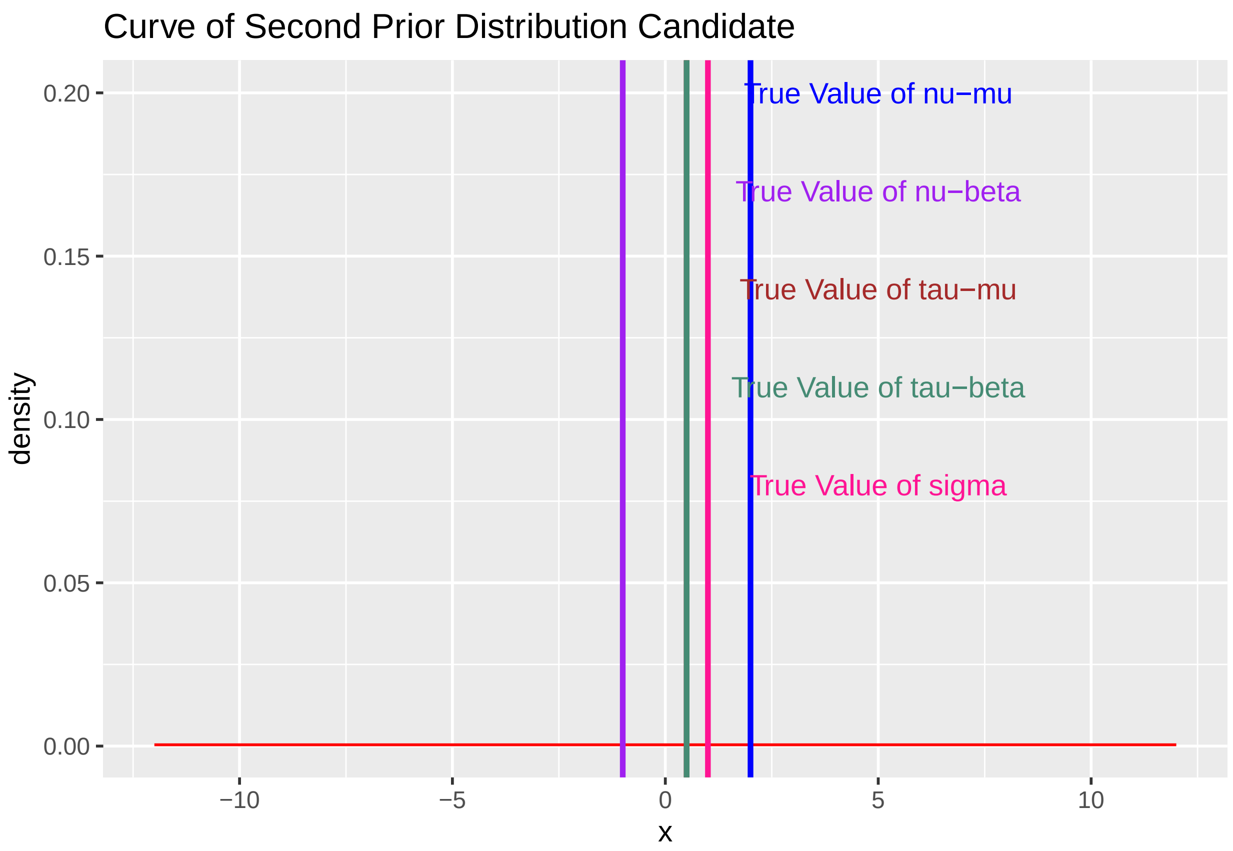
```
#-------------------------------------------------------------------------------
#Second Prior Distribution
#-------------------------------------------------------------------------------
nu.mu.p2 <- rnorm(1, mean=0,sd=1000)
tau.mu.p2 <- rgamma(1, shape=1000,scale=1000)
nu.beta.p2 <- rnorm(1, mean=0,sd=1000)
tau.beta.p2 <- rgamma(1, shape=1000,scale=1000)
sigma.p2 <- rgamma(1, shape=1000,scale=1000)
mu.hm.p2 <- rnorm(n=20, mean = nu.mu.p2, sd=tau.mu.p2)
beta.hm.p2 <- rnorm(n=20, mean = nu.beta.p2, sd= tau.beta.p2)

y.mean.hier.p2 <- c(rep(mu.hm.p2, each = 100) +
                  rep(beta.hm.p2, each = 100)*
                  rep(x.hm, 20))
y.hier.p2 <- rnorm(n = 20*100, mean = y.mean.hier.p2, sigma.p2)
sim.data.hier.p2 <- tibble(x = rep(x.hm, 20), y.hier.p2,
                          y.mean.hier.p2,
                          group = paste("Group",
                          rep(1:20, each = 100)))

#Graph of Prior Predictive Datasets for Second Prior Distribution
ggplot(data=sim.data.hier.p2, aes(x,y.hier.p2)) + geom_point() +
  geom_line(aes(x,y.mean.hier.p2), col="blue") +
  ylab("Simulated Data") +
  ggtitle("Prior Predictive Datasets for the Second Prior Distribution Candidate") +
  facet_wrap(~group)
```



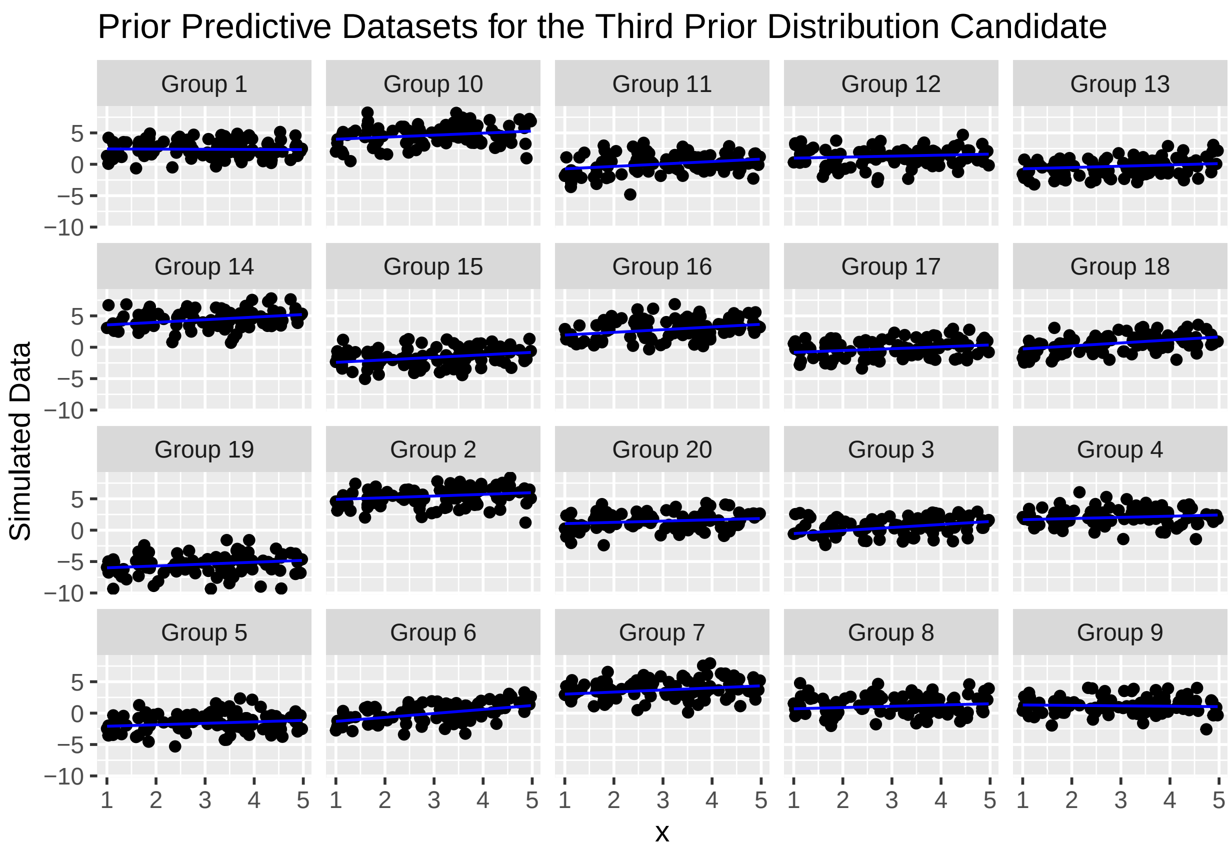
```
#Overlay of Original Lines Onto Simulated Lines for Second Prior Distribution
ggplot(data=sim.data.hier.p2, aes(x,y.hier.p2)) +
  geom_line(aes(x,y.mean.hier.p2), col="blue") +
  geom_line(aes(x,y.mean.hier), col="red") +
  ylab("Overlays") +
  ggtitle("Overlay of Original Lines (red) onto Simulated Lines (blue) for Second Distribution") +
  facet_wrap(~group)
```



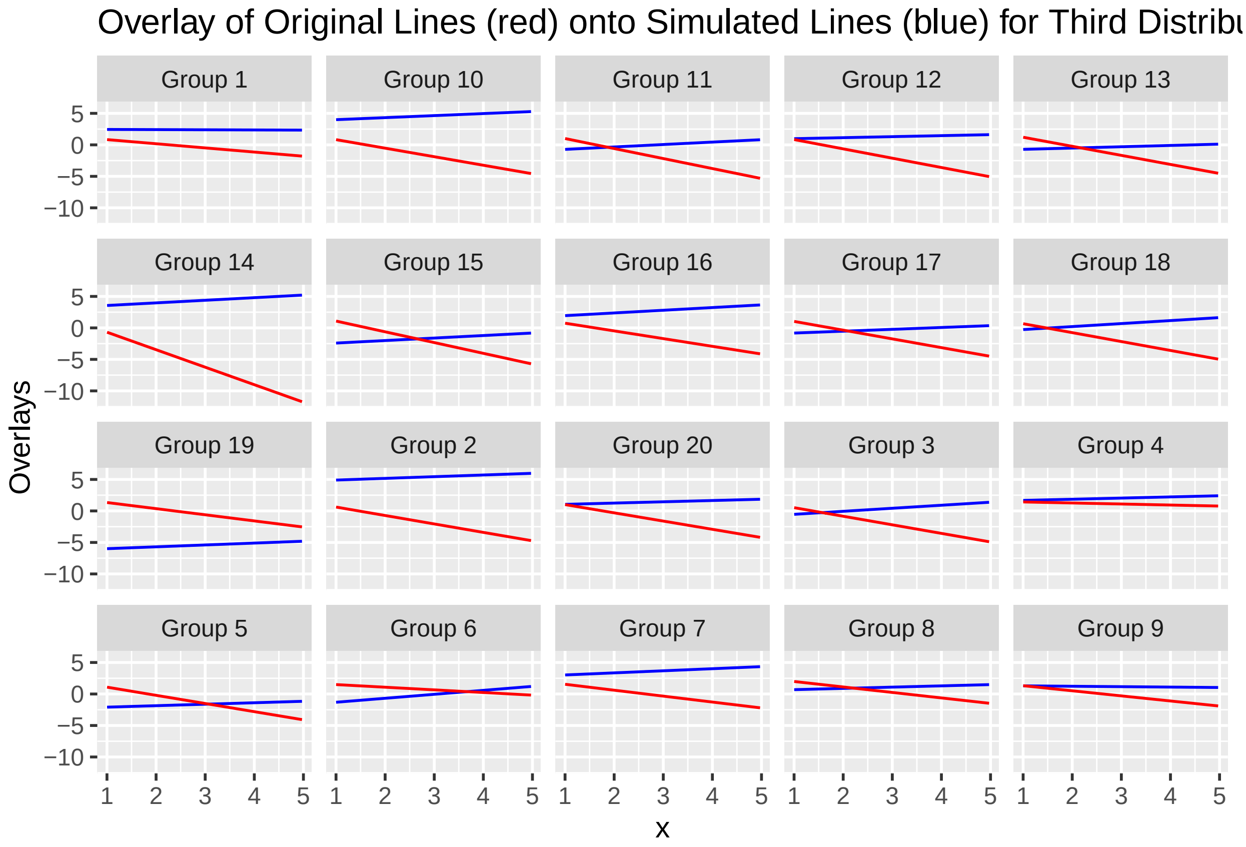
```
#Curve of Second Prior Distribution
ggplot(data.frame(bx = seq(-12, 12, length.out = 100)), aes(bx)) +
  stat_function(fun = dnorm, args = list(mean=0,sd=1000), colour="red") +
  geom_vline(xintercept = nu.mu, color="blue", size=1) +
  geom_vline(xintercept = nu.beta, color="purple", size=1) +
  geom_vline(xintercept = tau.mu, color="brown", size=1) +
  geom_vline(xintercept = tau.beta, color="aquamarine4", size=1) +
  geom_vline(xintercept = sigma, color="deeppink", size=1) +
  annotate("text", x=5,y=0.2, label="True Value of nu-mu", col="blue") +
  annotate("text", x=5,y=0.17, label="True Value of nu-beta", col="purple") +
  annotate("text", x=5,y=0.14, label="True Value of tau-mu", col="brown") +
  annotate("text", x=5,y=0.11, label="True Value of tau-beta", col="aquamarine4") +
  annotate("text", x=5,y=0.08, label="True Value of sigma", col="deeppink") +
  ylab("density") +
  xlab("x") +
  ggtitle("Curve of Second Prior Distribution Candidate")
```



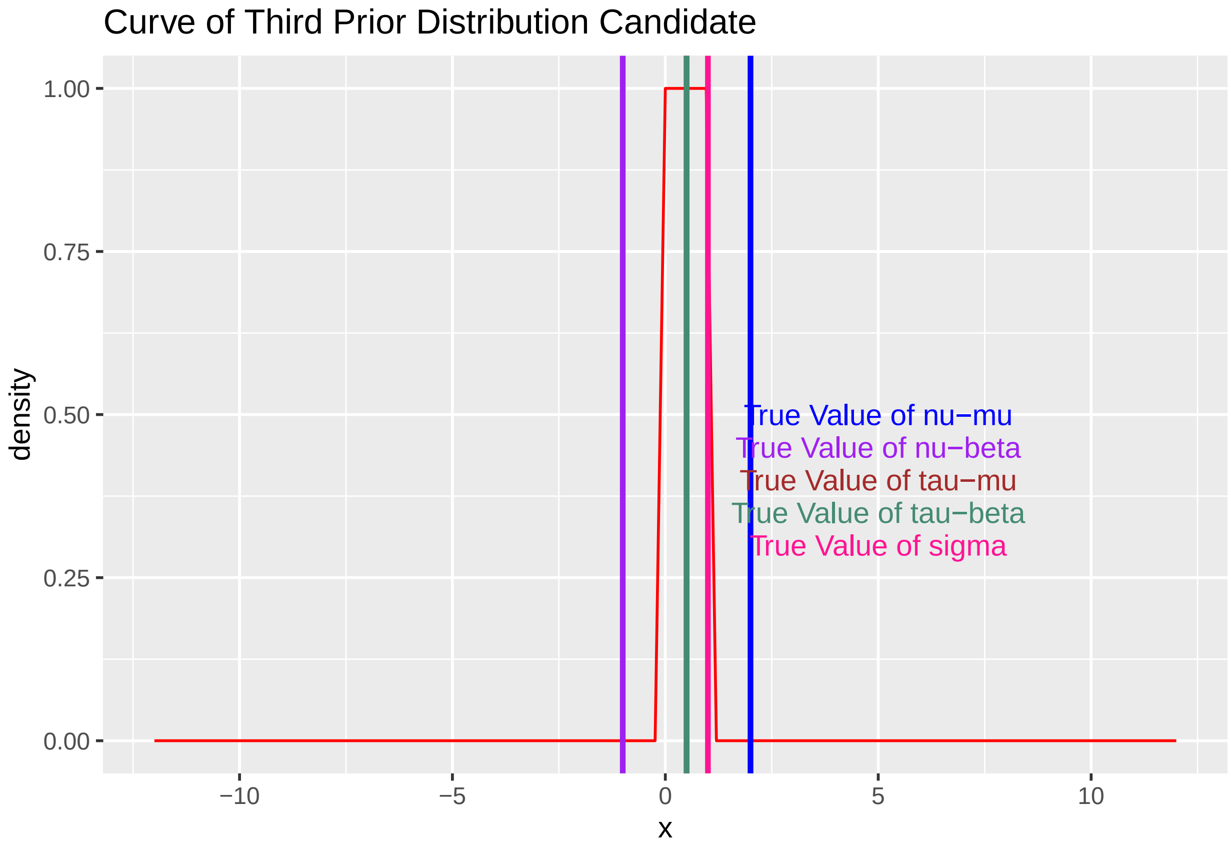
```r
#-------------------------------------------------------------------------------
#Third Prior Distribution
#-------------------------------------------------------------------------------
nu.mu.p3 <- runif(n = 1, min=0, max=1)
tau.mu.p3 <- rexp(n = 1, rate=1)
nu.beta.p3 <- runif(n = 1, min=0, max=1)
tau.beta.p3 <- rexp(n = 1, rate=1)
sigma.p3 <- rexp(n = 1, rate=1)
mu.hm.p3 <- rnorm(n=20, mean = nu.mu.p3, sd=tau.mu.p3)
beta.hm.p3 <- rnorm(n=20, mean = nu.beta.p3, sd= tau.beta.p3)

y.mean.hier.p3 <- c(rep(mu.hm.p3, each = 100) +
                  rep(beta.hm.p3, each = 100)*
                  rep(x.hm, 20))
y.hier.p3 <- rnorm(n = 20*100, mean = y.mean.hier.p3, sigma.p3)
sim.data.hier.p3 <- tibble(x = rep(x.hm, 20), y.hier.p3,
                          y.mean.hier.p3,
                          group = paste("Group",
                          rep(1:20, each = 100)))

#Graph of Prior Predictive Datasets for Third Prior Distribution
ggplot(data=sim.data.hier.p3, aes(x,y.hier.p3)) + geom_point() +
  geom_line(aes(x,y.mean.hier.p3), col="blue") +
  ylab("Simulated Data") +
  ggtitle("Prior Predictive Datasets for the Third Prior Distribution Candidate") +
  facet_wrap(~group)
```



```
#Overlay of Original Lines Onto Simulated Lines for Third Prior Distribution
ggplot(data=sim.data.hier.p3, aes(x,y.hier.p3)) +
  geom_line(aes(x,y.mean.hier.p3), col="blue") +
  geom_line(aes(x,y.mean.hier), col="red") +
  ylab("Overlays") +
  ggtitle("Overlay of Original Lines (red) onto Simulated Lines (blue) for Third Distribution") +
  facet_wrap(~group)
```



```
#Curve of Third Prior Distribution
ggplot(data.frame(bx = seq(-12, 12, length.out = 100)), aes(bx)) +
  stat_function(fun = dunif, args = list(min=0,max=1), colour="red") +
  geom_vline(xintercept = nu.mu, color="blue", size=1) +
  geom_vline(xintercept = nu.beta, color="purple", size=1) +
  geom_vline(xintercept = tau.mu, color="brown", size=1) +
  geom_vline(xintercept = tau.beta, color="aquamarine4", size=1) +
  geom_vline(xintercept = sigma, color="deeppink", size=1) +
  annotate("text", x=5,y=0.5, label="True Value of nu-mu", col="blue") +
  annotate("text", x=5,y=0.45, label="True Value of nu-beta", col="purple") +
  annotate("text", x=5,y=0.4, label="True Value of tau-mu", col="brown") +
  annotate("text", x=5,y=0.35, label="True Value of tau-beta", col="aquamarine4") +
  annotate("text", x=5,y=0.3, label="True Value of sigma", col="deeppink") +
  ylab("density") +
  xlab("x") +
  ggtitle("Curve of Third Prior Distribution Candidate")
```



The results from the simulation mostly lead us to the same conclusions as the ones we obtained in 3.1: As before, Choosing a noninformative prior such as Normal(0,1000) results in data that seems evenly spread along the line from which the data is generated with huge variations and no discernible patterns, meaning that it does not provide any specific information about the unknown parameters. This is reflected by the curve of the prior distribution, which shows a huge range of possible values with very flat density, making it very unlikely to get close to the true parameter values. Choosing a more informative prior such as Normal(0,1) results in data that has far less variance, and comparing original lines with the simulated lines show that the lines are very close to each other in most cases. Choosing a prior distribution such as Unif(0,1) led to simulated lines that often opposed the direction of the true line (negative slope vs. positive slope), which is expected since the true values of some parameters such as ν_β and ν_μ are not included between 0 and 1. This shows that we need have a good idea of the where the true value of the parameter is to ensure it is included in the prior distribution before we decide to choose strict priors such as Unif(0,1).

Question 4.1: Fitting a Linear Regression Model

```
#Question 4.1
load("bayes-vis.RData")
latcab <- GM[GM$super_region == 5,] # Data from Latin America/Carribean
```

We will use $\beta_0, \beta_1 \sim N(0,1)$ and $\tau \sim Gamma(0.01, 0.01)$ as the first prior:

```
#---------------------------------------------------------------------------------
#First Prior 4.1
#---------------------------------------------------------------------------------
```

```
prior.fixed <- list(mean.intercept = 0, prec.intercept = 1,
                    mean = 0, prec = 1)
prior.prec <- list(prec = list(prior = "loggamma",
                               param = c(0.01, 0.01)))

completepool <- inla(formula = pm25 ~ 1 + sat_2014,
                     data = data.frame(latcab),
                     control.fixed = prior.fixed,
                     control.family = list(hyper =
                                   list(prec = prior.prec)),
                     control.compute = list(config = TRUE))
summary(completepool)
```

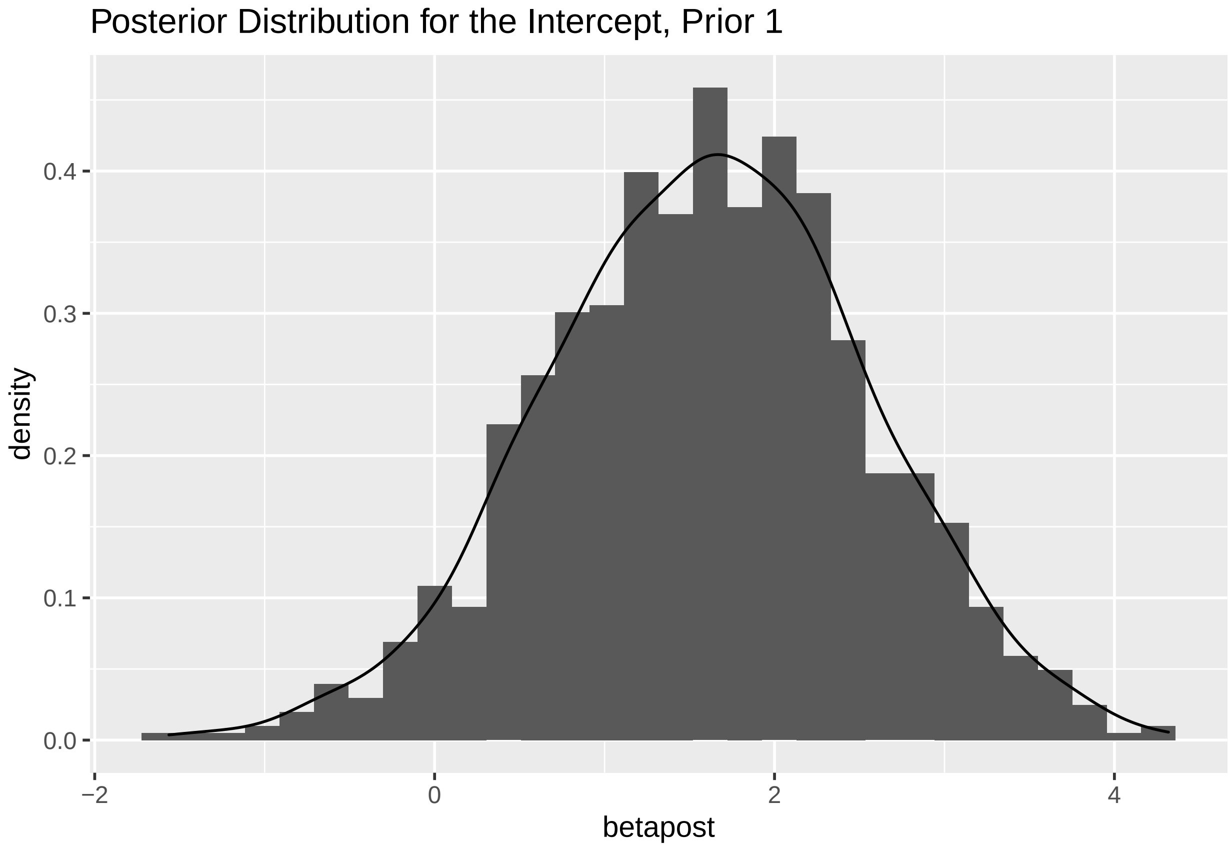
```
##
## Call:
##    c("inla.core(formula = formula, family = family, contrasts = contrasts,
##    ", " data = data, quantiles = quantiles, E = E, offset = offset, ", "
##    scale = scale, weights = weights, Ntrials = Ntrials, strata = strata,
##    ", " lp.scale = lp.scale, link.covariates = link.covariates, verbose =
##    verbose, ", " lincomb = lincomb, selection = selection, control.compute
##    = control.compute, ", " control.predictor = control.predictor,
##    control.family = control.family, ", " control.inla = control.inla,
##    control.fixed = control.fixed, ", " control.mode = control.mode,
##    control.expert = control.expert, ", " control.hazard = control.hazard,
##    control.lincomb = control.lincomb, ", " control.update =
##    control.update, control.lp.scale = control.lp.scale, ", "
##    control.pardiso = control.pardiso, only.hyperparam = only.hyperparam,
##    ", " inla.call = inla.call, inla.arg = inla.arg, num.threads =
##    num.threads, ", " blas.num.threads = blas.num.threads, keep = keep,
##    working.directory = working.directory, ", " silent = silent, inla.mode
##    = inla.mode, safe = FALSE, debug = debug, ", " .parent.frame =
##    .parent.frame)")
## Time used:
##     Pre = 0.635, Running = 0.125, Post = 0.0808, Total = 0.841
## Fixed effects:
##              mean    sd 0.025quant 0.5quant 0.975quant  mode kld
## (Intercept) 1.545 0.957     -0.335    1.546      3.420 1.548   0
## sat_2014    2.383 0.176      2.038    2.382      2.731 2.381   0
##
## Model hyperparameters:
##                                          mean    sd 0.025quant 0.5quant
## Precision for the Gaussian observations 0.011 0.002      0.009    0.011
##                                         0.975quant  mode
## Precision for the Gaussian observations      0.015 0.011
##
## Marginal log-Likelihood:  -398.53
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also ’control.compute=list(return.marginals.predictor=TRUE)’)
```

```
set.seed(465)
postdraws.cpool <- inla.posterior.sample(1000, completepool, seed=465)
```

```
## Warning in inla.posterior.sample(1000, completepool, seed = 465): Since ’seed!
## =0', parallel model is disabled and serial model is selected, num.threads='1:1'
```

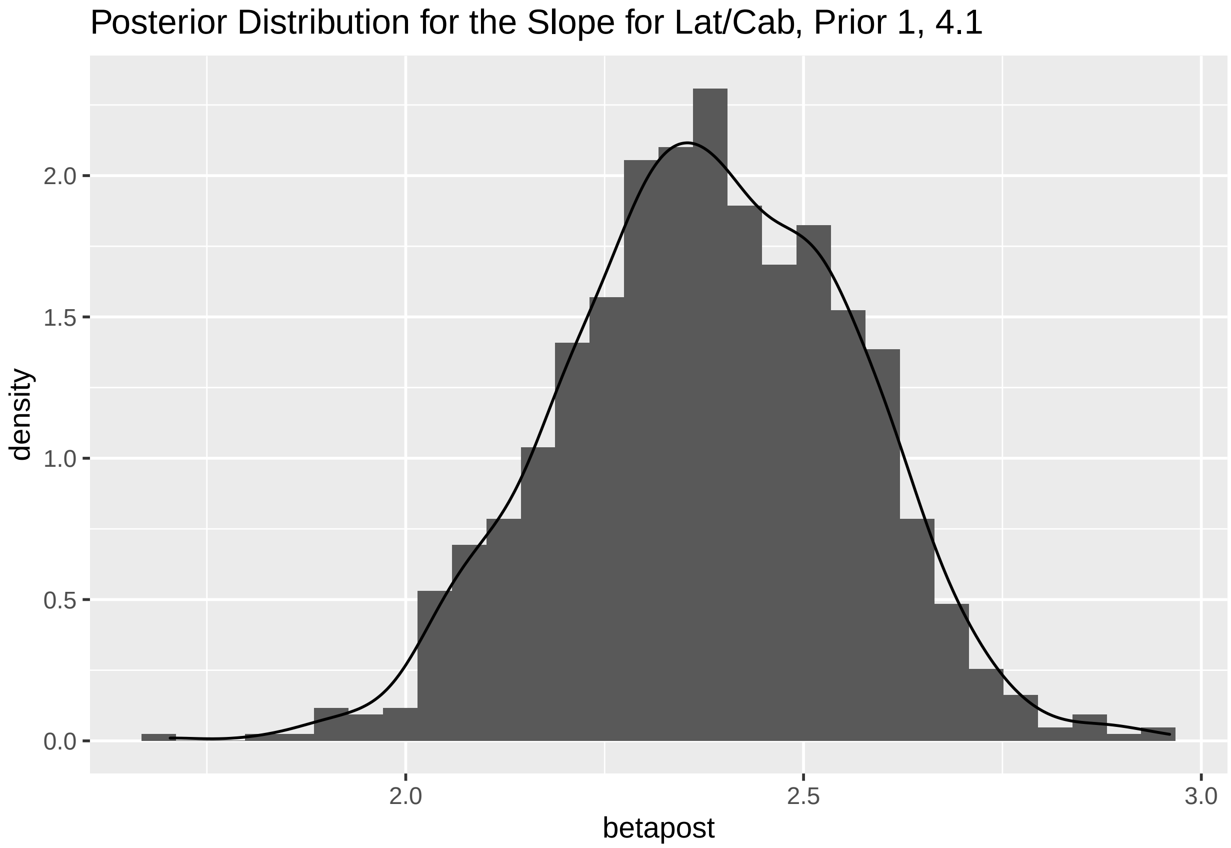
```
#Intercept
beta.post <- numeric(1000)
for(j in 1:1000){
  beta.post[j] <- postdraws.cpool[[j]]$latent[104]
}

ggplot(data = data.frame(betapost = beta.post), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("Posterior Distribution for the Intercept, Prior 1")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
#Slope
beta1.post <- numeric(1000)
for(j in 1:1000){
  beta1.post[j] <- postdraws.cpool[[j]]$latent[105]
}

ggplot(data = data.frame(betapost = beta1.post), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("Posterior Distribution for the Slope for Lat/Cab, Prior 1, 4.1")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
#Note
#latent[104] This corresponds to the intercept
#latent[105] This corresponds to the slope

#Organize Results in a Table
#Credible Interval: timestamp lab 2 20:12
cred.int <- data.frame(LowerBound = completepool$summary.fixed$`0.025quant`,
                       UpperBound = completepool$summary.fixed$`0.975quant`,
                       Estimate = completepool$summary.fixed$mode)
rownames(cred.int)<- c("Intercept","Slope")
cred.int %>%
  kable(
    caption = "95% Credible Intervals For Fixed Parameter Estimates, Prior 1, 4.1",
    col.names = c("Lower Bound", "Upper Bound", "Estimate"),
    row.names = TRUE,
    digits = 4,
    booktabs = TRUE
  )
```

Table 1: 95% Credible Intervals For Fixed Parameter Estimates, Prior 1, 4.1

	Lower Bound	Upper Bound	Estimate
Intercept	-0.3351	3.4197	1.5477
Slope	2.0378	2.7306	2.3813

We will use $\beta_0, \beta_1 \sim N(0, 10)$ and $\tau \sim Gamma(0.001, 0.001)$ as the second prior:

```r
#--------------------------------------------------------------------------------
#Second Prior 4.1
#--------------------------------------------------------------------------------
prior.fixed2 <- list(mean.intercept = 0, prec.intercept = 0.1,
                    mean = 0, prec = 0.1)
prior.prec2 <- list(prec = list(prior = "loggamma",
                               param = c(0.001, 0.001)))

completepool2 <- inla(formula = pm25 ~ 1 + sat_2014,
                     data = data.frame(latcab),
                     control.fixed = prior.fixed2,
                     control.family = list(hyper =
                                            list(prec = prior.prec2)),
                     control.compute = list(config = TRUE))
summary(completepool2)
```

```
##
## Call:
##    c("inla.core(formula = formula, family = family, contrasts = contrasts,
##    ", " data = data, quantiles = quantiles, E = E, offset = offset, ", "
##    scale = scale, weights = weights, Ntrials = Ntrials, strata = strata,
##    ", " lp.scale = lp.scale, link.covariates = link.covariates, verbose =
##    verbose, ", " lincomb = lincomb, selection = selection, control.compute
##    = control.compute, ", " control.predictor = control.predictor,
##    control.family = control.family, ", " control.inla = control.inla,
##    control.fixed = control.fixed, ", " control.mode = control.mode,
##    control.expert = control.expert, ", " control.hazard = control.hazard,
##    control.lincomb = control.lincomb, ", " control.update =
##    control.update, control.lp.scale = control.lp.scale, ", "
##    control.pardiso = control.pardiso, only.hyperparam = only.hyperparam,
##    ", " inla.call = inla.call, inla.arg = inla.arg, num.threads =
##    num.threads, ", " blas.num.threads = blas.num.threads, keep = keep,
##    working.directory = working.directory, ", " silent = silent, inla.mode
##    = inla.mode, safe = FALSE, debug = debug, ", " .parent.frame =
##    .parent.frame)")
## Time used:
##     Pre = 0.375, Running = 0.115, Post = 0.0778, Total = 0.567
## Fixed effects:
##              mean    sd 0.025quant 0.5quant 0.975quant  mode kld
## (Intercept) 7.243 2.195      2.884    7.259     11.508 7.293   0
## sat_2014    1.672 0.312      1.066    1.669      2.291 1.665   0
##
## Model hyperparameters:
```

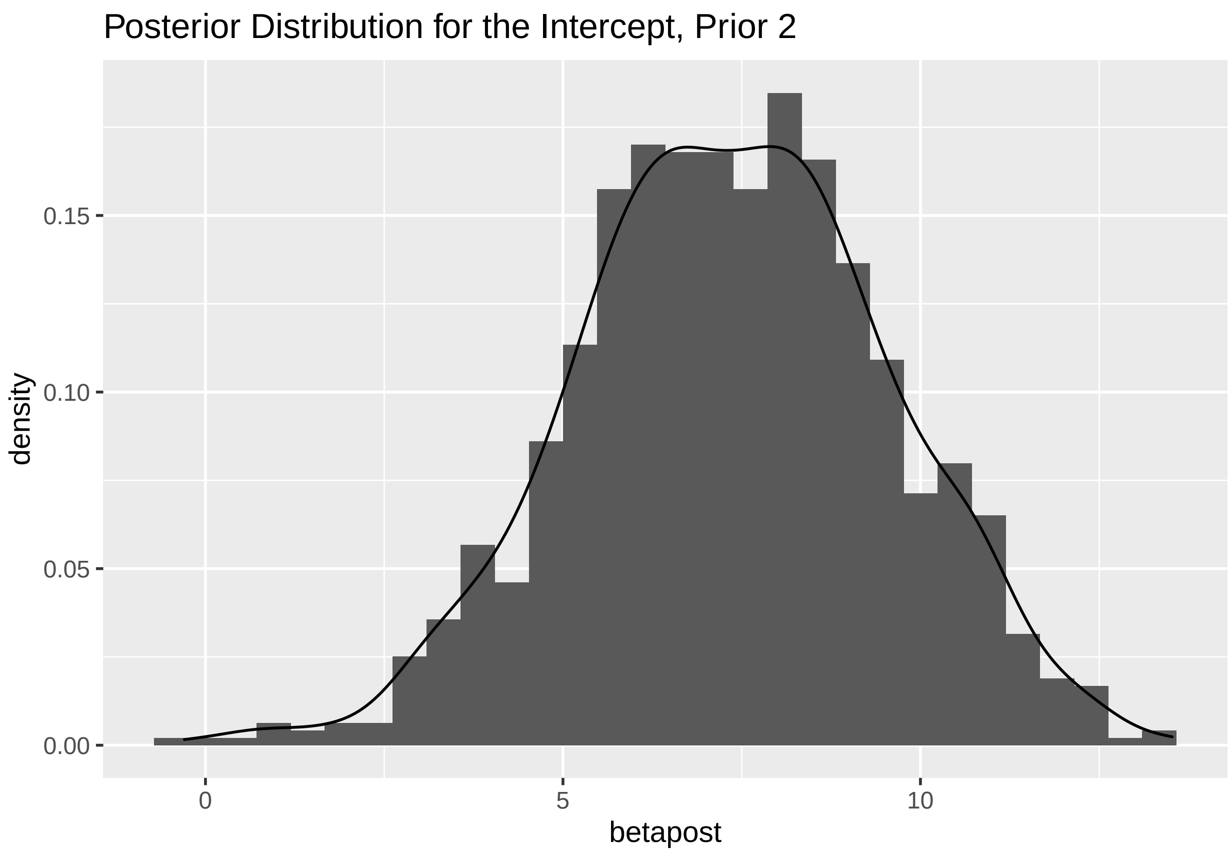
```
##                                          mean    sd 0.025quant 0.5quant
## Precision for the Gaussian observations 0.013 0.002      0.009    0.013
##                                         0.975quant  mode
## Precision for the Gaussian observations      0.016 0.012
##
## Marginal log-Likelihood:  -393.05
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
```

```
set.seed(465)
postdraws.cpool2 <- inla.posterior.sample(1000, completepool2, seed=465)
```

```
## Warning in inla.posterior.sample(1000, completepool2, seed = 465): Since 'seed!
## =0', parallel model is disabled and serial model is selected, num.threads='1:1'
```

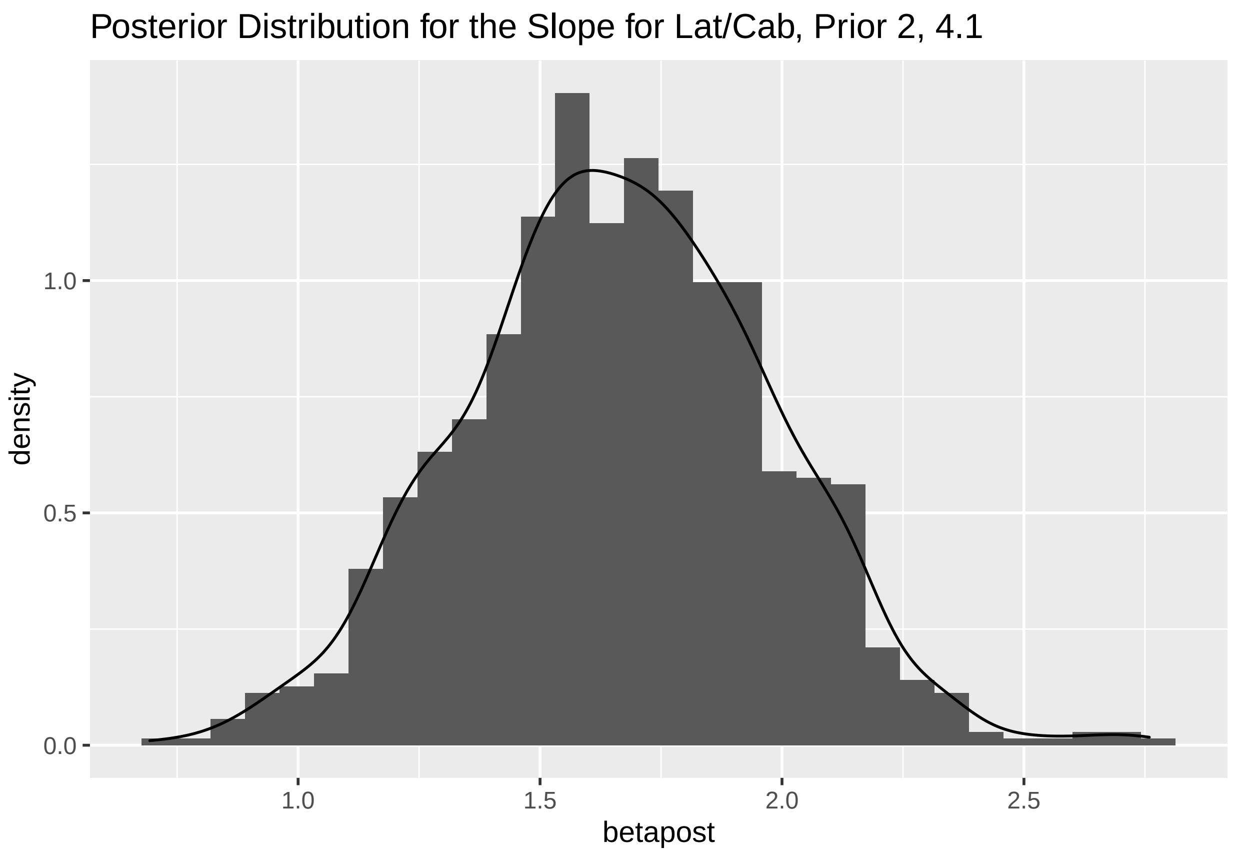
```
#intercept
beta.post2 <- numeric(1000)
for(j in 1:1000){
  beta.post2[j] <- postdraws.cpool2[[j]]$latent[104]
}

ggplot(data = data.frame(betapost = beta.post2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("Posterior Distribution for the Intercept, Prior 2")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
#Slope
beta1.post2 <- numeric(1000)
for(j in 1:1000){
  beta1.post2[j] <- postdraws.cpool2[[j]]$latent[105]
}

ggplot(data = data.frame(betapost = beta1.post2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("Posterior Distribution for the Slope for Lat/Cab, Prior 2, 4.1")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
#Note
#latent[104] This corresponds to the intercept
#latent[105] This corresponds to the slope

#Organize Results in a Table
#Credible Interval: timestamp lab 2 20:12
cred.int2 <- data.frame(LowerBound = completepool2$summary.fixed$`0.025quant`,
                       UpperBound = completepool2$summary.fixed$`0.975quant`,
                       Estimate = completepool2$summary.fixed$mode)
rownames(cred.int2)<- c("Intercept","Slope")
cred.int2 %>%
  kable(
    caption = "95% Credible Intervals For Fixed Parameter Estimates, Prior 2, 4.1",
    col.names = c("Lower Bound", "Upper Bound", "Estimate"),
    row.names = TRUE,
    digits = 4,
    booktabs = TRUE
  )
```

Table 2: 95% Credible Intervals For Fixed Parameter Estimates, Prior 2, 4.1

	Lower Bound	Upper Bound	Estimate
Intercept	2.8841	11.5082	7.2925
Slope	1.0662	2.2908	1.6646

Question 4.2: Fitting a Multilevel Regression Model

We will use $\beta_0, \beta_1 \sim N(0, 1)$ and $\tau \sim Gamma(0.01, 0.01), \tau_\mu \sim Gamma(0.01, 0.01), \tau_\beta \sim Gamma(0.01, 0.01)$ as the first prior:

```r
#Question 4.2
#Using more than just Latin America/Carribean data
#Add a column with numbers
GM$super_region_name_dup <- GM$super_region_name

#-------------------------------------------------------------------------------
#First Prior 4.2
#-------------------------------------------------------------------------------
prior.fixed.m1 <- list(mean.intercept = 0, prec.intercept = 1,
                    mean = 0, prec = 1)
prior.prec.m1 <- list(prec = list(prior = "loggamma",
                               param = c(0.01, 0.01)))

partialpool <- inla(formula = pm25 ~ 1 +
                       f(super_region_name,
                         model = "iid",
                         hyper = prior.prec.m1) +
                       f(super_region_name_dup, sat_2014,
                         model = "iid",
                         hyper = prior.prec.m1),
                     data = data.frame(GM),
                     control.fixed = prior.fixed.m1,
                     control.family = list(hyper =
                                             list(prec = prior.prec.m1)),
                    control.compute = list(config = TRUE))

summary(partialpool)
```

```
##
## Call:
##    c("inla.core(formula = formula, family = family, contrasts = contrasts,
##    ", " data = data, quantiles = quantiles, E = E, offset = offset, ", "
##    scale = scale, weights = weights, Ntrials = Ntrials, strata = strata,
##    ", " lp.scale = lp.scale, link.covariates = link.covariates, verbose =
##    verbose, ", " lincomb = lincomb, selection = selection, control.compute
##    = control.compute, ", " control.predictor = control.predictor,
##    control.family = control.family, ", " control.inla = control.inla,
##    control.fixed = control.fixed, ", " control.mode = control.mode,
##    control.expert = control.expert, ", " control.hazard = control.hazard,
##    control.lincomb = control.lincomb, ", " control.update =
##    control.update, control.lp.scale = control.lp.scale, ", "
##    control.pardiso = control.pardiso, only.hyperparam = only.hyperparam,
##    ", " inla.call = inla.call, inla.arg = inla.arg, num.threads =
##    num.threads, ", " blas.num.threads = blas.num.threads, keep = keep,
##    working.directory = working.directory, ", " silent = silent, inla.mode
##    = inla.mode, safe = FALSE, debug = debug, ", " .parent.frame =
##    .parent.frame)")
## Time used:
##     Pre = 0.394, Running = 2.44, Post = 0.0799, Total = 2.91
## Fixed effects:
##              mean    sd 0.025quant 0.5quant 0.975quant  mode kld
## (Intercept) 0.488 0.987     -1.451    0.488      2.422 0.489   0
##
## Random effects:
##   Name     Model
##     super_region_name IID model
##    super_region_name_dup IID model
##
## Model hyperparameters:
##                                           mean    sd 0.025quant 0.5quant
## Precision for the Gaussian observations 0.010 0.001      0.009    0.010
## Precision for super_region_name         0.003 0.002      0.000    0.003
## Precision for super_region_name_dup       Inf   NaN      0.000    0.000
##                                         0.975quant  mode
## Precision for the Gaussian observations      0.011 0.010
## Precision for super_region_name              0.009 0.001
## Precision for super_region_name_dup            Inf   NaN
##
## Marginal log-Likelihood:  -11179.43
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
```

```r
set.seed(465)
postdraws.ppool <- inla.posterior.sample(1000, partialpool, seed=465)
```

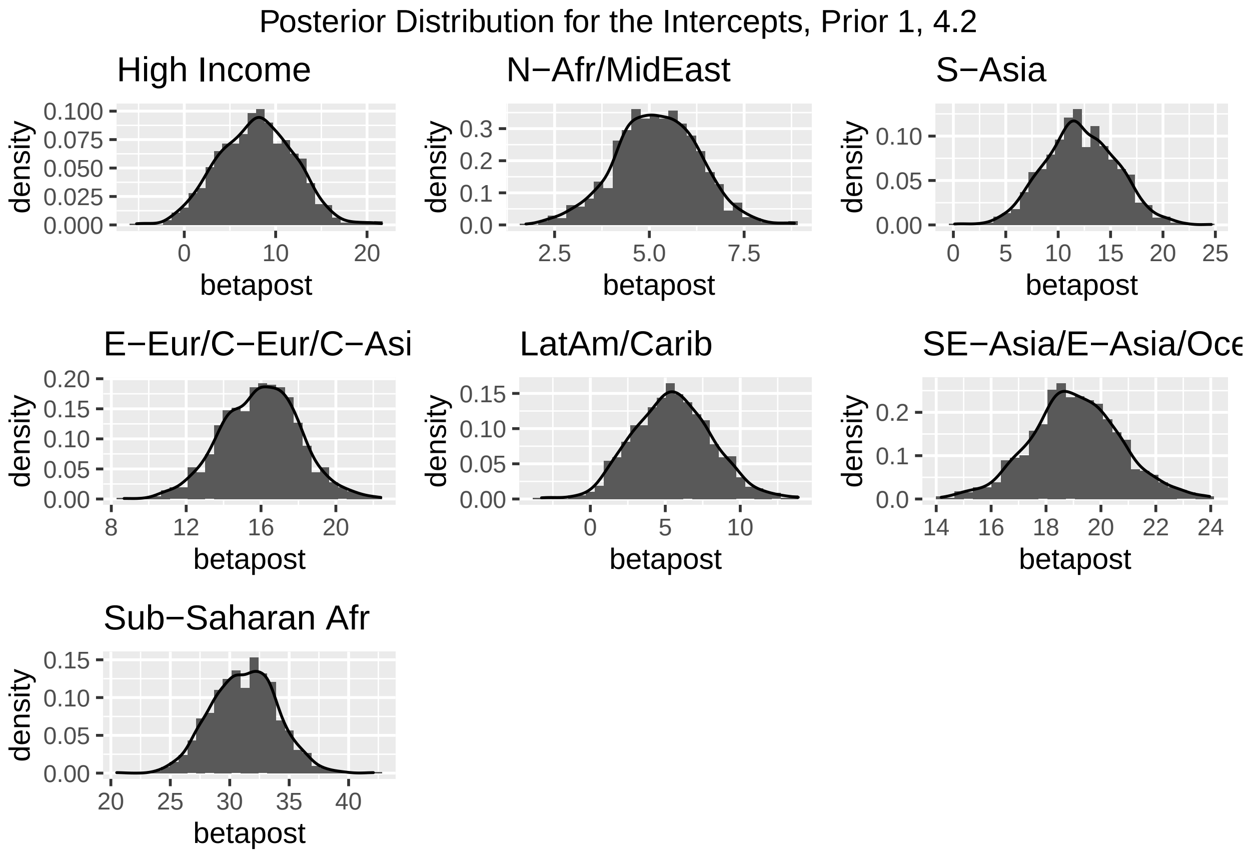
```
## Warning in inla.posterior.sample(1000, partialpool, seed = 465): Since 'seed!
## =0', parallel model is disabled and serial model is selected, num.threads='1:1'
```

```r
#Intercept 1
beta01.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta01.post.m1[j] <- postdraws.ppool[[j]]$latent[2981]
}

plot1 <- ggplot(data = data.frame(betapost = beta01.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("High Income")

#Intercept 2
beta02.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta02.post.m1[j] <- postdraws.ppool[[j]]$latent[2982]
}

plot2 <- ggplot(data = data.frame(betapost = beta02.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("N-Afr/MidEast")

#Intercept 3
beta03.post.m1 <- numeric(1000)
```

```r
for(j in 1:1000){
  beta03.post.m1[j] <- postdraws.ppool[[j]]$latent[2983]
}

plot3 <- ggplot(data = data.frame(betapost = beta03.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("S-Asia")

#Intercept 4
beta04.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta04.post.m1[j] <- postdraws.ppool[[j]]$latent[2984]
}

plot4 <- ggplot(data = data.frame(betapost = beta04.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("E-Eur/C-Eur/C-Asia")

#Intercept 5
beta05.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta05.post.m1[j] <- postdraws.ppool[[j]]$latent[2985]
}

plot5 <- ggplot(data = data.frame(betapost = beta05.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("LatAm/Carib")

#Intercept 6
beta06.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta06.post.m1[j] <- postdraws.ppool[[j]]$latent[2986]
}

plot6 <- ggplot(data = data.frame(betapost = beta06.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("SE-Asia/E-Asia/Oceania")

#Intercept 7
beta07.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta07.post.m1[j] <- postdraws.ppool[[j]]$latent[2987]
}

plot7 <- ggplot(data = data.frame(betapost = beta07.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("Sub-Saharan Afr")
```

```
grid.arrange(plot1,plot2,plot3,plot4,plot5,plot6,plot7,ncol=3,
             top="Posterior Distribution for the Intercepts, Prior 1, 4.2")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

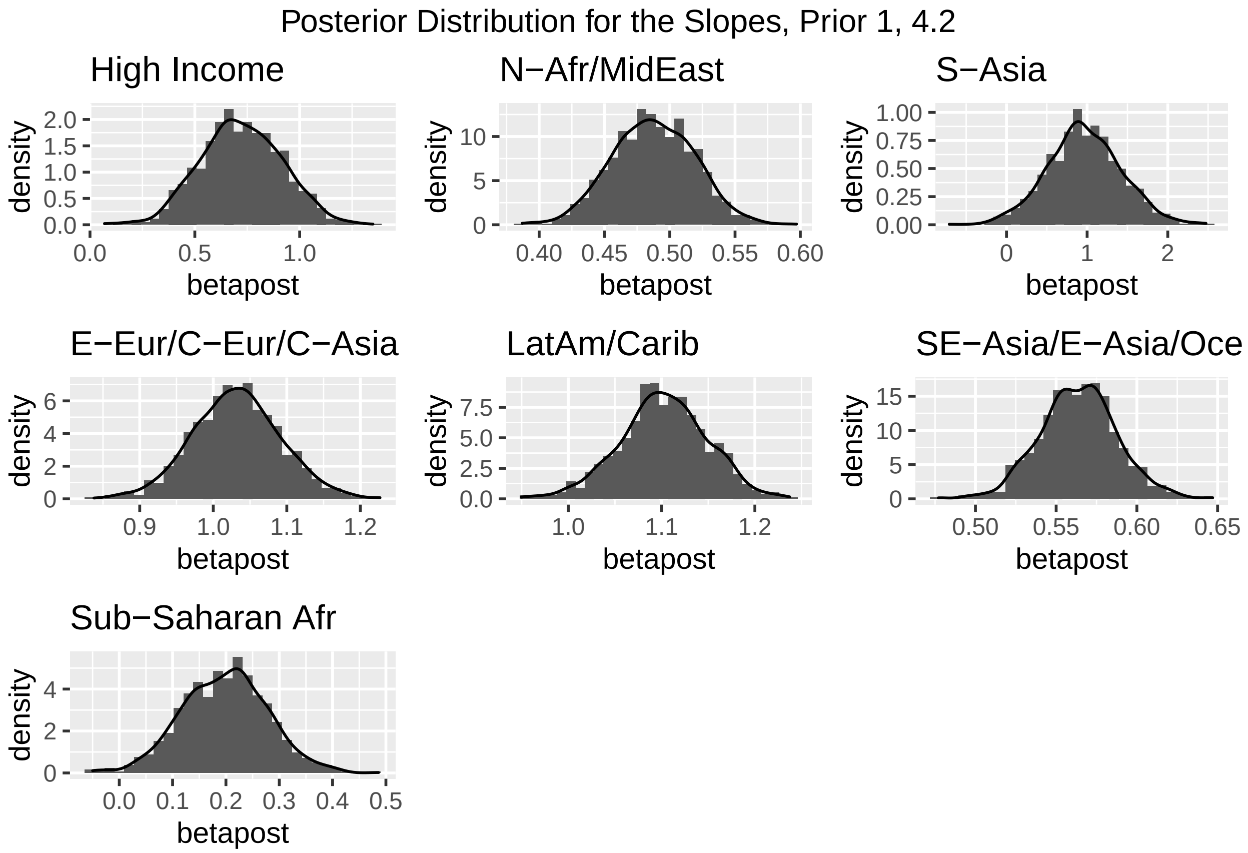


```
#Slope 1
beta11.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta11.post.m1[j] <- postdraws.ppool[[j]]$latent[2988]
}

plot1 <- ggplot(data = data.frame(betapost = beta11.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("High Income")

#Slope 2
beta12.post.m1 <- numeric(1000)
```

```
for(j in 1:1000){
  beta12.post.m1[j] <- postdraws.ppool[[j]]$latent[2989]
}

plot2 <- ggplot(data = data.frame(betapost = beta12.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("N-Afr/MidEast")

#Slope 3
beta13.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta13.post.m1[j] <- postdraws.ppool[[j]]$latent[2990]
}

plot3 <- ggplot(data = data.frame(betapost = beta13.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("S-Asia")

#Slope 4
beta14.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta14.post.m1[j] <- postdraws.ppool[[j]]$latent[2991]
}

plot4 <- ggplot(data = data.frame(betapost = beta14.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("E-Eur/C-Eur/C-Asia")

#Slope 5
beta15.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta15.post.m1[j] <- postdraws.ppool[[j]]$latent[2992]
}

plot5 <- ggplot(data = data.frame(betapost = beta15.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("LatAm/Carib")

#Slope 6
beta16.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta16.post.m1[j] <- postdraws.ppool[[j]]$latent[2993]
}

plot6 <- ggplot(data = data.frame(betapost = beta16.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("SE-Asia/E-Asia/Oceania")
```

```
#Slope 7
beta17.post.m1 <- numeric(1000)
for(j in 1:1000){
  beta17.post.m1[j] <- postdraws.ppool[[j]]$latent[2994]
}

plot7 <- ggplot(data = data.frame(betapost = beta17.post.m1), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("Sub-Saharan Afr")

grid.arrange(plot1,plot2,plot3,plot4,plot5,plot6,plot7,ncol=3,
             top="Posterior Distribution for the Slopes, Prior 1, 4.2")
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



```
#Note
#latent[2995] corresponds to the intercept
#latent[2988-2994] corresponds to the 7 different slopes
```

```
#Organize Results in a Table
#Different intercepts and slopes for each group
#super_region_name are the intercepts!
#super_region_name_dup are the slopes!
cred.int.m1 <- data.frame(LowerBound = c(partialpool$summary.random$super_region_name$`0.025quant`,
                                         partialpool$summary.random$super_region_name_dup$`0.025quant`)
                       UpperBound = c(partialpool$summary.random$super_region_name$`0.975quant`,
                                      partialpool$summary.random$super_region_name_dup$`0.975quant`),
                       Estimate = c(partialpool$summary.random$super_region_name$mode,partialpool$summar
rownames(cred.int.m1)<- c("Intercept1","Intercept2","Intercept3","Intercept4",
                          "Intercept5","Intercept6","Intercept7",
                          "High Income(1)","N-Afr/MidEast(2)","S-Asia(3)","E-Eur/C-Eur/C-Asia(4)",
                          "LatAm/Carib(5)","SE-Asia/E-Asia/Oceania(6)","Sub-Saharan Afr(7)")

cred.int.m1 %>%
  kable(
    caption = "95% Credible Intervals For Parameter Estimates, Prior 1, 4.2",
    col.names = c("Lower Bound", "Upper Bound", "Estimate"),
    row.names = TRUE,
    digits = 4,
    booktabs = TRUE
  )
```

Table 3: 95% Credible Intervals For Parameter Estimates, Prior 1, 4.2

	Lower Bound	Upper Bound	Estimate
Intercept1	-0.1829	16.1606	7.9834
Intercept2	3.1025	7.3806	5.2404
Intercept3	5.4860	18.8766	12.1774
Intercept4	11.9669	20.0646	16.0149
Intercept5	0.1927	10.7489	5.4706
Intercept6	15.8250	22.2687	19.0460
Intercept7	25.7595	36.6534	31.1991
High Income(1)	0.3318	1.1039	0.7182
N-Afr/MidEast(2)	0.4240	0.5488	0.4864
S-Asia(3)	0.0754	1.8485	0.9627
E-Eur/C-Eur/C-Asia(4)	0.9136	1.1475	1.0306
LatAm/Carib(5)	1.0097	1.1957	1.1028
SE-Asia/E-Asia/Oceania(6)	0.5184	0.6103	0.5644
Sub-Saharan Afr(7)	0.0448	0.3536	0.1993

We will use $\beta_0, \beta_1 \sim N(0, 1/0.1)$ and $\tau \sim Gamma(0.001, 0.001), \tau_\mu \sim Gamma(0.001, 0.001), \tau_\beta \sim Gamma(0.001, 0.001)$ as the second prior:

```
#-------------------------------------------------------------------------------
#Second Prior 4.2
#-------------------------------------------------------------------------------
GM$super_region_name_dup <- GM$super_region_name

prior.fixed.m2 <- list(mean.intercept = 0, prec.intercept = 0.1,
                    mean = 0, prec = 0.1)
prior.prec.m2 <- list(prec = list(prior = "loggamma",
                               param = c(0.001, 0.001)))

partialpool2 <- inla(formula = pm25 ~ 1 +
                       f(super_region_name,
                         model = "iid",
                         hyper = prior.prec.m2) +
                       f(super_region_name_dup, sat_2014,
                         model = "iid",
                         hyper = prior.prec.m2),
                     data = data.frame(GM),
                     control.fixed = prior.fixed.m2,
                     control.family = list(hyper =
                                             list(prec = prior.prec.m2)),
                    control.compute = list(config = TRUE))

summary(partialpool2)
```

```
##
## Call:
##    c("inla.core(formula = formula, family = family, contrasts = contrasts,
##    ", " data = data, quantiles = quantiles, E = E, offset = offset, ", "
##    scale = scale, weights = weights, Ntrials = Ntrials, strata = strata,
##    ", " lp.scale = lp.scale, link.covariates = link.covariates, verbose =
##    verbose, ", " lincomb = lincomb, selection = selection, control.compute
##    = control.compute, ", " control.predictor = control.predictor,
##    control.family = control.family, ", " control.inla = control.inla,
##    control.fixed = control.fixed, ", " control.mode = control.mode,
##    control.expert = control.expert, ", " control.hazard = control.hazard,
##    control.lincomb = control.lincomb, ", " control.update =
##    control.update, control.lp.scale = control.lp.scale, ", "
##    control.pardiso = control.pardiso, only.hyperparam = only.hyperparam,
##    ", " inla.call = inla.call, inla.arg = inla.arg, num.threads =
##    num.threads, ", " blas.num.threads = blas.num.threads, keep = keep,
##    working.directory = working.directory, ", " silent = silent, inla.mode
##    = inla.mode, safe = FALSE, debug = debug, ", " .parent.frame =
##    .parent.frame)")
## Time used:
##     Pre = 0.375, Running = 1.25, Post = 0.0773, Total = 1.7
## Fixed effects:
##               mean    sd 0.025quant 0.5quant 0.975quant mode kld
## (Intercept) 3.882 3.223     -2.562    3.938     10.004 4.05   0
##
## Random effects:
##   Name     Model
##     super_region_name IID model
##    super_region_name_dup IID model
##
## Model hyperparameters:
##                                           mean    sd 0.025quant 0.5quant
## Precision for the Gaussian observations 0.010 0.000      0.009    0.010
## Precision for super_region_name         0.006 0.004      0.001    0.005
## Precision for super_region_name_dup     1.849 1.030      0.470    1.655
##                                         0.975quant  mode
## Precision for the Gaussian observations      0.010 0.010
## Precision for super_region_name              0.016 0.003
## Precision for super_region_name_dup          4.373 1.204
##
## Marginal log-Likelihood:  -11184.20
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also ’control.compute=list(return.marginals.predictor=TRUE)’)
```

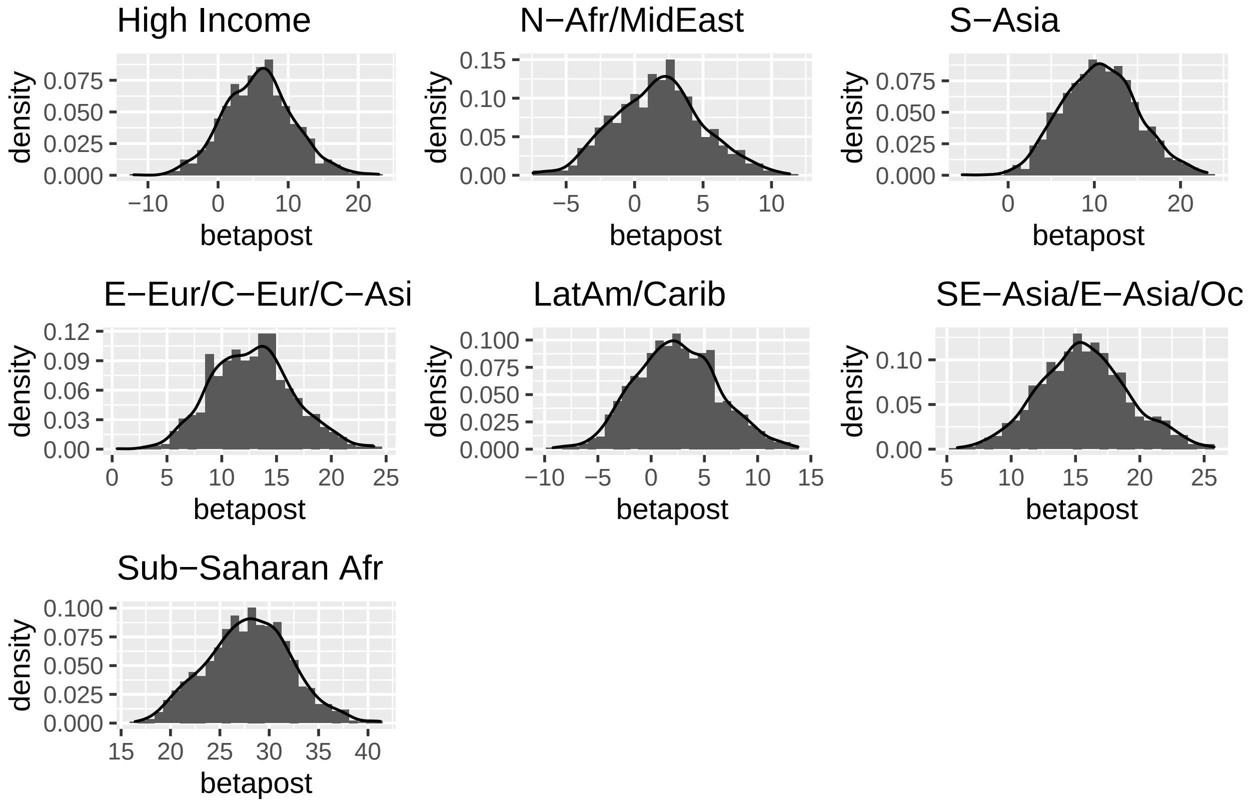
```
set.seed(465)
postdraws.ppool2 <- inla.posterior.sample(1000, partialpool2, seed=465)
```

```
## Warning in inla.posterior.sample(1000, partialpool2, seed = 465): Since ’seed!
## =0’, parallel model is disabled and serial model is selected, num.threads=’1:1’
```

```
#Intercept 1
beta01.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta01.post.m2[j] <- postdraws.ppool2[[j]]$latent[2981]
}

plot1 <- ggplot(data = data.frame(betapost = beta01.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("High Income")

#Intercept 2
beta02.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta02.post.m2[j] <- postdraws.ppool2[[j]]$latent[2982]
}

plot2 <- ggplot(data = data.frame(betapost = beta02.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("N-Afr/MidEast")

#Intercept 3
beta03.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta03.post.m2[j] <- postdraws.ppool2[[j]]$latent[2983]
}

plot3 <- ggplot(data = data.frame(betapost = beta03.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("S-Asia")

#Intercept 4
beta04.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta04.post.m2[j] <- postdraws.ppool2[[j]]$latent[2984]
}

plot4 <- ggplot(data = data.frame(betapost = beta04.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("E-Eur/C-Eur/C-Asia")

#Intercept 5
beta05.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta05.post.m2[j] <- postdraws.ppool2[[j]]$latent[2985]
}

plot5 <- ggplot(data = data.frame(betapost = beta05.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("LatAm/Carib")

#Intercept 6
beta06.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta06.post.m2[j] <- postdraws.ppool2[[j]]$latent[2986]
}

plot6 <- ggplot(data = data.frame(betapost = beta06.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("SE-Asia/E-Asia/Oceania")

#Intercept 7
beta07.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta07.post.m2[j] <- postdraws.ppool2[[j]]$latent[2987]
}

plot7 <- ggplot(data = data.frame(betapost = beta07.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("Sub-Saharan Afr")

grid.arrange(plot1,plot2,plot3,plot4,plot5,plot6,plot7,ncol=3,
             top="Posterior Distribution for the Intercepts, Prior 2, 4.2")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

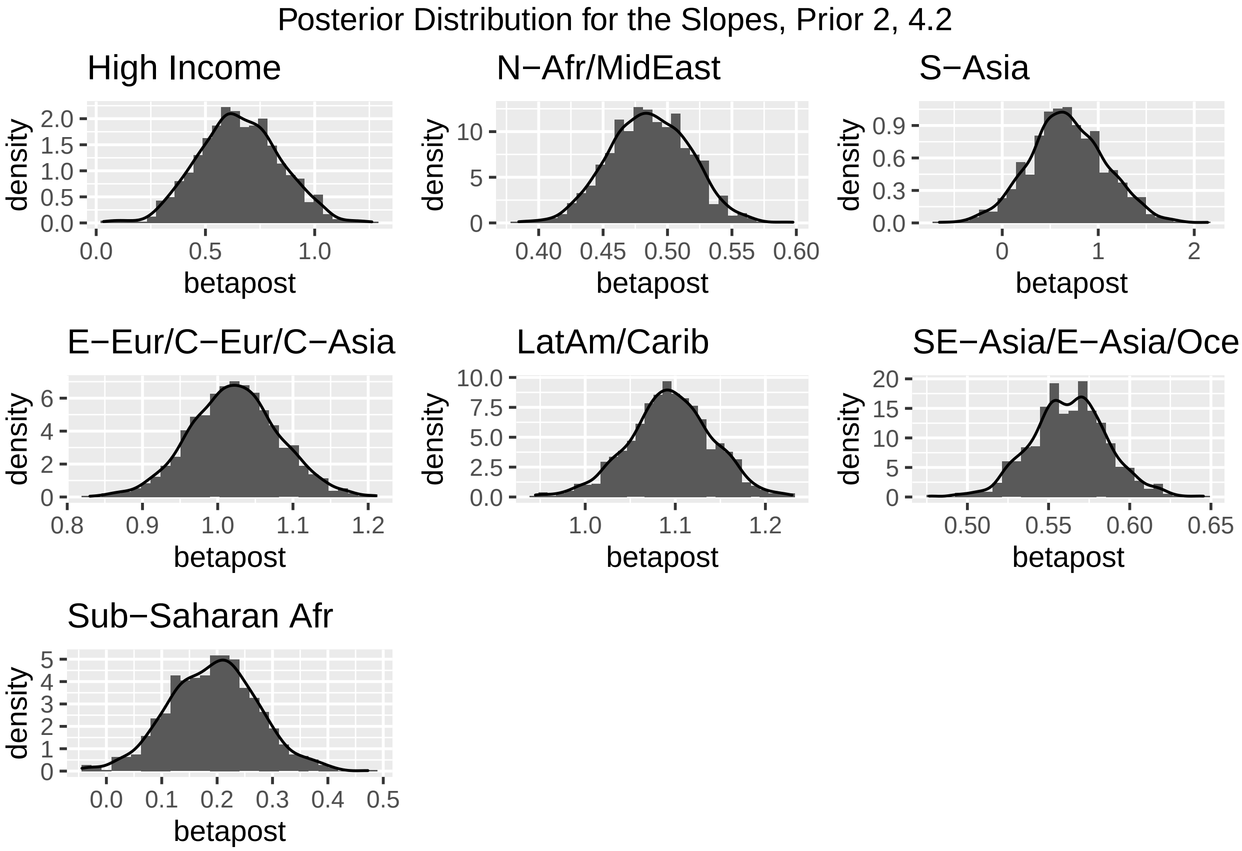
Posterior Distribution for the Intercepts, Prior 2, 4.2



```
#Slope 1
beta11.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta11.post.m2[j] <- postdraws.ppool2[[j]]$latent[2988]
}

plot1 <- ggplot(data = data.frame(betapost = beta11.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("High Income")

#Slope 2
beta12.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta12.post.m2[j] <- postdraws.ppool2[[j]]$latent[2989]
}

plot2 <- ggplot(data = data.frame(betapost = beta12.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("N-Afr/MidEast")

#Slope 3
beta13.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta13.post.m2[j] <- postdraws.ppool2[[j]]$latent[2990]
```

```r
}

plot3 <- ggplot(data = data.frame(betapost = beta13.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("S-Asia")

#Slope 4
beta14.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta14.post.m2[j] <- postdraws.ppool2[[j]]$latent[2991]
}

plot4 <- ggplot(data = data.frame(betapost = beta14.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("E-Eur/C-Eur/C-Asia")

#Slope 5
beta15.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta15.post.m2[j] <- postdraws.ppool2[[j]]$latent[2992]
}

plot5 <- ggplot(data = data.frame(betapost = beta15.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("LatAm/Carib")

#Slope 6
beta16.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta16.post.m2[j] <- postdraws.ppool2[[j]]$latent[2993]
}

plot6 <- ggplot(data = data.frame(betapost = beta16.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("SE-Asia/E-Asia/Oceania")

#Slope 7
beta17.post.m2 <- numeric(1000)
for(j in 1:1000){
  beta17.post.m2[j] <- postdraws.ppool2[[j]]$latent[2994]
}

plot7 <- ggplot(data = data.frame(betapost = beta17.post.m2), aes(betapost)) +
  geom_histogram(aes(y=..density..)) +
  geom_density() +
  ggtitle("Sub-Saharan Afr")

grid.arrange(plot1,plot2,plot3,plot4,plot5,plot6,plot7,ncol=3,
             top="Posterior Distribution for the Slopes, Prior 2, 4.2")
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



```
#Note
#latent[2995] corresponds to the 7 different intercept
#latent[2988-2994] corresponds to the 7 different slopes

#Organize Results in a Table
#Different intercepts and slopes for each group
#super_region_name are the intercepts!
#super_region_name_dup are the slopes!
cred.int.m2 <- data.frame(LowerBound = c(partialpool2$summary.random$super_region_name$`0.025quant`,
                                         partialpool2$summary.random$super_region_name_dup$`0.025quant`)
                      UpperBound = c(partialpool2$summary.random$super_region_name$`0.975quant`,
                                     partialpool2$summary.random$super_region_name_dup$`0.975quant`),
                      Estimate = c(partialpool2$summary.random$super_region_name$mode,partialpool2$summ
rownames(cred.int.m2)<- c("Intercept1","Intercept2","Intercept3","Intercept4",
                          "Intercept5","Intercept6","Intercept7",
                          "High Income(1)","N-Afr/MidEast(2)","S-Asia(3)","E-Eur/C-Eur/C-Asia(4)",
                          "LatAm/Carib(5)","SE-Asia/E-Asia/Oceania(6)","Sub-Saharan Afr(7)")
cred.int.m2 %>%
kable(
  caption = "95% Credible Intervals For Parameter Estimates, Prior 2, 4.2",
  col.names = c("Lower Bound", "Upper Bound", "Estimate"),
  row.names = TRUE,
  digits = 4,
  booktabs = TRUE
)
```

Table 4: 95% Credible Intervals For Parameter Estimates, Prior 2, 4.2

	Lower Bound	Upper Bound	Estimate
Intercept1	-3.6440	15.9069	5.7229
Intercept2	-4.3131	8.3621	1.6949
Intercept3	2.3702	19.3392	10.6441
Intercept4	5.7848	20.2517	12.7256
Intercept5	-5.1994	10.4460	2.2693
Intercept6	9.0442	22.6686	15.5589
Intercept7	19.6141	36.4553	27.8934
High Income(1)	0.2821	1.0241	0.6518
N-Afr/MidEast(2)	0.4235	0.5468	0.4852
S-Asia(3)	-0.0741	1.4716	0.6602
E-Eur/C-Eur/C-Asia(4)	0.9060	1.1373	1.0217
LatAm/Carib(5)	1.0035	1.1873	1.0954
SE-Asia/E-Asia/Oceania(6)	0.5178	0.6088	0.5633
Sub-Saharan Afr(7)	0.0412	0.3486	0.1950

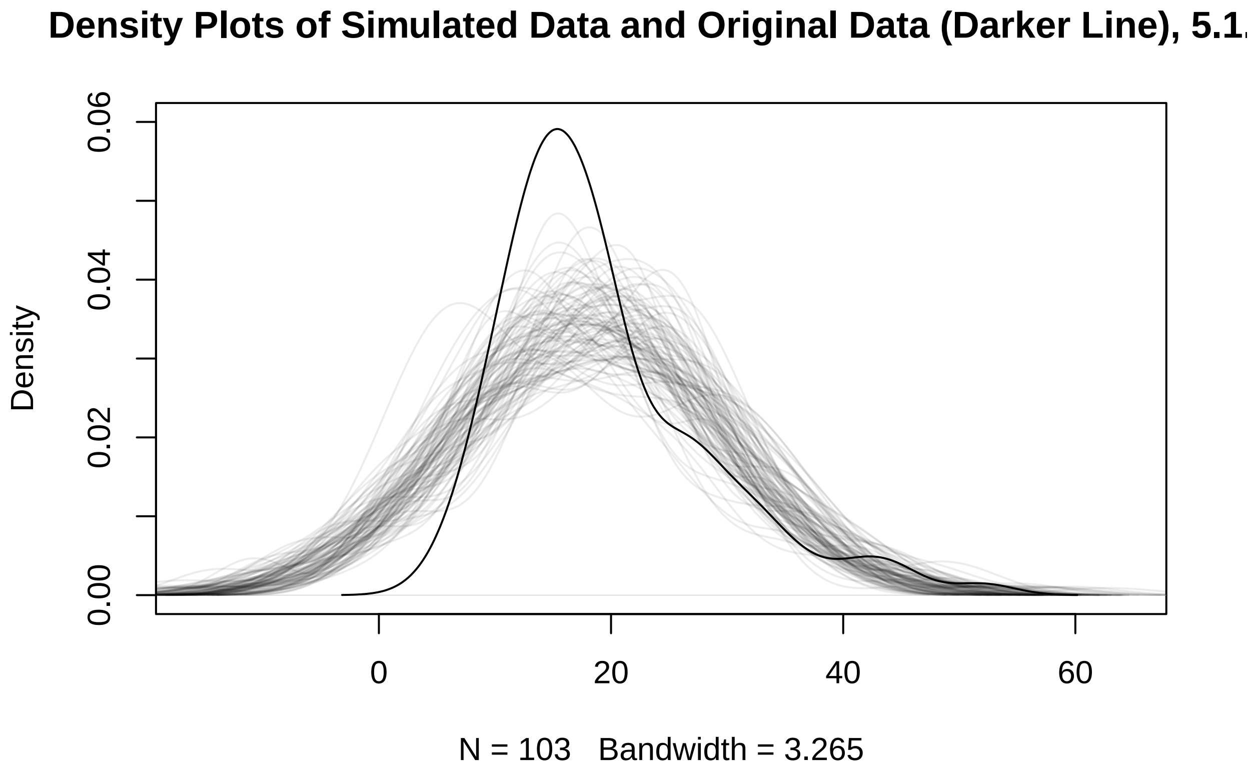
Question 5.1

```
#Question 5.1
#--------------------------------------------------------------------------------
#Using Estimates from 4.1 Prior 1
#--------------------------------------------------------------------------------

#---REDACTED---
#beta0 <- completepool$summary.fixed$mode[1]
#beta1 <- completepool$summary.fixed$mode[2]
#sigma <- sqrt(1/completepool$summary.hyperpar$mode) # the precision, tau = 1/sigma^2
#---REDACTED---

#Sample 100 draws from the posterior distribution
set.seed(465)
postdraws.cpool.511 <- inla.posterior.sample(100, completepool, seed=465)
```

```
## Warning in inla.posterior.sample(100, completepool, seed = 465): Since 'seed!
## =0', parallel model is disabled and serial model is selected, num.threads='1:1'
```

```
#Intercept
beta0.post.511 <- numeric(100)
```

```r
for(j in 1:100){
  beta0.post.511[j] <- postdraws.cpool.511[[j]]$latent[104]
}
#Slope
beta1.post.511 <- numeric(100)
for(j in 1:100){
  beta1.post.511[j] <- postdraws.cpool.511[[j]]$latent[105]
}
sigma <- sqrt(1/completepool$summary.hyperpar$mode) # the precision, tau = 1/sigma^2

#Generate 100 datasets of size 103
sim.data.511 <- data.frame(x = rep(latcab$sat_2014, 100),
                           y.mean = rep(NA, 100*103),
                           y = rep(NA, 100*103),
                           group = rep(1:100, each = 103))

for(j in 1:100){
  y.mean.511.sim <- beta0.post.511[j] + beta1.post.511[j]*latcab$sat_2014
  y.511.sim <- rnorm(n = length(latcab),
                     mean = y.mean.511.sim,
                     sd=sigma)
  sim.data.511$y.mean[sim.data.511$group == j] <- y.mean.511.sim
  sim.data.511$y[sim.data.511$group == j] <- y.511.sim
}

#Draw Density Curves
#for some reason ggplot doesn't want to separate the densities...
dat <- filter(sim.data.511, group==1)
plot(density(dat$y), col = alpha("black", 0.08),
     main="Density Plots of Simulated Data and Original Data (Darker Line), 5.1.1",
     ylim=c(0,0.06))
for(i in 2:100){
  dat <- filter(sim.data.511, group==i)
  lines(density(dat$y), col = alpha("black", 0.08))
}
#the original density
lines(density(latcab$pm25))
```



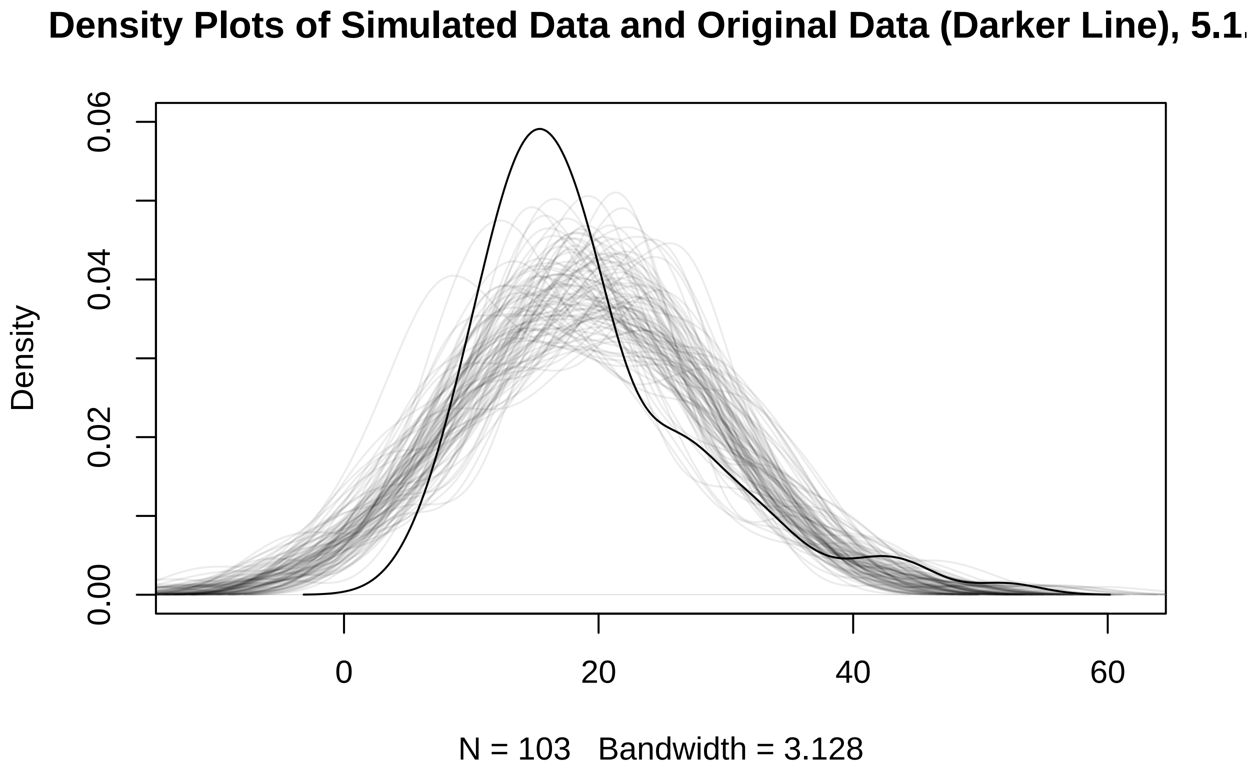
For the first prior, the right tail of the simulated density plots seem to coincide well with the original data. The simulated densities' peaks mostly align with the original data's as well, though they are less pronounced and the densities overall are less right-skewed than the original data.

```
#------------------------------------------------------------------------------
#Using Estimates from 4.1 Prior 2
#------------------------------------------------------------------------------

#Sample 100 draws from the posterior distribution
set.seed(465)
postdraws.cpool.512 <- inla.posterior.sample(100, completepool2, seed=465)
```

```
## Warning in inla.posterior.sample(100, completepool2, seed = 465): Since 'seed!
## =0', parallel model is disabled and serial model is selected, num.threads='1:1'
```

```
#Intercept
beta0.post.512 <- numeric(100)
for(j in 1:100){
  beta0.post.512[j] <- postdraws.cpool.512[[j]]$latent[104]
}
#Slope
beta1.post.512 <- numeric(100)
for(j in 1:100){
  beta1.post.512[j] <- postdraws.cpool.512[[j]]$latent[105]
}
sigma <- sqrt(1/completepool2$summary.hyperpar$mode) # the precision, tau = 1/sigma^2
```

```
#Generate 100 datasets of size 103
sim.data.512 <- data.frame(x = rep(latcab$sat_2014, 100),
                           y.mean = rep(NA, 100*103),
                           y = rep(NA, 100*103),
                           group = rep(1:100, each = 103))

for(j in 1:100){
  y.mean.512.sim <- beta0.post.512[j] + beta1.post.512[j]*latcab$sat_2014
  y.512.sim <- rnorm(n = length(latcab),
                    mean = y.mean.512.sim,
                    sd=sigma)
  sim.data.512$y.mean[sim.data.512$group == j] <- y.mean.512.sim
  sim.data.512$y[sim.data.512$group == j] <- y.512.sim
}

#Draw Density Curves
#for some reason ggplot doesn't want to separate the densities...
dat <- filter(sim.data.512, group==1)
plot(density(dat$y), col = alpha("black", 0.08),
     main="Density Plots of Simulated Data and Original Data (Darker Line), 5.1.2",
     ylim=c(0,0.06))
for(i in 2:100){
  dat <- filter(sim.data.512, group==i)
  lines(density(dat$y), col = alpha("black", 0.08))
}
#the original density
lines(density(latcab$pm25))
```



For the second prior, the simulated data seem to get closer to the original data than those from the first prior: The right tail of the simulated density plots seem to align well with the original data and the gap between the left tail of the simulated and the original data is smaller. The simulated densities' peaks overall seem more centered than right-skewed than the simulated densities from the first prior.

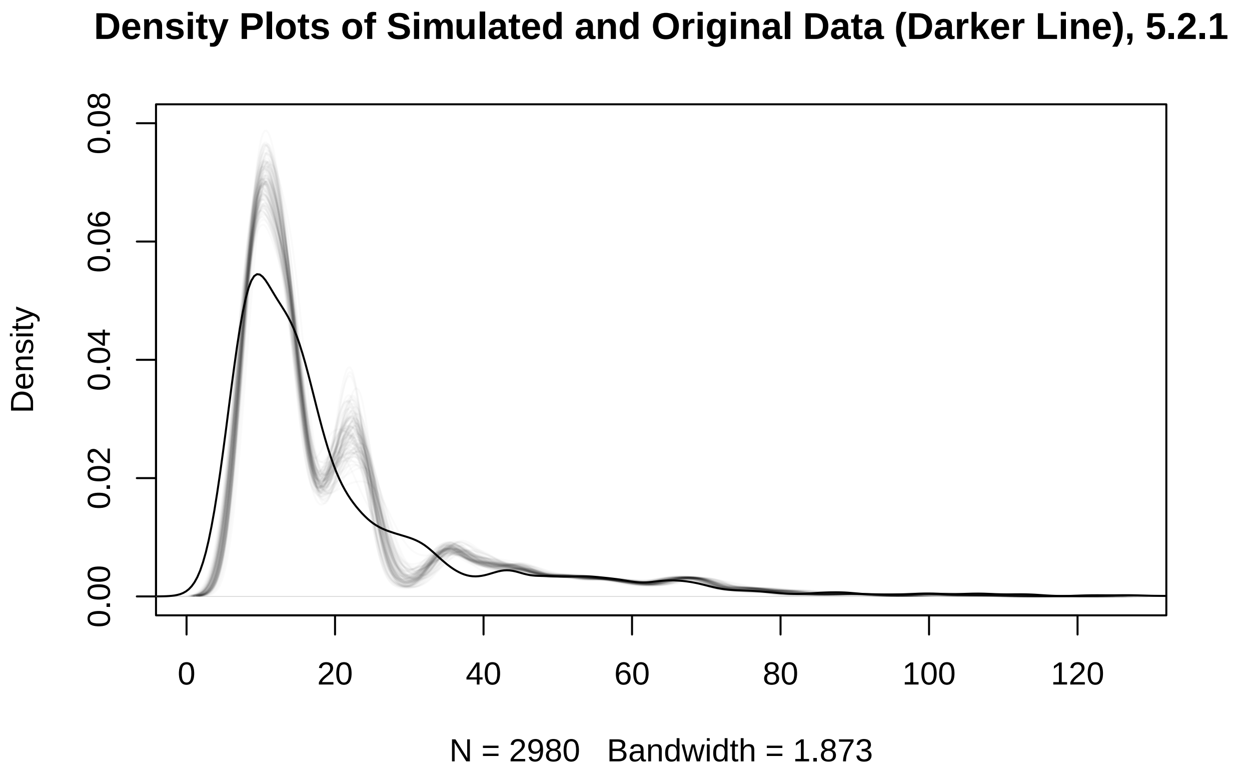
Question 5.2

```
#Question 5.2
#------------------------------------------------------------------------------
#Using Estimates from 4.2 Prior 1
#------------------------------------------------------------------------------

#Sample 100 draws from the posterior distribution
#Multiple Slopes/Means for Hierarchical Model
set.seed(465)
postdraws.ppool.521 <- inla.posterior.sample(100, partialpool, seed=465)
```

```
## Warning in inla.posterior.sample(100, partialpool, seed = 465): Since 'seed!=0',
## parallel model is disabled and serial model is selected, num.threads='1:1'
```

```
sim.data.521 <- data.frame(x = rep(GM$sat_2014, 100),
                           y = rep(NA, 100*2980),
                           group = rep(1:100, each = 2980))
```

```r
for(j in 1:100){
  y.521.sim <- postdraws.ppool.521[[j]]$latent[1:2980]
  sim.data.521$y[sim.data.521$group == j] <- y.521.sim
}

#Draw Density Curves
#for some reason ggplot doesn't want to separate the densities...
dat <- filter(sim.data.521, group==1)
plot(density(dat$y), col = alpha("black", 0.02),
     main="Density Plots of Simulated and Original Data (Darker Line), 5.2.1",
     ylim=c(0,0.08))
for(i in 2:100){
  dat <- filter(sim.data.521, group==i)
  lines(density(dat$y), col = alpha("black", 0.02))
}
#the original density
lines(density(GM$pm25))
```



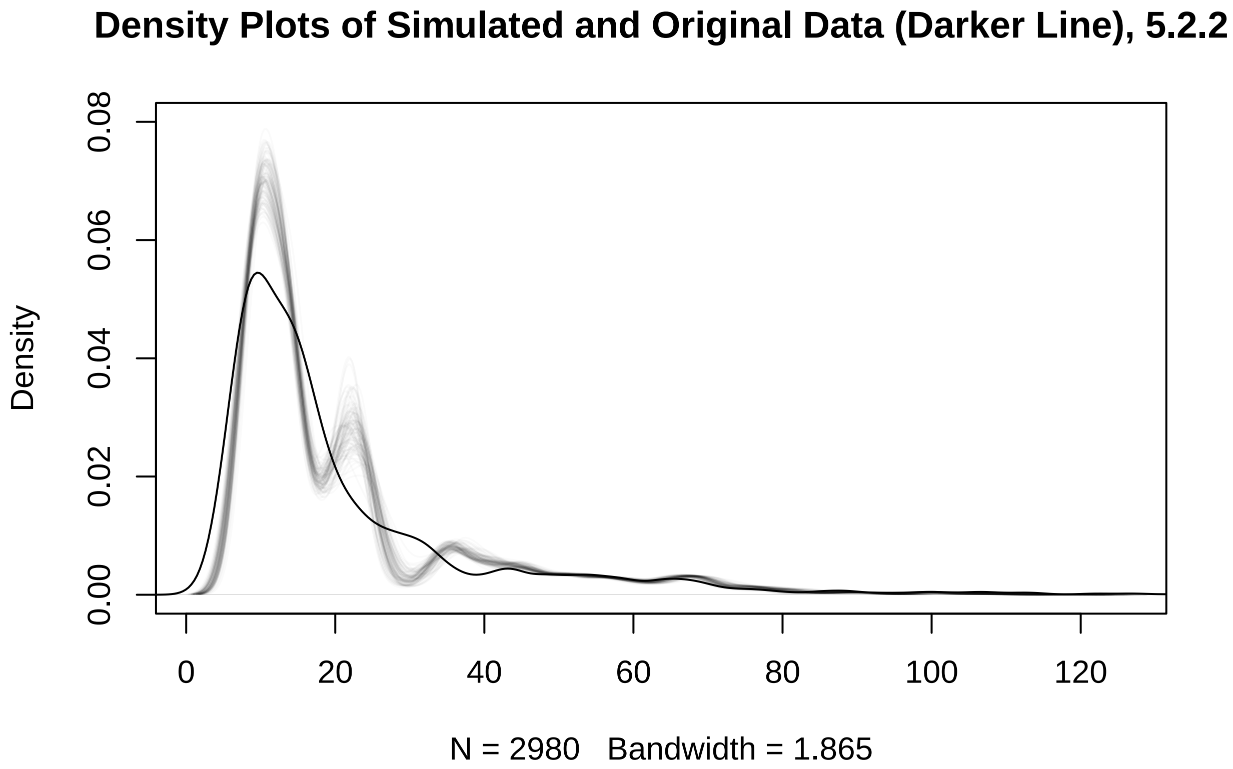
The simulated density plots align very close with the original data. The right-skewness is mostly the same for the simulated and original density plots, and the tails follow similar patterns. There seems to be a minor peak near approximately 22 for the simulated datasets, which is not present in the original data. The peak for the simulated densities are also higher than the original data.

```r
#-----------------------------------------------------------------------------
#Using Estimates from 4.2 Prior 2
#-----------------------------------------------------------------------------
```

```
#Sample 100 draws from the posterior distribution
#Multiple Slopes/Means for Hierarchical Model
set.seed(465)
postdraws.ppool.522 <- inla.posterior.sample(100, partialpool2, seed=465)
```

```
## Warning in inla.posterior.sample(100, partialpool2, seed = 465): Since 'seed!
## =0', parallel model is disabled and serial model is selected, num.threads='1:1'
```

```
sim.data.522 <- data.frame(x = rep(GM$sat_2014, 100),
                           y = rep(NA, 100*2980),
                           group = rep(1:100, each = 2980))

for(j in 1:100){
  y.522.sim <- postdraws.ppool.522[[j]]$latent[1:2980]
  sim.data.522$y[sim.data.522$group == j] <- y.522.sim
}

#Draw Density Curves
#for some reason ggplot doesn't want to separate the densities...
dat <- filter(sim.data.522, group==1)
plot(density(dat$y), col = alpha("black", 0.02),
     main="Density Plots of Simulated and Original Data (Darker Line), 5.2.2",
     ylim=c(0,0.08))
for(i in 2:100){
  dat <- filter(sim.data.522, group==i)
  lines(density(dat$y), col = alpha("black", 0.02))
}
#the original density
lines(density(GM$pm25))
```



The simulated density plots for the second prior distribution show similar results as the ones from the first prior distribution: The right-skewness is mostly the same for the simulated and original density plots, and the tails follow similar patterns. There seems to be a minor peak near approximately 22 for the simulated datasets, which is not present in the original data. The peak for the simulated densities are also higher than the original data.

References

- Lab Simulation Codes
- Textbook: Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A., & Rubin, D. (2013). Bayesian Data Analysis, Third edition. Chapman and Hall/CRC.