ps 4 template

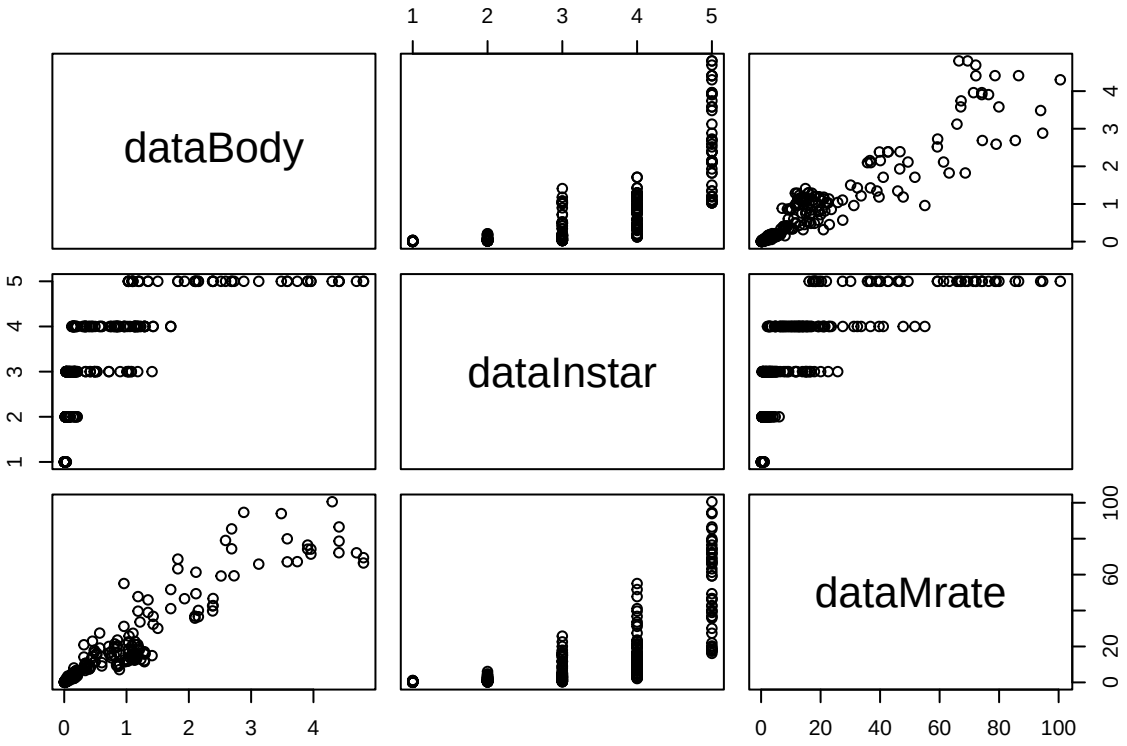
preliminaries

```
# clear work space
rm(list = ls())
data = read.csv("~/Desktop/MetabolicRate.csv")
dataBody = data$BodySize

dataInstar = data$Instar

dataMrate = data$Mrate
```

Question 1

Step 1

```
# make a scatter plot matrix to visualize corelations between different variables
dat <- data.frame(dataBody, dataInstar, dataMrate)
pairs(dat)
```



We note that all pairs of the three variables seem to have correlation. for example, there is clear, strong linear

correlation between Body Size and Metabolic rate (as seen in the upper right and lower left plots).

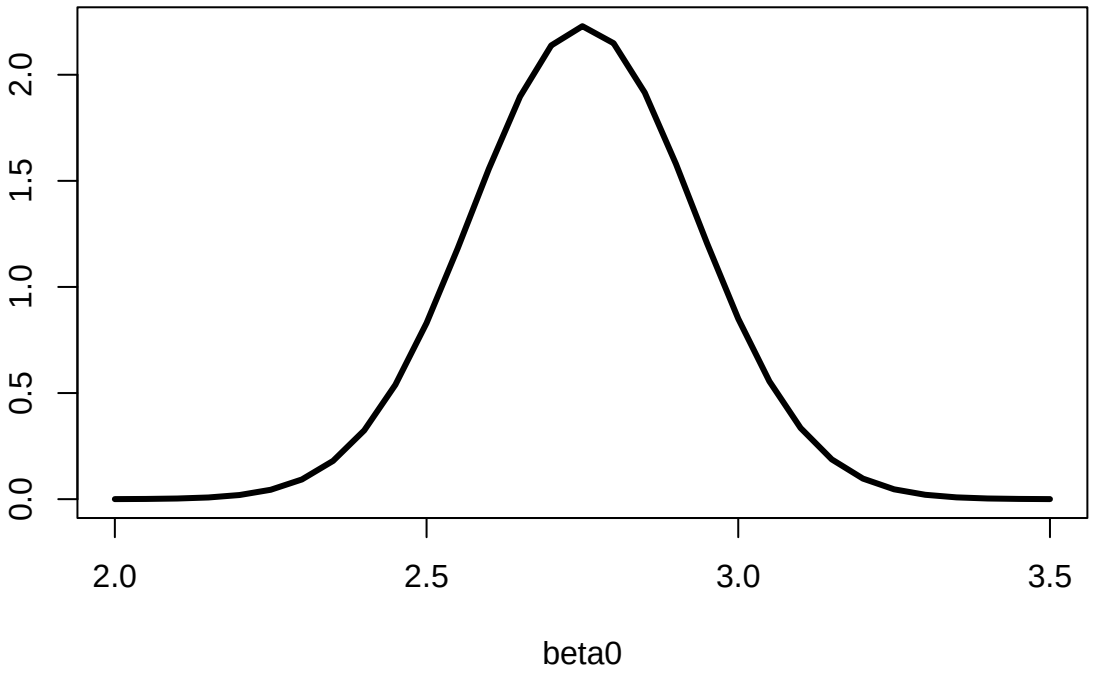
Step 2

Enter discussion here.

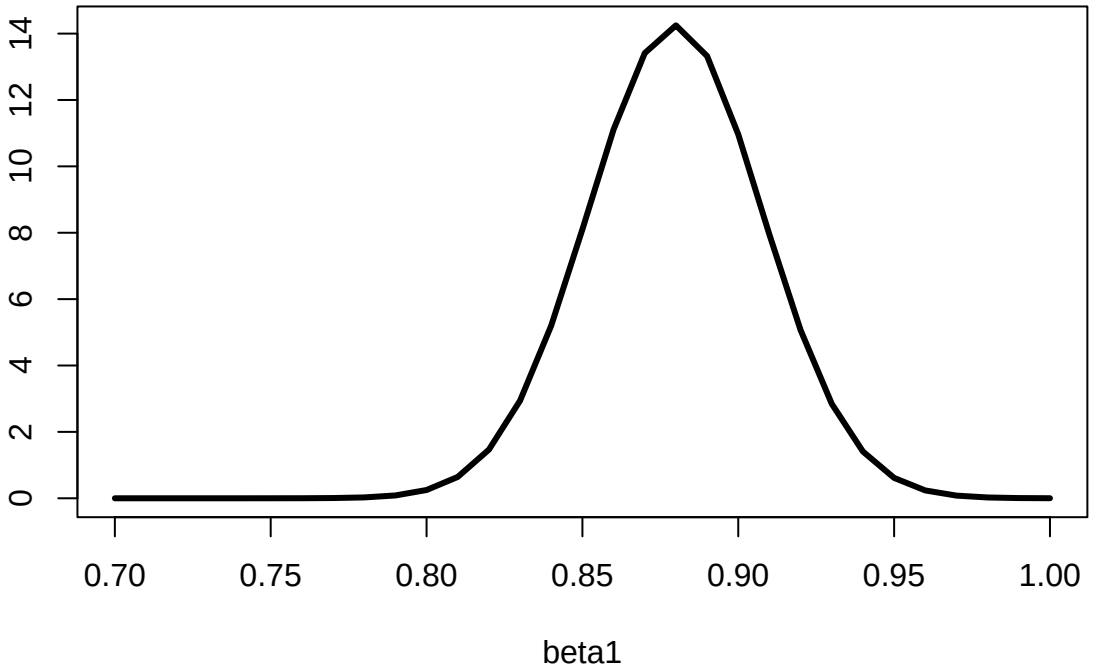
```
# define functions

# grid utility functions
stepSize = function(grid) {
  if (length(grid)==1) {
    step = 1
  }
  else {
    step = (max(grid) - min(grid)) / (length(grid) - 1)
  }
  return(step)
}

# build priors
buildPriorMultivar = function(beta0Grid,beta1Grid,beta2Grid,sigmaGrid) {
  # build useful grid objects
  nBeta0Grid = length(beta0Grid)
  nBeta1Grid = length(beta1Grid)
  nBeta2Grid = length(beta2Grid)
  nSigmaGrid = length(sigmaGrid)
  #
  prior = array( rep(1, nBeta0Grid * nBeta1Grid * nBeta2Grid * nSigmaGrid ),
                 dim = c(nBeta0Grid, nBeta1Grid, nBeta2Grid, nSigmaGrid ))
  #
  for (nB0 in 1:nBeta0Grid) {
    for (nB1 in 1:nBeta1Grid) {
      for (nB2 in 1:nBeta2Grid) {
        for (nSig in 1:nSigmaGrid) {
          # change next expression to set different priors
          prior[nB0,nB1,nB2, nSig] = 1 / nSig^2
        }
      }
    }
  }
  return(prior)
}

#likelihood
likelihoodMultivar = function(y,x1, x2, b0L, b1L, b2L, sL){
  loglike = sum(log(dnorm(y-b0L-b1L*x1-b2L*x2, mean = 0, sd=sL)))
  like = exp(loglike)
  return(like)
}

#compute posterior function
compPostMultivar = function(y,x1, x2, prior, beta0Grid,beta1Grid,beta2Grid,sigmaGrid) {
  # build useful grid objects
  nBeta0Grid = length(beta0Grid)
```

```
  nBeta1Grid = length(beta1Grid)
  nBeta2Grid = length(beta2Grid)
  nSigmaGrid = length(sigmaGrid)
  #initialize local posterior
  post = array( rep(-1, nBeta0Grid * nBeta1Grid * nBeta2Grid * nSigmaGrid ),
                dim = c(nBeta0Grid, nBeta1Grid, nBeta2Grid, nSigmaGrid ))
  # compute posterior
  for (nBeta0 in 1:nBeta0Grid) {
    b0 = beta0Grid[nBeta0]
    for (nBeta1 in 1:nBeta1Grid) {
      b1 = beta1Grid[nBeta1]
      for (nBeta2 in 1:nBeta2Grid) {
        b2 = beta2Grid[nBeta2]
        for (nSigma in 1:nSigmaGrid) {
          s = sigmaGrid[nSigma]
          post[nBeta0,nBeta1,nBeta2,nSigma] = likelihoodMultivar(y,x1,x2,b0,b1,b2,s) * prior[nBeta0,nBe
        }
      }
    }
  }
  # normalize posterior
  post = post / (sum(post) *stepSize(beta0Grid) * stepSize(beta1Grid) * stepSize(beta2Grid) * stepSize(s
  # return
  return(post)
}

# construct grids
beta0Grid = seq(2, 3.5, by = 0.05)
beta1Grid = seq(0.7, 1, by = 0.01)
beta2Grid = seq(-0.15, 0.25, by = 0.01)
sigmaGrid = seq(0.2, 0.6, by = 0.05)

# build prior
prior = buildPriorMultivar(beta0Grid,beta1Grid,beta2Grid,sigmaGrid)

post = compPostMultivar(log(dataMrate), log(dataBody), dataInstar,
                            prior,beta0Grid,beta1Grid,
                            beta2Grid,sigmaGrid)

#compute marginal posteriors
margPostBeta0 = apply(post,c(1),sum)
margPostBeta0 = margPostBeta0 / (sum(margPostBeta0) * stepSize(beta0Grid))

margPostBeta1 = apply(post,c(2),sum)
margPostBeta1 = margPostBeta1 / (sum(margPostBeta1) * stepSize(beta1Grid))

margPostBeta2 = apply(post,c(3),sum)
margPostBeta2 = margPostBeta2 / (sum(margPostBeta2) * stepSize(beta2Grid))

margPostSigma = apply(post,c(4),sum)
margPostSigma = margPostSigma / (sum(margPostSigma) * stepSize(sigmaGrid))

#plot marginal posteriors
```

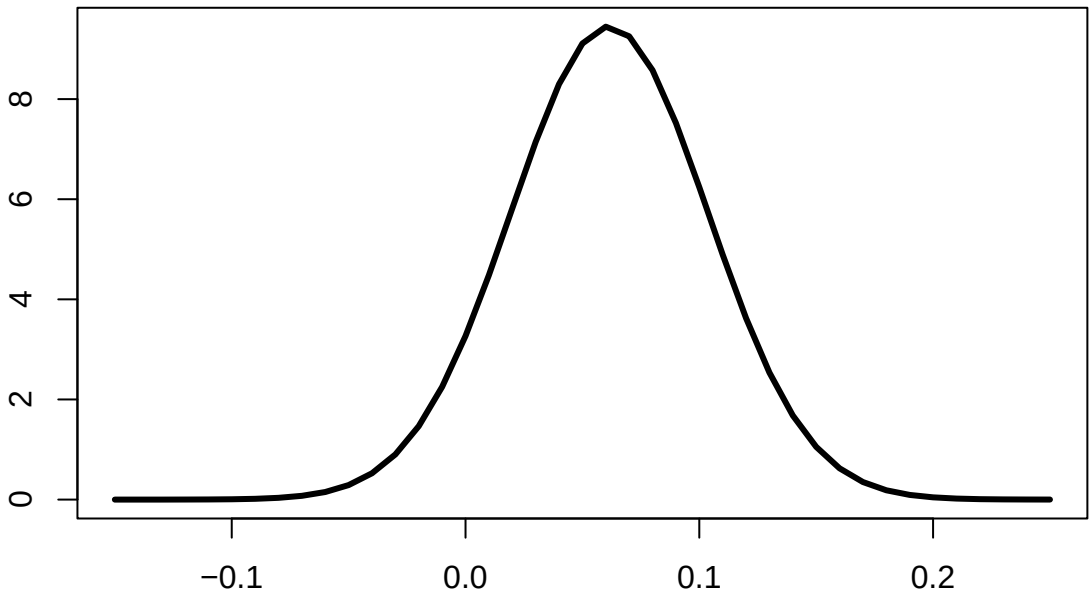
```
plot(beta0Grid, margPostBeta0, xlab = "beta0", ylab="", type = "l", lwd = 3)
```



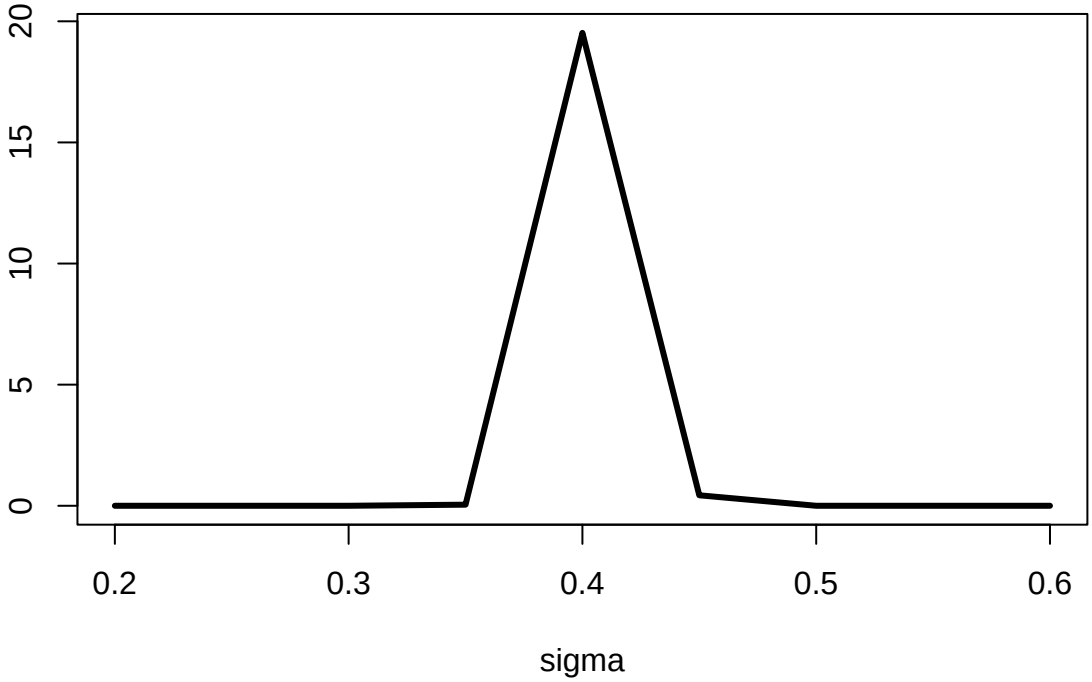

```
plot(beta1Grid, margPostBeta1, xlab = "beta1", ylab="", type = "l", lwd = 3)
```




```
plot(beta2Grid, margPostBeta2, xlab = "beta2", ylab="", type = "l", lwd = 3)
```




```
plot(sigmaGrid, margPostSigma, xlab = "sigma", ylab="", type = "l", lwd = 3)
```

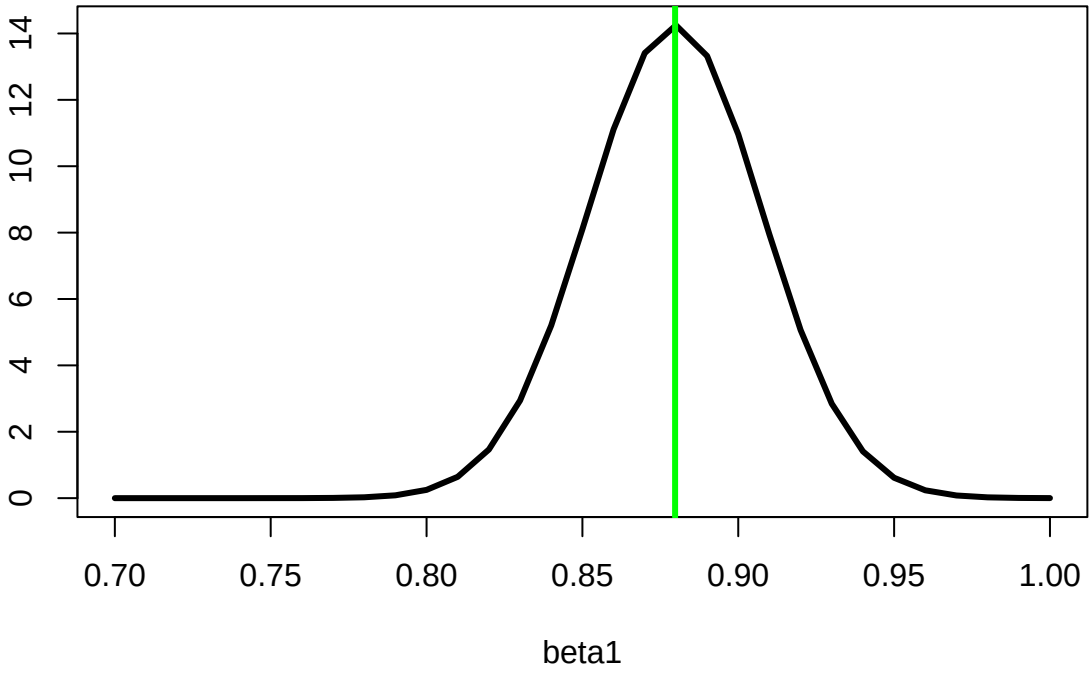


Final grid sizes: beta0: {2, 2.05, . . . , 2.95, 3} beta1: {0.7, 0.71, . . . , 0.99, 1} beta2: {-0.15, -0.16, . . . , 0.24, 0.25} sigma: {0.2, 0.25, . . . , 0.55, 0.6}

Step 3

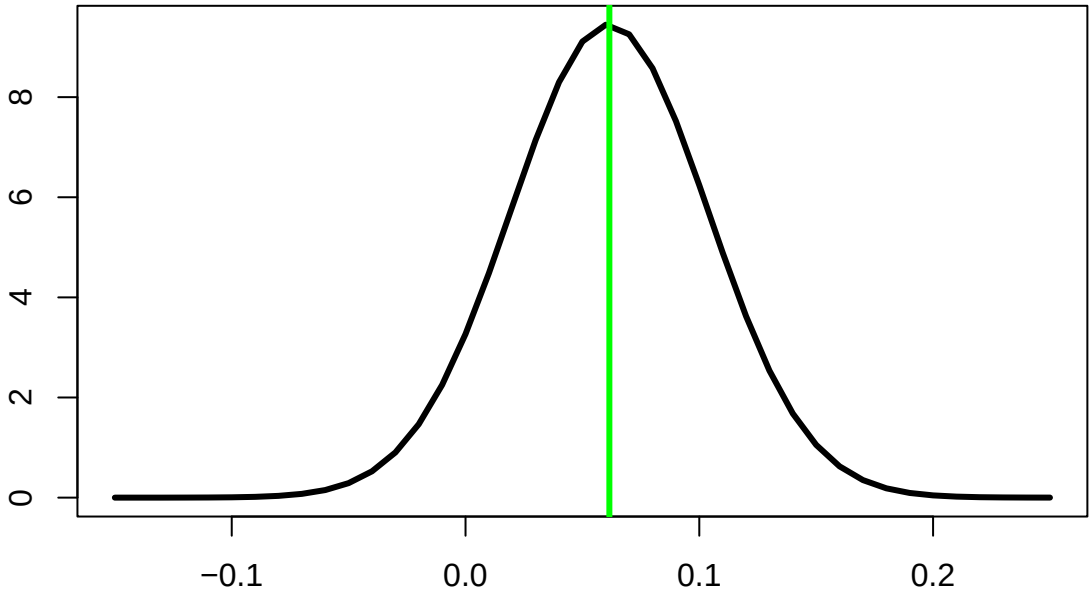
```
#plot marginal posteriors for beta1 and beta2 with means
mu1 = 0
for (i in 1:length(margPostBeta1)){
```

```
  mu1 = mu1 + margPostBeta1[i] * beta1Grid[i] * stepSize(beta1Grid)
}
mu2 = 0
for (i in 1:length(margPostBeta2)){
  mu2 = mu2 + margPostBeta2[i] * beta2Grid[i] * stepSize(beta2Grid)
}

plot(beta1Grid, margPostBeta1, xlab = "beta1", ylab="", type = "l", lwd = 3)
abline(v=mu1, lwd=3, col="green")
```

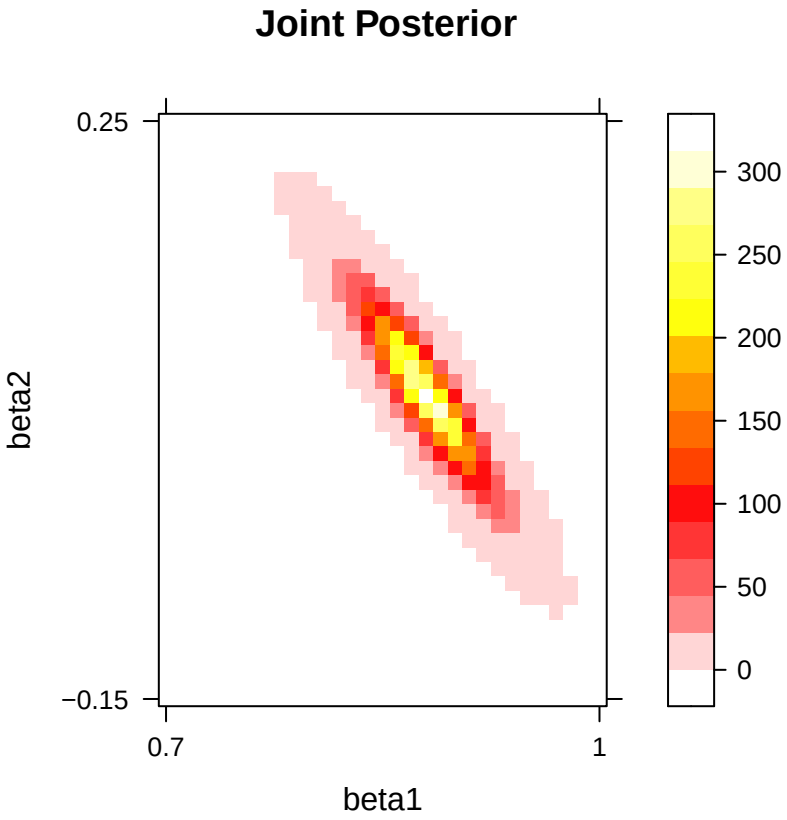



```
plot(beta2Grid, margPostBeta2, xlab = "beta2", ylab="", type = "l", lwd = 3)
abline(v=mu2, lwd=3, col="green")

#compute beta1, beta2 joint posterior
nBeta0Grid = length(beta0Grid)
nBeta1Grid = length(beta1Grid)
nBeta2Grid = length(beta2Grid)
nSigmaGrid = length(sigmaGrid)
b1b2JointPost = array(rep(1, nBeta1Grid * nBeta2Grid),
                      dim = c(nBeta1Grid, nBeta2Grid))

for (i in 1:nBeta1Grid){
  for (j in 1:nBeta2Grid){
    valk = 0
    for (k in 1:nBeta0Grid){
      vall = 0
      for (l in 1:nSigmaGrid){
        vall = vall + stepSize(sigmaGrid) * post[k, i, j, l]
      }
      valk = valk + vall * stepSize(beta0Grid)
    }
    b1b2JointPost[i,j] = valk
```

```
  }
}

# visualize joint posterior
library(lattice)
```




```
new.palette=colorRampPalette(c("white","red","yellow","white"),space="rgb")
levelplot(b1b2JointPost, col.regions=new.palette(20),
          xlab = "beta1", ylab = "beta2",
          main = "Joint Posterior",
          scales=list(x=list(at=c(1,nBeta1Grid), labels=c(min(beta1Grid),max(beta1Grid))),
                      y=list(at=c(1,nBeta2Grid), labels=c(min(beta2Grid),max(beta2Grid)))))
```



Step 4

Recalling the result of part 1, we know there is correlation between the log(bodysize) and instar predictors. Therefore, we would expect that our regression will give us a joint posterior for beta1 and beta 2 that is elongated and elliptical–and therefore indicitave of correlation between these two predictors.

Step 5

```
# compute probability that b1 and b2 are >0
probi = 0
for (i in 1:nBeta1Grid){
  probj = 0
  for (j in 1:nBeta2Grid){
    if (beta2Grid[j] > 0){
      probj = probj + stepSize(beta2Grid) * b1b2JointPost[i, j]
    }
  }
  probi = probi + probj * stepSize(beta1Grid)
}
totProb = probi
print(totProb)
```

```
## [1] 0.9100993
```

Thus, there's a ~91% chance that both beta1 and beta2 will be > 0.

Question 2

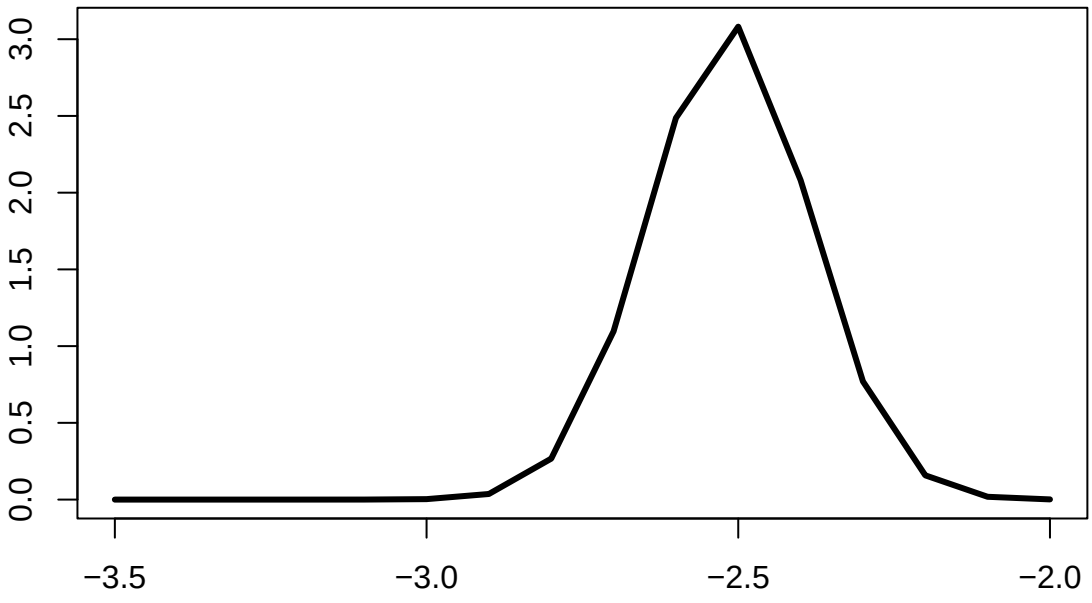
Step 1

Given our findings in Problem 1, we'd expect that the uni-variate model will give a marginal posterior distribution for beta1 that is shifted away from its true distribution because of the correlation between the x1 and x2 predictors. Furthermore, we'd expect that the shift is upwards (or larger).

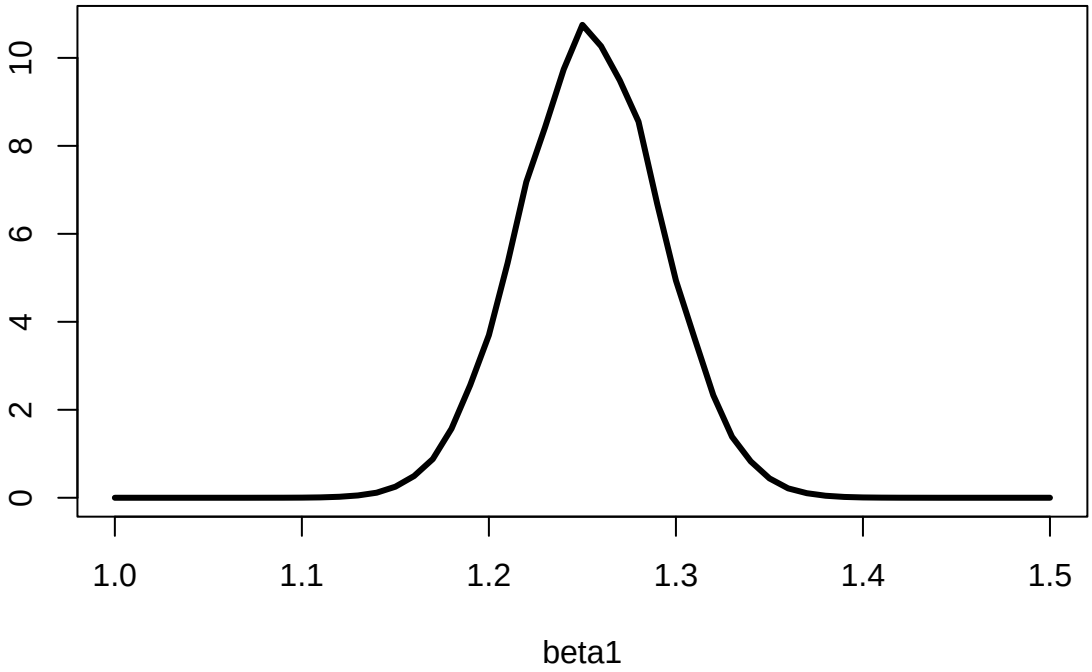
Step 2

```
# define functions

buildPriorUnivar = function(beta0Grid,betaPGrid,sigmaGrid) {
  # build useful grid objects
  nBeta0Grid = length(beta0Grid)
  nBetaPGrid = length(betaPGrid)
  nSigmaGrid = length(sigmaGrid)
  #
  prior = array( rep(1, nBeta0Grid * nBetaPGrid * nSigmaGrid ),
                 dim = c(nBeta0Grid, nBetaPGrid, nSigmaGrid ))
  #
  for (nB0 in 1:nBeta0Grid) {
    for (nBP in 1:nBetaPGrid) {
      for (nSig in 1:nSigmaGrid) {
        # change next expression to set different priors
        prior[nB0,nBP, nSig] = 1 / nSig^2
      }
    }
  }
  return(prior)
}

likelihoodUnivar = function(y,xP, b0L, bPL, sL){
  loglike = sum(log(dnorm(y-b0L-bPL*xP, mean = 0, sd=sL)))
  like = exp(loglike)
  return(like)
}

compPostUnivar = function(y,xP, prior, beta0Grid,betaPGrid,sigmaGrid) {
  # build useful grid objects
  nBeta0Grid = length(beta0Grid)
  nBetaPGrid = length(betaPGrid)
  nSigmaGrid = length(sigmaGrid)
  #initialize local posterior
  post = array( rep(-1, nBeta0Grid * nBetaPGrid * nSigmaGrid ),
                dim = c(nBeta0Grid, nBetaPGrid, nSigmaGrid ))
  # compute posterior
  for (nBeta0 in 1:nBeta0Grid) {
    b0 = beta0Grid[nBeta0]
    for (nBetaP in 1:nBetaPGrid) {
      bP = betaPGrid[nBetaP]
      for (nSigma in 1:nSigmaGrid) {
```

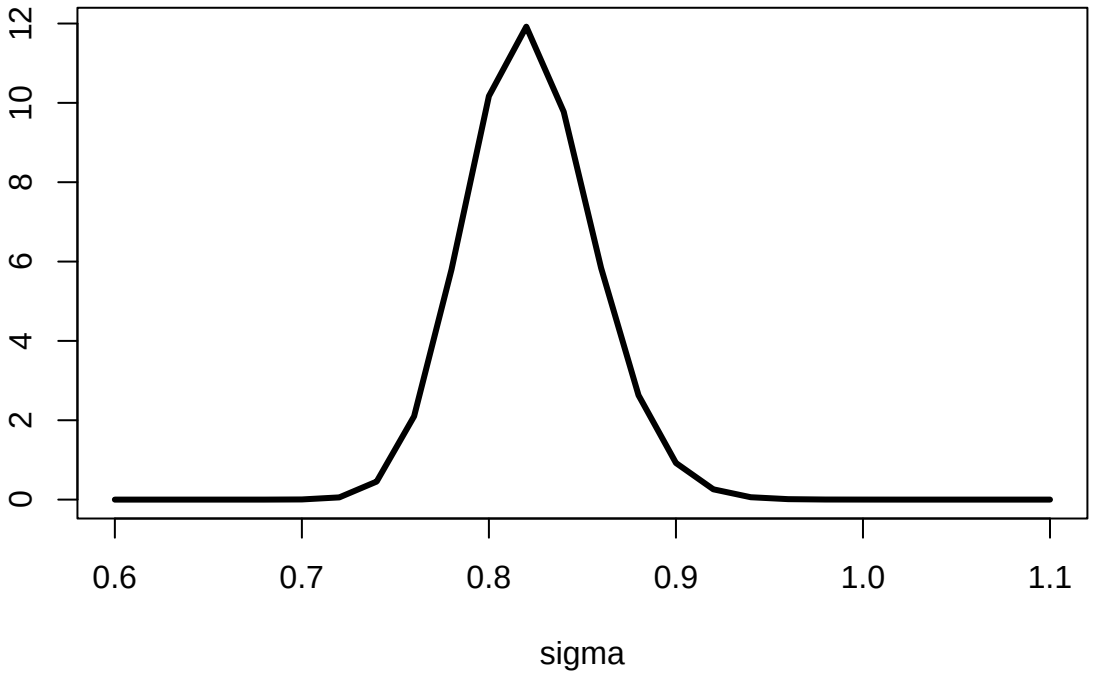
```
        s = sigmaGrid[nSigma]
        post[nBeta0,nBetaP,nSigma] = likelihoodUnivar(y,xP,b0,bP,s) * prior[nBeta0,nBetaP,nSigma]
      }
    }
  }
  # normalize posterior
  post = post / (sum(post) *stepSize(beta0Grid) * stepSize(betaPGrid) * stepSize(sigmaGrid))
  # return
  return(post)
}

# construct grids
beta0GridM1 = seq(-3.5, -2, by = 0.1)
beta1GridM1 = seq(1, 1.5, by = 0.01)
beta2GridM1 = c(0)
sigmaGridM1 = seq(0.6, 1.1, by = 0.02)

# build priors
priorM1 = buildPriorUnivar(beta0GridM1,beta1GridM1,sigmaGridM1)

# build posterior
postM1 = compPostUnivar(log(dataMrate), dataInstar, priorM1, beta0GridM1,
                        beta1GridM1,sigmaGridM1)

#compute marginal posteriors
margPostBeta0M1 = apply(postM1,c(1),sum)
margPostBeta0M1 = margPostBeta0M1 / (sum(margPostBeta0M1) * stepSize(beta0GridM1))

margPostBeta1M1 = apply(postM1,c(2),sum)
margPostBeta1M1 = margPostBeta1M1 / (sum(margPostBeta1M1) * stepSize(beta1GridM1))

margPostSigmaM1 = apply(postM1,c(3),sum)
margPostSigmaM1 = margPostSigmaM1 / (sum(margPostSigmaM1) * stepSize(sigmaGridM1))

#plot marginal posteriors
plot(beta0GridM1, margPostBeta0M1, xlab = "beta0", ylab="", type = "l", lwd = 3)
```




```
plot(beta1GridM1, margPostBeta1M1, xlab = "beta1", ylab="", type = "l", lwd = 3)
```

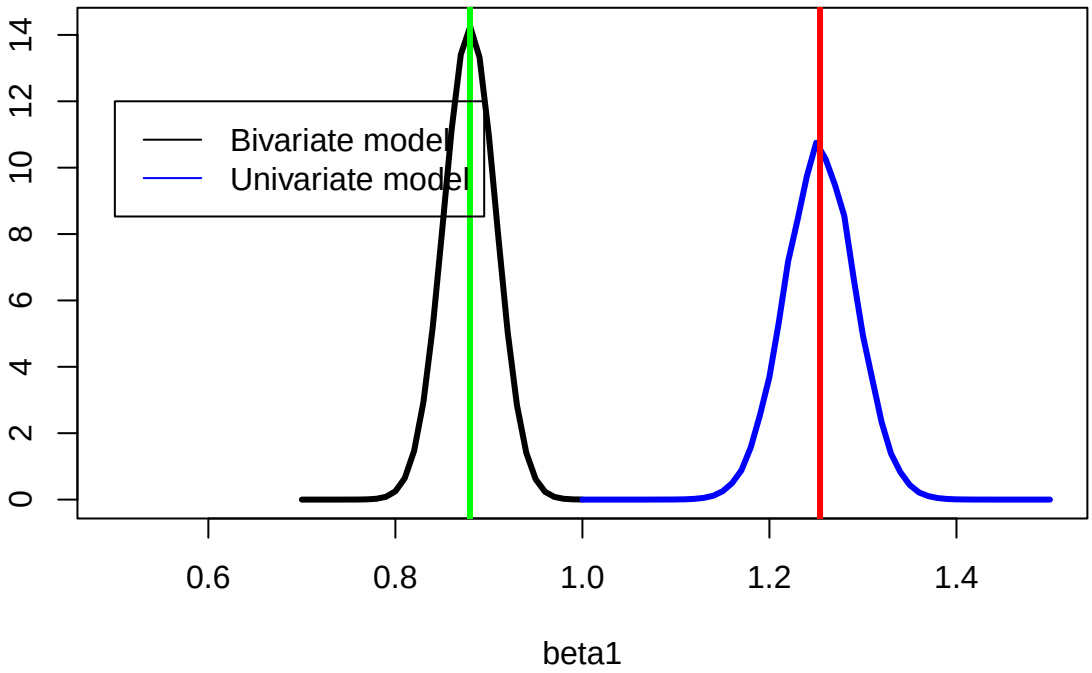



```
plot(sigmaGridM1, margPostSigmaM1, xlab = "sigma", ylab="", type = "l", lwd = 3)
```



Step 3

```
# compare marginal posterior distributions for beta1 in the univariate and bivariate model
muM1 = 0
for (i in 1:length(margPostBeta1M1)){
  muM1 = muM1 + margPostBeta1M1[i] * beta1GridM1[i] * stepSize(beta1GridM1)
}

plot(beta1Grid, margPostBeta1, xlab = "beta1", ylab="", type = "l", lwd = 3,
     col = "black", xlim = c(0.5, 1.5))
points(beta1GridM1, margPostBeta1M1,
     type = "l", lwd = 3, col = "blue")
abline(v=mu1, lwd=3, col="green")
abline(v=muM1, lwd=3, col="red")
legend(0.5, 12, legend=c("Bivariate model", "Univariate model"),
       col=c("black", "blue"), lty = c(1, 1))
```

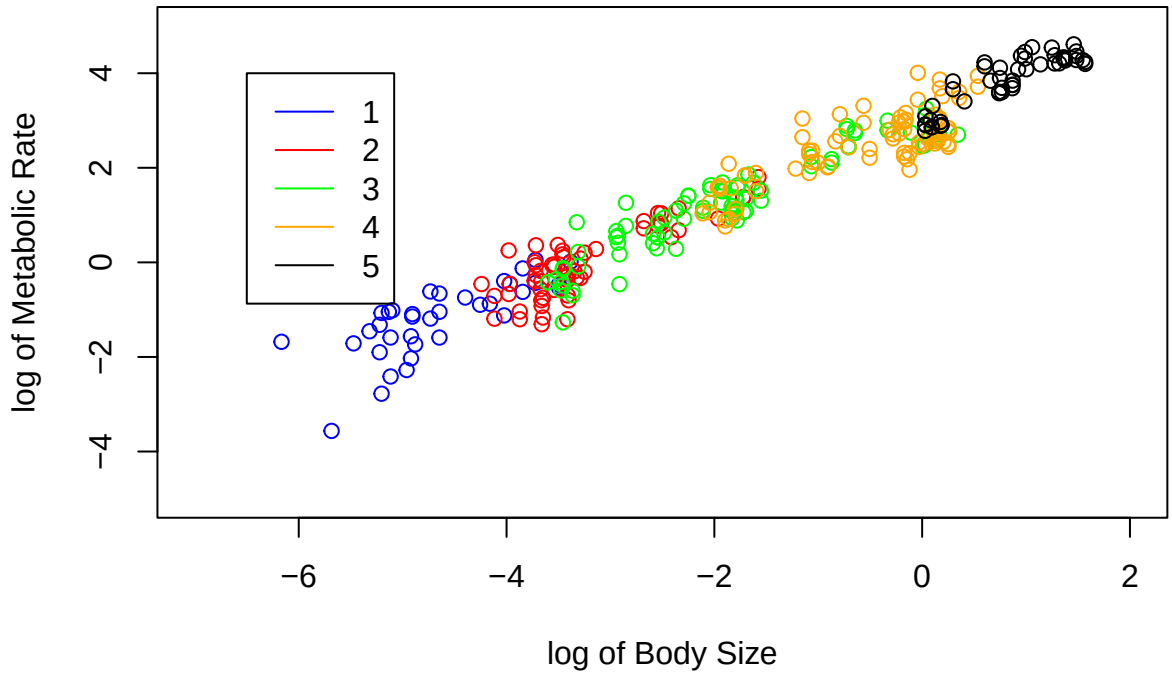


These marginal posterior distributions are quite different because the uni-variate one is biased. It's biased because we omitted a predictor (x2=body size) which is correlated with predictor x1=Instar.

Step 4

```
## sort the data by instar value
dataBody1 = c()
dataMrate1 = c()

dataBody2 = c()
dataMrate2 = c()

dataBody3 = c()
dataMrate3 = c()

dataBody4 = c()
dataMrate4 = c()

dataBody5 = c()
dataMrate5 = c()

for (i in 1:305){
  if (dataInstar[i] == 1){
    dataBody1 = append(dataBody1, c(dataBody[i]))
    dataMrate1 = append(dataMrate1, c(dataMrate[i]))
  }
  if (dataInstar[i] == 2){
    dataBody2 = append(dataBody2, c(dataBody[i]))
    dataMrate2 = append(dataMrate2, c(dataMrate[i]))
  }
  if (dataInstar[i] == 3){
```

```
    dataBody3 = append(dataBody3, c(dataBody[i]))
    dataMrate3 = append(dataMrate3, c(dataMrate[i]))
  }
  if (dataInstar[i] == 4){
    dataBody4 = append(dataBody4, c(dataBody[i]))
    dataMrate4 = append(dataMrate4, c(dataMrate[i]))
  }
  if (dataInstar[i] == 5){
    dataBody5 = append(dataBody5, c(dataBody[i]))
    dataMrate5 = append(dataMrate5, c(dataMrate[i]))
  }
}
plot(log(dataBody1), log(dataMrate1), xlab = "log of Body Size",
     ylab="log of Metabolic Rate",type = "p", col="blue", xlim=c(-7, 2), ylim=c(-5, 5))
points(log(dataBody2), log(dataMrate2), col="red")
points(log(dataBody3), log(dataMrate3), col="green")
points(log(dataBody4), log(dataMrate4), col="orange")
points(log(dataBody5), log(dataMrate5), col="black")
legend(-6.5, 4, legend=c("1", "2", "3", "4", "5"),
       col=c("blue", "red", "green", "orange", "black"), lty = c(1, 1))
```



This plot clearly shows that there is a linear correlation between log(body size) and log(metabolic rate). Furthermore, we note that the Instar value is a spurious predictor (just like the latin last names in the lecture example). It correlates with log of metabolic rate, but it's not sufficient to fully model the system–we also need the body size.