Math 3606: Bayesian Statistics
Problem Set #5

Max Thrush Hukill

End of October/Start of November, 2019

Collaborators: Juliana Taube, Sam Harder, Connor Fitch

Problem V.1. Behold: The Gibbs Sampler: For the data data.txt on Piazza, assume that the data comes from the following model:

$$\mu \sim \mathcal{N}\left(\eta, \frac{1}{\xi}\right)$$
$$\tau \sim GAMMA(\alpha, \beta)$$
$$x_1, \ldots, x_N \overset{iid}{\sim} \mathcal{N}\left(\mu, \frac{1}{\tau}\right)$$

Now, let's do some stuff.

```
data <- c(3.63013576853802,2.51023630417296,4.08246288055139,
1.81723363937268,3.42951046261361)
# Make the Mesh for Posterior
res <- 1/100 # resolution
mus <- seq(0,6, by = res) # note that data never goes outside this range
taus <- seq(0, 5, by = res) # these are used for the mesh

post.matrix <- matrix(0,nrow=length(mus), ncol=length(taus)) # make mesh matrix
for(m in 1:length(mus)){
  for(n in 1:length(taus)){
    llk <- prod(dnorm(data, mus[m],sd=1/sqrt(taus[n])))
    post.matrix[m,n] <- llk * dnorm(mus[m],mean=0, sd=10) * (dgamma(taus[n],1,1)) #llk*prior
  }
}
post.matrix <- post.matrix/sum(post.matrix) # bivariate posterior distribution

# Update Functions
update.tau <- function(mu){ # update tau, given fixed alpha, beta
  alpha.new <- alpha.o + length(data)/2
  beta.new <- beta.o + sum((data-mu)^2)/2
  return(c(alpha.new,beta.new))
}

update.mu <- function(tau){ # update mu, given fixed xi, eta
  xi.new = xi.o + length(data)*tau
  eta.new = (xi.o*eta.o + tau*sum(data))/xi.new
  return(c(xi.new, eta.new))
}

# Initialize Values for Algorithm
T = 10000
alpha.o = 1 # prior alpha
beta.o = 1 # prior beta
eta.o = mean(data) # prior eta
xi.o = 1/var(data) # prior xi
tau.mu.mat <- matrix(0, ncol=2, nrow=T) # storage matrix for tau and mu values
tau.mu.mat[1,1] = 1/var(data) # first tau
tau.mu.mat[1,2] = mean(data) # first mu

# Run Algorithm
for(t in 2:T)
{
  if(t %% 2 == 1) # t is odd
  {
    tau.mu.mat[t,1] <- tau.mu.mat[t-1,1] # fix tau
    xi.new = update.mu(tau.mu.mat[t,1])[1]  # find new xi
    eta.new = update.mu(tau.mu.mat[t,1])[2]  # find new eta
    tau.mu.mat[t,2] <- rnorm(1,eta.new,1/sqrt(xi.new)) # find new mu
  }
  else # t is even
  {
    tau.mu.mat[t,2] <- tau.mu.mat[t-1,2] # fix mu
    alpha.new = update.tau(tau.mu.mat[t,2])[1] # find new xi
    beta.new = update.tau(tau.mu.mat[t,2])[2] # find new beta
    tau.mu.mat[t,1] <- rgamma(1,alpha.new, beta.new) # find new tau
  }
}

image(post.matrix, ylab=expression(tau), col=topo.colors(12, alpha=1), xlab=expression(mu),
axes=FALSE) axis(1,at=seq(0,1, length.out = 9), labels=round(seq(0,6,length.out = 9),2))
axis(2,at=seq(0,1, length.out = 9), labels=round(seq(0,5,length.out = 9),2))
points(tau.mu.mat[,2]/6, tau.mu.mat[,1]/5, col=rgb(0,0,0,0.3)) # adding algorithm points

prob_omega = sum(post.matrix[which(mus>3),which(taus>2.5)]) # omega region probability
fraction_omega_points = sum(tau.mu.mat[,1]>2.5 & tau.mu.mat[,2]>3)/T # points in omega region
```

```
# Correlation Station
cor.vec = rep(0,20)
for(k in 1:20) # going through each k value
{
  group.1 = tau.mu.mat[1:(T-k),2] # first mu series to compare
  group.2 = tau.mu.mat[(k+1):T,2] # second
  cor.vec[k] = cor(group.1, group.2) # find correlation
}
plot(seq(1,20,by=1), cor.vec, ylab='Correlation', xlab='k values')
```

One can construct “independent” samples by looking at two μ values that came from the same distribution: ie the same τ value (or $(\mu|\tau_i)$). This yields $\boxed{T/2}$ distinct independent μ samples (because each τ generates two μ realizations).

```
subseq.A = rep(0,0)
subseq.B = rep(0,0)

for(i in 2:(T-1)){
  if(i %% 2 == 1)
  {
    subseq.A = c(subseq.A,tau.mu.mat[i,2])
  }
  else
  {
    subseq.B = c(subseq.B,tau.mu.mat[i,2])
  }
}
independent.cor = cor(subseq.A,subseq.B)
print(independent.cor)
```

Note, the correlation changes upon each sourcing, but a typical value is on the order of magnitude of 10^{-3} – better known by the engineers, physicists, and statisticians as zero. Thus, the samples are “independent.”

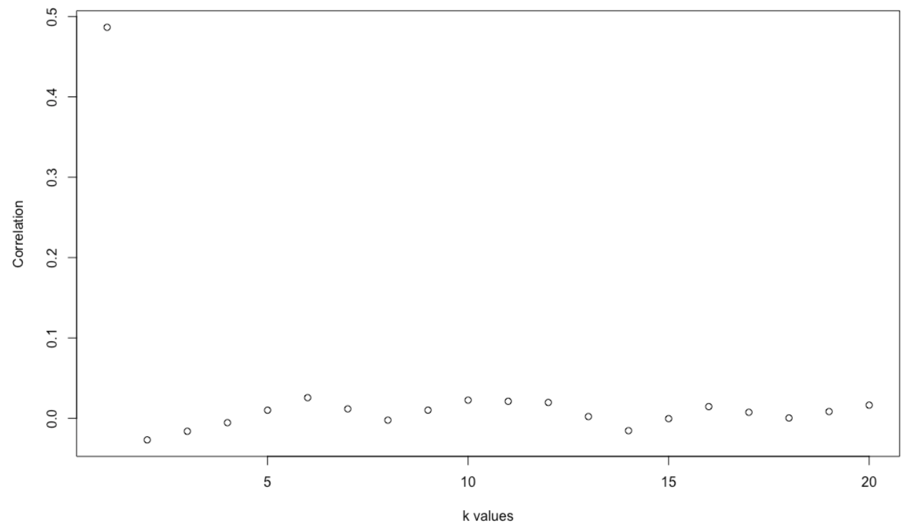


Figure 1: Autocorrelation of Samples

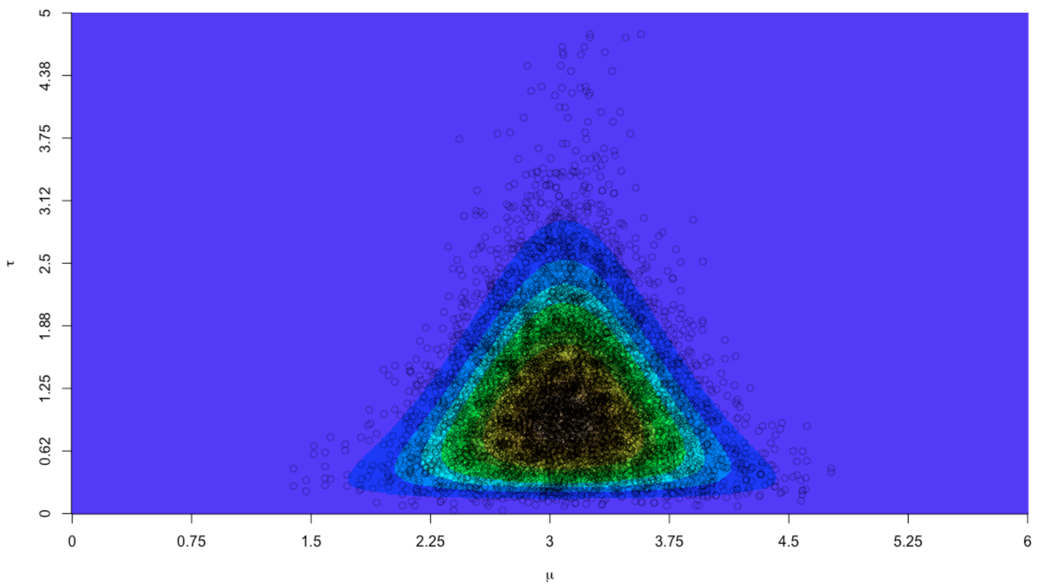


Figure 2: Gibbs Sampler Heat Map Over the Posterior Mesh

Problem V.2. Linear Regression with a Gibbs Sampler. As given by the data set on Piazza, we'll be analyzing house prices as a function of age (A_i) and square footage (S_i) through linear regression. Suppose our model is defined thus:

$$\begin{aligned}
\beta_i &\overset{iid}{\sim} \mathcal{N}(\eta, \Omega), \quad \text{with} \quad \eta = 0 \\
\tau &\sim GAMMA(a, b) \\
y_i &\overset{iid}{\sim} \mathcal{N}\left(\beta_0 + \beta_1 \cdot S_i + \beta_2 \cdot A_i, \frac{1}{\tau}\right).
\end{aligned}$$

We can consider a directed acyclic graph (DAG) representing the model:

$$\begin{tikzcd}
 & & & a \arrow[d] & b \arrow[dl] \\
 & \beta_2 \arrow[dr, dotted] & & \tau \arrow[dd] & \\
\eta, \Omega \arrow[ur] \arrow[r] \arrow[dr] & \beta_1 \arrow[r, dotted] & \vec{\beta} \arrow[dr] & & \\
 & \beta_0 \arrow[ur, dotted] & & y_i & \\
 & & & S_i \arrow[u] & A_i \arrow[ul]
\end{tikzcd}$$

Figure 3: DAG of Regression Model

Let's begin by finding the conjugate update for each β_i. Consider the general likelihood given the β_i values:

$$\mathscr{L}(\mathbf{y}, \mathbf{S}, \mathbf{A} | \beta_0, \beta_1, \beta_2, \tau) = \prod_{i=1}^{N} \mathcal{N}\left(y_i - \beta_0 - \beta_1 S_i - \beta_2 A_i \middle| 0, \frac{1}{\tau}\right).$$

By clever rearrangement of this likelihood expression, we can derive the β_i posteriors. To put this in practice, let's consider β_0. First, let's grab the likelihood expression:

$$\mathscr{L} = (\beta_0 | \mathbf{y}, \mathbf{S}, \mathbf{A}, \beta_1, \beta_2, \tau) = \mathcal{N}\left(\mathbf{y} - \beta_0 - \beta_1 \mathbf{S} - \beta_2 \mathbf{A} \middle| 0, \frac{1}{N\tau}\right).$$

Now, we can consider the posterior:

$$\begin{aligned}
\pi(\beta_0 | \mathbf{y}, \mathbf{S}, \mathbf{A}, \beta_1, \beta_2, \tau) &\propto \mathcal{N}\left(\mathbf{y} - \beta_1 \mathbf{S} - \beta_2 \mathbf{A}, \frac{1}{N\tau}\right) \cdot \mathcal{N}(\beta_0, \Omega) \\
\pi(\beta_0 | \mathbf{y}, \mathbf{S}, \mathbf{A}, \beta_1, \beta_2, \tau) &= \mathcal{N}\left(\beta_0 \middle| \frac{N\tau\Omega(\mathbf{y} - \beta_1 \mathbf{S} - \beta_2 \mathbf{A})}{N\tau\Omega + 1}, \frac{\Omega}{N\tau\Omega + 1}\right),
\end{aligned}$$

following from the normal-normal conjugate update rule. Next, consider β_1. The likelihood expression is vastly more complex than before:

$$\begin{aligned}
\mathscr{L}\left(\mathbf{y}, \mathbf{S}, \mathbf{A} \middle| \beta_0, \beta_1, \beta_2, \tau\right) &= \prod_{i=1}^{N} \mathcal{N}\left(y_i - \beta_0 - \beta_1 S_i - \beta_2 A_i \middle| 0, \frac{1}{\tau}\right) \\
\log \mathscr{L} &= \frac{\tau}{2} \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 S_i - \beta_2 A_i)^2 \\
&\propto \frac{\tau}{2}\left(\beta_1^2 \sum S_i^2 - 2\beta_1 \sum \left[y_i S_i - \beta_0 S_i - \beta_2 A_i S_i\right] + C\right) \\
&\propto \frac{\tau}{2}\left(\sum S_i^2 \left(\beta_1 - \frac{\sum y_i S_i - \beta_0 S_i - \beta_2 A_i S_i}{\sum S_i^2}\right)^2\right) \\
&\propto \mathcal{N}\left(\beta_1 \middle| \frac{\sum y_i S_i - \beta_0 S_i - \beta_2 A_i S_i}{\sum S_i^2}, \frac{1}{\tau \sum S_i^2}\right).
\end{aligned}$$

Next, we can create the posterior:

$$\begin{aligned}
\pi(\beta_1 | \mathbf{y}, \mathbf{S}, \mathbf{A}, \beta_0, \beta_2, \tau) &\propto \mathcal{N}\left(\beta_1 \middle| \frac{\sum y_i S_i - \beta_0 S_i - \beta_2 A_i S_i}{\sum S_i^2}, \frac{1}{\tau \sum S_i^2}\right) \cdot \mathcal{N}\left(\beta_1 \middle| \eta, \Omega\right) \\
\pi(\beta_1 | \mathbf{y}, \mathbf{S}, \mathbf{A}, \beta_0, \beta_2, \tau) &= \mathcal{N}\left(\beta_1 \middle| \frac{\Omega\tau \sum y_i S_i - \beta_0 S_i - \beta_2 A_i S_i}{1 + \Omega\tau \sum S_i^2}, \frac{\Omega}{1 + \Omega\tau \sum S_i^2}\right).
\end{aligned}$$

Note that β_1 and β_2 are of the same form in the regression equation. Thus, we can employ the above strategy for the posterior of β_2; we need only substitute A_i for S_i, and β_2 for β_1. Hence,

$$\pi(\beta_2 | \mathbf{y}, \mathbf{S}, \mathbf{A}, \beta_0, \beta_1, \tau) = \mathcal{N}\left(\beta_2 \middle| \frac{\Omega\tau \sum y_i A_i - \beta_0 A_i - \beta_1 S_i A_i}{1 + \Omega\tau \sum A_i^2}, \frac{\Omega}{1 + \Omega\tau \sum A_i^2}\right).$$

As for the conjugate update of τ, we can define the likelihood as

$$\mathscr{L}(\tau | \mathbf{y}, \mathbf{S}, \mathbf{A}, \beta_0, \beta_1, \beta_2) = \prod_{i=1}^{N} \mathcal{N}\left(y_i - \beta_0 - \beta_1 S_i - \beta_2 A_i \middle| 0, \frac{1}{\tau}\right).$$

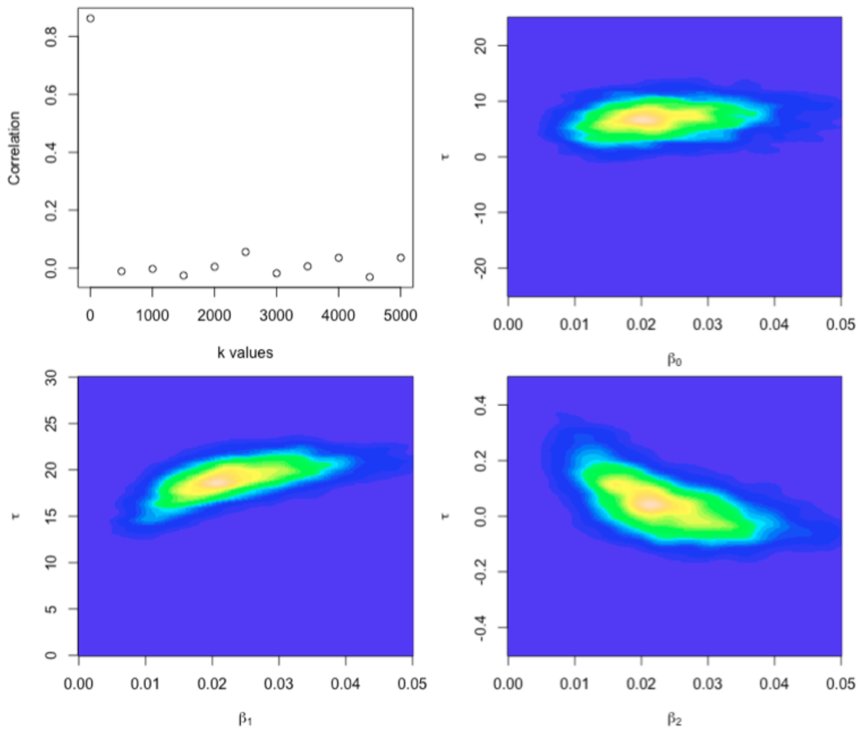
Now, combining with the prior, we can construct the posterior:

$$\begin{aligned}
\pi(\tau | \mathbf{y}, \mathbf{S}, \mathbf{A}, \beta_0, \beta_1, \beta_2) &\propto \prod_{i=1}^{N} \frac{\sqrt{\tau}}{\sqrt{2\pi}} \exp\left(\frac{-(y_i - \beta_0 - \beta_1 S_i - \beta_2 A_i)^2}{2/\tau}\right) \cdot \left(\frac{b^a \tau^{a-1} \exp(-b\tau)}{\Gamma(a)}\right) \\
&\propto \tau^{\frac{N}{2}+\alpha-1} \cdot \exp\left(\frac{-\tau}{2} \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 S_i - \beta_2 A_i)^2 - b\tau\right) \\
&\propto \tau^{\frac{N}{2}+\alpha-1} \cdot \exp\left(-\left(b + \frac{1}{2}\sum (y_i - \beta_0 - \beta_1 S_i - \beta_2 A_i)^2\right)\tau\right) \\
\pi(\tau | \mathbf{y}, \mathbf{S}, \mathbf{A}, \beta_0, \beta_1, \beta_2) &= GAMMA\left(\left(\alpha - 1 + \frac{N}{2}\right), \left(b + \frac{1}{2}\sum (y_i - \beta_0 - \beta_1 S_i - \beta_2 A_i)^2\right)\right).
\end{aligned}$$

Finally, we can construct the Gibbs Sampler in *R*. Let's begin.

```r
# Update Functions
update.tau <- function(beta_0, beta_1, beta_2){
  new.a <- a - 1 + N*0.5
  new.b <- b + 0.5*sum((price - beta_0 - beta_1*sqft - beta_2*age)^2)
  new.tau = rgamma(1, shape=new.a,scale=(1/new.b))
  return(new.tau)
}
update.beta.0 <- function(tau, beta_1, beta_2){
  new.eta <- (tau*omega*(sum(price - beta_1*sqft - beta_2*age)))/(tau*N*omega + 1)
  new.omega <- omega/(tau*N*omega + 1)
  new.beta.0 <- rnorm(1, new.eta, sqrt(new.omega))
  return(new.beta.0)
}
update.beta.1 <- function(tau, beta_0, beta_2){
  new.eta <- (omega*tau*(sum(price*sqft - beta_0*sqft -
                              beta_2*sqft*age)))/(omega*(sum(sqft^2))*tau + 1)
  new.omega <- omega/(omega*(sum(sqft^2))*tau + 1)
  new.beta.1 <- rnorm(1, new.eta, sqrt(new.omega))
  return(new.beta.1)
}
update.beta.2 <- function(tau,beta_0,beta_1){
  new.eta <- (omega*tau*(sum(price*age - beta_0*age
                              - beta_1*sqft*age)))/(omega*(sum(age^2))*tau + 1)
  new.omega <- omega/(omega*(sum(age^2))*tau + 1)
  new.beta.2 <- rnorm(1, new.eta, sqrt(new.omega))
  return(new.beta.2)
}
# Initialize Values for Algorithm
T = 10000 # Matrix
m <- matrix(0, ncol=4, nrow=T) # [tau, beta_0, beta_1, beta_2]
a = 1 # Priors
b = 1
eta = 0
omega = 10
m[1,] <- c(rgamma(1,a,b),rnorm(3,eta,omega)) # initialize parameter values
for(t in 2:T)
{
  m[t,] <- m[t-1,] # carry down the previous state
  if(t %% 4 == 1)
  {
    m[t,1] <- update.tau(m[t-1, 2], m[t-1, 3], m[t-1, 4])
  }
  if(t %% 4 == 2)
  {
    m[t,2] <- update.beta.0(m[t-1, 1], m[t-1, 3],m[t-1, 4])
  }
  if(t %% 4 == 3)
  {
    m[t,3] <- update.beta.1(m[t-1, 1], m[t-1, 2],m[t-1, 4])
  }
  if(t %% 4 == 0)
  {
    m[t,4] <- update.beta.2(m[t-1, 1], m[t-1, 2],m[t-1, 3])
  }
}
ks = seq(1,5001, by = 500)
cor.vec = 0*ks
for(k in 1:length(ks))
{
  group.1 = m[1:(T-ks[k]),1]
  group.2 = m[(ks[k]+1):T,1]
  cor.vec[k] = cor(group.1, group.2)
}
```

The credible intervals for β_0, β_1, and β_2 were 0.9838, 0.998, and 0.3464 respectively.


Correlation
k values
τ
β₀
τ
β₁
τ
β₂