STA 602 HW9

William Tirone

7.4

a)

I'll choose the usual priors for θ and Σ:

$$p(\theta) = MVN(\mu_0, \Lambda_0)$$

And we'll say $\mu_0 = (40, 40)$ and $\Lambda_0 =$ a 2x2 matrix with 150 on the diagonal and 100 on the off diagonal, which indicates that I think the average age is 40 for both husband and wife, and there is moderate correlation among the two partner's ages.

I'll use an inverse wishart for Σ:

$$p(\Sigma) = Inverse - Wishart(\nu_o, S_0^{-1})$$

To express a weak prior, I'm setting $\nu_0 = p + 2$ and $\Lambda_0^{-1} = S_0^{-1}$

b)

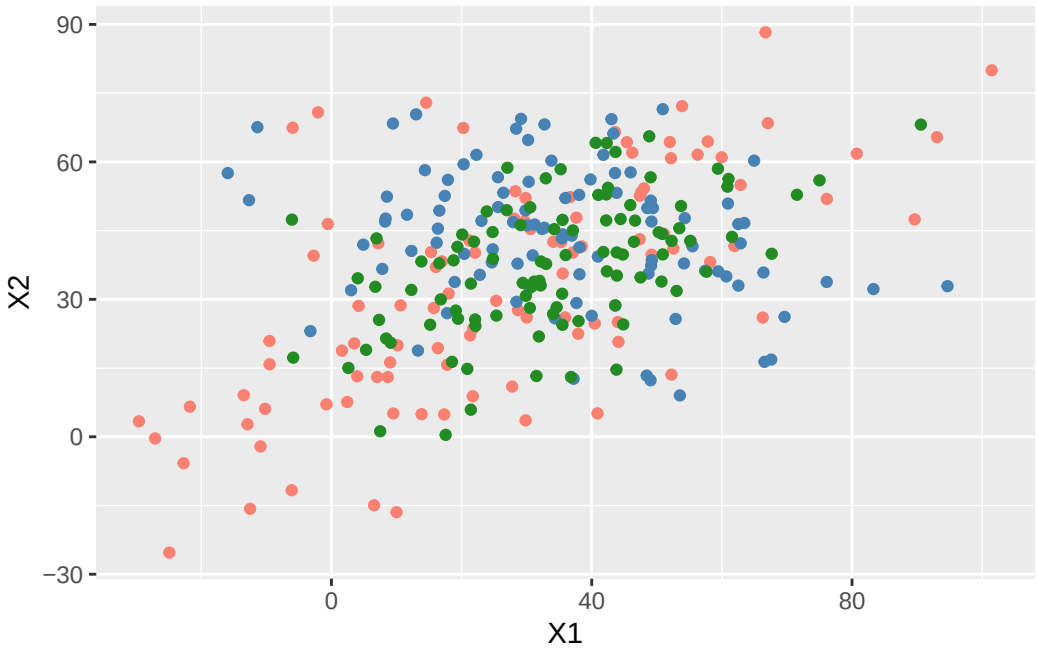
Making 3 prior predictive data sets:

```
# prior on theta
mu_0 = matrix(c(40,40),nrow=2)
L_0 = matrix(c(150,100,100,150), nrow=2, ncol=2, byrow=TRUE)

# prior on sigma
p = 2 # two features
nu_0 = p + 2
S_0 = L_0
```

```
# sampling data set 1
t1 = rmvnorm(1, mean = mu_0, sigma = L_0)
sig1 = rWishart(1,df = nu_0, S_0)
y1 = data.frame(rmvnorm(100, t1, sig1[,,1]))


# set 2
t2 = rmvnorm(1, mean = mu_0, sigma = L_0)
sig2 = rWishart(1,df = nu_0, S_0)
y2 = data.frame(rmvnorm(100, t2, sig2[,,1]))

# set 3
t3 = rmvnorm(1, mean = mu_0, sigma = L_0)
sig3 = rWishart(1,df = nu_0, S_0)
y3 = data.frame(rmvnorm(100, t3, sig3[,,1]))
```

Plotting the three prior predictive data sets. This roughly represents my belief about the age of the two married couples without looking at the data, that they're roughly linear with some weak correlation.

```
ggplot() +
  geom_point(data = y1, aes(x=X1, y=X2), colour='salmon') +
  geom_point(data = y2, aes(x=X1, y=X2), colour="steelblue") +
  geom_point(data = y3, aes(x=X1, y=X2), colour="forestgreen")
```



c)

We will need the full conditionals to build a Gibb's sampler, and we start with:

$$p(\theta, \Sigma | y_1, ...y_{100}) = p(\Sigma, \theta) p(y_1, ..., y_{100} | \theta, \Sigma)$$

and from p. 112 of PH we have that the full conditionals are:

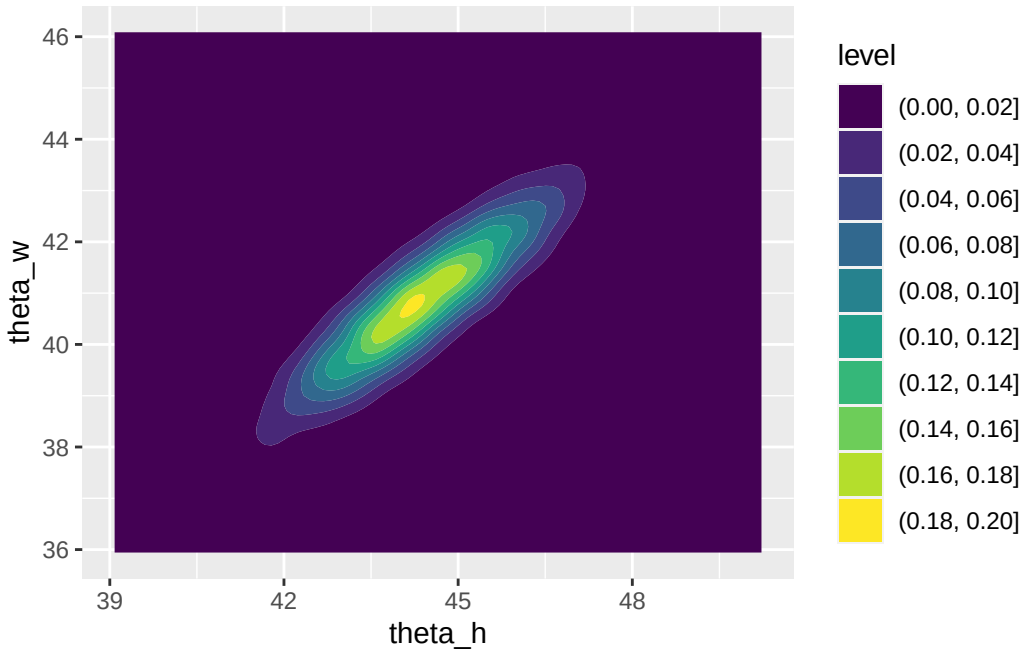
$$p(\theta | y_1, ...y_{100}, \Sigma) \sim MVN(\mu_n, \Lambda_n)$$
$$p(\Sigma | y_1, ...y_{100}, \theta) \sim I.W.(\nu_n, S_n^{-1})$$

```
#data
y = age
y_bar = apply(y,MARGIN=2,FUN=mean)
n = dim(y)[1]

# prior, same as above
# prior on theta
mu_0 = c(40,40)

# prior on sigma
```

```r
nu_0 = 4 # 2 + two features
S_0 = L_0

# loops
S = 10000

# starting point
sigma = cov(y)
THETA = NULL
SIG = array(c(0), dim = c(2,2,1))

for (i in 1:S) {

  # draw theta
  L_n = solve(solve(L_0) + n * solve(sigma))
  mu_n = L_n %*% (solve(L_0) %*% mu_0 + n * solve(sigma) %*% y_bar)
  theta = rmvnorm(1, mean = mu_n, sigma = L_n)

  # draw sigma
  # need the ,,1 element because of how rWishart stores the matrices
  S_n = S_0 + (t(y) - c(theta)) %*% t(t(y) - c(theta))
  sigma = solve(rWishart(1, nu_0 + n, solve(S_n))[,,1])

  # storing results
  THETA = rbind(THETA,theta)

  # same thing with the ,,1 element because of rWishart
  SIG = array(c(SIG, sigma), dim=c(2,2,i))

}
```

Plotting the joint posterior distribution of θ_h, θ_w

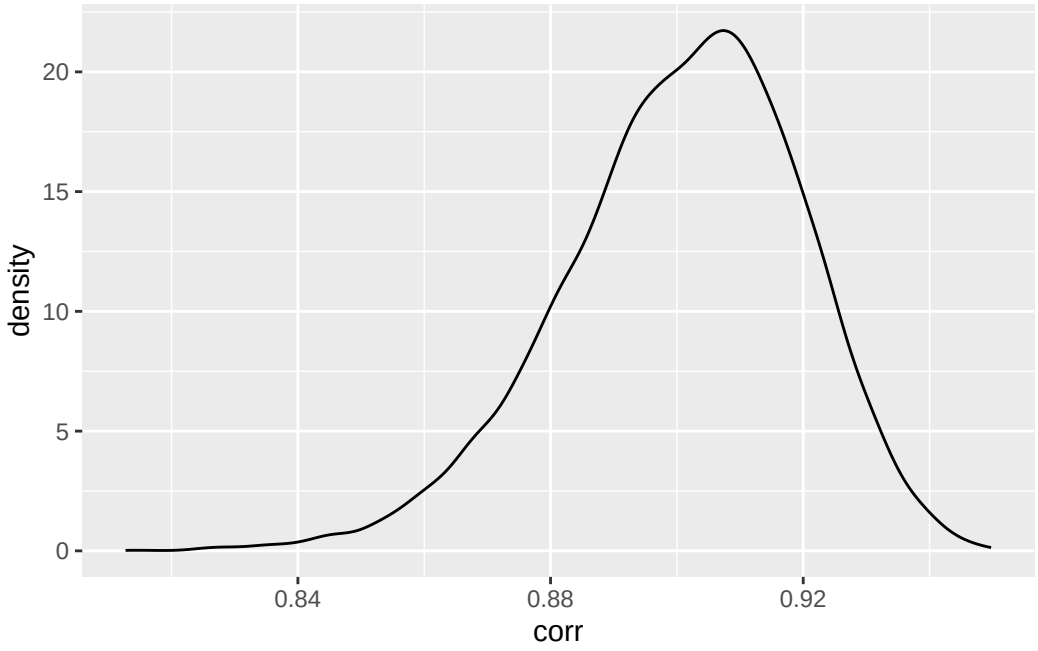
```r
THETA = data.frame(THETA)
THETA = THETA |> rename(theta_h = X1, theta_w = X2)

ggplot(THETA, aes(x=theta_h, y=theta_w)) + geom_density2d_filled()
```



Plotting the marginal posterior of the correlation between Y_h, Y_n. First we have to pull them out of the COV matrices (using the same code I wrote for HW 9). It looks like there's very high correlation among the ages of a husband and wife based on the correlation density plot which is probably expected. This also agrees with the density of theta being so stretched and not looking like a standard bivariate normal distribution.

```
custom_corr = function(m){
  return(m[2,1] / sqrt((m[1,1] * m[2,2])))
}

# calculating the correlations first
corr=c()
for (i in 1:S) {

  # orange correlation
  c1 = custom_corr(SIG[,,i])
  corr = c(corr,c1)
}

corr = data.frame(corr)

ggplot(corr, aes(corr)) + geom_density()
```

```
Warning: Removed 1 rows containing non-finite values (`stat_density()`).
```



Obtaining posterior confidence intervals:

```
th.CI = quantile(THETA[,1], probs = c(0.025,0.975))
tw.CI = quantile(THETA[,2], probs = c(0.025,0.975))
corr.CI = quantile(corr$corr, probs = c(0.025,0.975),na.rm = TRUE)

cat("theta h 95% confidence interval : ", th.CI, "\n")
```

```
theta h 95% confidence interval :  41.74736 46.96428
```

```
cat("theta w 95% confidence interval : ", tw.CI, "\n")
```

```
theta w 95% confidence interval :  38.37805 43.33555
```

```
cat("correlation 95% confidence interval : ", corr.CI)
```

```
correlation 95% confidence interval :  0.859782 0.9328084
```

d)

i)

With the Jeffrey's prior from 7.1, we had the result of the full conditionals as:

$$p(\theta|all) = MVN(\bar{y}, \Sigma/n)$$
$$p(\Sigma|all) = IW(n+1, S_\theta^{-1})$$

So I can just substitute those in to the Gibb's sampler and re-run as usual:

```
#data
y = age
y_bar = apply(y,MARGIN=2,FUN=mean)
n = dim(y)[1]

# starting point
sigma = cov(y)
THETA = NULL
SIG = array(c(0), dim = c(2,2,1))


for (i in 1:S) {

  # draw theta
  theta = rmvnorm(1, mean = y_bar, sigma = sigma/n)

  # draw sigma
  # need the ,,1 element because of how rWishart stores the matrices
  S_n = (t(y) - c(theta)) %*% t(t(y) - c(theta))
  sigma = solve(rWishart(1, n+1, solve(S_n))[,,1])

  # storing results
  THETA = rbind(THETA,theta)

  # same thing with the ,,1 element because of rWishart
  SIG = array(c(SIG, sigma), dim=c(2,2,i))

}


corr=c()
```

```
  for (i in 1:S) {

    # orange correlation
    c1 = custom_corr(SIG[,,i])
    corr = c(corr,c1)
  }

  corr = data.frame(corr)
  THETA = data.frame(THETA)

  th.CI = quantile(THETA[,1], probs = c(0.025,0.975))
  tw.CI = quantile(THETA[,2], probs = c(0.025,0.975))
  corr.CI = quantile(corr$corr, probs = c(0.025,0.975),na.rm = TRUE)

  cat("theta h 95% confidence interval : ", th.CI, "\n")
```

```
theta h 95% confidence interval :  41.68665 47.13775
```

```
  cat("theta w 95% confidence interval : ", tw.CI, "\n")
```

```
theta w 95% confidence interval :  38.33197 43.40317
```

```
  cat("correlation 95% confidence interval : ", corr.CI)
```

```
correlation 95% confidence interval :  0.8613648 0.9346892
```

ii)

From 7.2, the unit information prior results in the following full conditionals (referencing the posted hw 9 solutions and p.5 from Lecture 13):

$$\begin{aligned} p(\theta|everything) &= MVN(\mu_n, \Lambda_n) \\ \mu_n &= ((n+1)\Sigma^{-1})^{-1} \\ \Lambda_n &= \mu_n((n+1)\Sigma^{-1}\bar{y}) \\ p(\Sigma|everything) &= IW(n+p, (n+1)(S_0+S_\theta)^{-1}) \end{aligned}$$

```r
#data
y = age
y_bar = apply(y,MARGIN=2,FUN=mean)
n = dim(y)[1]

# starting point
sigma = cov(y)
THETA = NULL
SIG = array(c(0), dim = c(2,2,1))
S=10000
S_0 = L_0

for (i in 1:S) {

  # draw theta
  L_n = solve((n+1) * solve(sigma))
  mu_n = L_n %*% ((n+1) * solve(sigma) %*% y_bar)
  theta = rmvnorm(1, mean = mu_n, sigma = L_n)

  # draw sigma
  # need the ,,1 element because of how rWishart stores the matrices
  S_n = S_0 + (t(y) - c(theta)) %*% t(t(y) - c(theta))
  sigma = solve(rWishart(1, n + 2, (n+1) *solve(S_n))[,,1])

  # storing results
  THETA = rbind(THETA,theta)

  # same thing with the ,,1 element because of rWishart
  SIG = array(c(SIG, sigma), dim=c(2,2,i))

}


corr=c()
for (i in 1:S) {

  # orange correlation
  c1 = custom_corr(SIG[,,i])
  corr = c(corr,c1)
}
```

```r
corr = data.frame(corr)
THETA = data.frame(THETA)

th.CI = quantile(THETA[,1], probs = c(0.025,0.975))
tw.CI = quantile(THETA[,2], probs = c(0.025,0.975))
corr.CI = quantile(corr$corr, probs = c(0.025,0.975),na.rm = TRUE)

cat("theta h 95% confidence interval : ", th.CI, "\n")
```

```
theta h 95% confidence interval :  44.16371 44.68737
```

```r
cat("theta w 95% confidence interval : ", tw.CI, "\n")
```

```
theta w 95% confidence interval :  40.64686 41.14407
```

```r
cat("correlation 95% confidence interval : ", corr.CI)
```

```
correlation 95% confidence interval :  0.8573808 0.9327567
```

iii)

and using a diffuse prior as the problem instructed:

```r
mu_0 = c(0,0)
L_0 = diag(10^5,nrow=2,ncol=2)
S_0 = diag(1000, nrow=2, ncol=2)
nu_0 = 3

# starting point
sigma = cov(y)
THETA = NULL
SIG = array(c(0), dim = c(2,2,1))

for (i in 1:S) {

  # draw theta
  L_n = solve(solve(L_0) + n * solve(sigma))
  mu_n = L_n %*% (solve(L_0) %*% mu_0 + n * solve(sigma) %*% y_bar)
  theta = rmvnorm(1, mean = mu_n, sigma = L_n)
```

```
  # draw sigma
  # need the ,,1 element because of how rWishart stores the matrices
  S_n = S_0 + (t(y) - c(theta)) %*% t(t(y) - c(theta))
  sigma = solve(rWishart(1, nu_0 + n, solve(S_n))[,,1])

  # storing results
  THETA = rbind(THETA,theta)

  # same thing with the ,,1 element because of rWishart
  SIG = array(c(SIG, sigma), dim=c(2,2,i))

}

corr=c()
for (i in 1:S) {

  # orange correlation
  c1 = custom_corr(SIG[,,i])
  corr = c(corr,c1)
}

corr = data.frame(corr)
THETA = data.frame(THETA)

th.CI = quantile(THETA[,1], probs = c(0.025,0.975))
tw.CI = quantile(THETA[,2], probs = c(0.025,0.975))
corr.CI = quantile(corr$corr, probs = c(0.025,0.975),na.rm = TRUE)

cat("theta h 95% confidence interval : ", th.CI, "\n")
```

```
theta h 95% confidence interval :  41.6761 47.1875
```

```
cat("theta w 95% confidence interval : ", tw.CI, "\n")
```

```
theta w 95% confidence interval :  38.25535 43.51061
```

```
cat("correlation 95% confidence interval : ", corr.CI)
```

```
correlation 95% confidence interval :  0.7940918 0.9003278
```

e)

Comparing the following:

- i) It looks like the Jeffrey's prior produces extremely similar results, which is pretty interesting actually. It seems like that indicates that the data dominates the posterior samples and the prior choice didn't make a huge difference.
- ii) Assuming I found my full conditionals correctly, the CI for θ_h is super narrow, which is impressive if I was correct. The interval for θ_w is also much narrower than the other priors, and the correlation is more similar to Jeffrey's prior than the diffuse or my own choice. I would say this prior is probably preferable.
- iii) The C.I.'s for θ_h and θ_w are quite similar to part c), though the correlation interval is a bit lower at (0.792,0.899) compared to (0.859, 0.932) for my prior.

Overall, it doesn't seem like my choice of prior has that much of an impact, and considering I made up the numbers, it doesn't seem like a great idea to use my prior. Though, with a large n value, the data dominates the posterior samples anyway. So, in the case of small n, we would probably want to do a sensitivity analysis based on our choice of prior to see how it affects the posterior samples.

7.5

a)

calculating a variety of values for part a

```
tA = mean(interrupt$yA, na.rm = TRUE)
tB = mean(interrupt$yB, na.rm = TRUE)

sigmaA = var(interrupt$yA, na.rm = TRUE)
sigmaB = var(interrupt$yB, na.rm = TRUE)

p_hat = cor(interrupt, use = "complete.obs")[1,2]
```

b)

imputing values:

```r
filled_data = interrupt

filled_data[is.na(filled_data$yB),][,2] = tB + (filled_data[is.na(filled_data$yB),][,1] -

filled_data[is.na(filled_data$yA),][,1] = tA + (filled_data[is.na(filled_data$yA),][,2] -
```

Doing a paired-sample t-test. We would reject the null that the true difference in means is equal to zero given the small p-value of 0.00177.

```r
# borrowed code from http://www.sthda.com/english/wiki/paired-samples-t-test-in-r
t.test(filled_data$yA, filled_data$yB, paired = TRUE, alternative = "two.sided")
```

```
    Paired t-test

data:  filled_data$yA and filled_data$yB
t = -3.2807, df = 57, p-value = 0.00177
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.9850730 -0.2383347
sample estimates:
mean of the differences
             -0.6117038
```

c)

First, I'm not entirely convinced this data is *not* MNAR, which would make this sampling method impossible, but we will proceed as if it's MAR / MCAR. Adapting code from Lecture 14 p. 23 below.

Here's how I approached this problem: I couldn't figure out the full conditionals using Jeffrey's prior directly, but consulting these lecture notes from a class at Duke in 2010, the Jeffrey's prior is a limiting case of the semi-conjugate priors. To handle this, I modified the priors below, and in the Gibb's sampler, I adjusted the code to avoid inverting zero matrices. This was too avoid re-solving for the full conditionals, but I think actually makes sense.

I'm not too sure if Jeffrey's prior would change the full conditional of the missing data though, but it doesn't seem like it would. That's really my main confusion here, but I haven't modified that part of the sampler at all.

```r
# getting stuff from data

y.original = interrupt
n <- nrow(y.original) # sample size
p <- ncol(y.original) # dimensionality
I <- !is.na(y.original) # missingness indicator, TRUE if present, 0 if missing
```

```r
# prior for theta
mu.0 <- c(0,0)
Lambda.0 = matrix(c(0,0,0,0),ncol=2,byrow=TRUE)

# prior for Sigma
nu.0 <- 1 # a very weak prior
S0 <- matrix(c(0,0,0,0),ncol=2,byrow=TRUE)
```

```r
niter <- 10000 # total number of iterations
nburnin <- 1000 # 1000 burn-in steps

ybar.original <- apply(y.original,2,mean,na.rm=TRUE) # the column means of the original da

y <- y.original ## y holds the imputed data (y.obs,y.mis)
# initialize y by filling in the NAs with the corresponding column means
for (i in 1:p) {
  y[I[,i]==0,i] <- ybar.original[i]
}

## Proceed as before like there are no missing data
ybar <- apply(y,2,mean)
nu.n <- nu.0 + n

THETA <- matrix(NA,nrow=niter,ncol=p) # matrix for storing the draws for theta
colnames(THETA) <- c("theta1","theta2")

THETA.init <- ybar # Initial values set to sample mean
THETA.curr <- THETA.init # the theta value at current iteration

SIGMA <- matrix(NA,nrow=niter,ncol=p*p) # matrix for storing the draws for Sigma
colnames(SIGMA) <- c("sigma11","sigma12","sigma21","sigma22")

SIGMA.init <- cov(y) # intial value set to sample covariance
```

```
SIGMA.curr <- SIGMA.init # the Sigma value at current iternation
```

```
for (t in 1:niter) {

  Lambda.n <- solve(n*solve(SIGMA.curr))
  mu.n <- Lambda.n %*% (n*solve(SIGMA.curr,ybar))

  ## Update theta
  THETA.curr <- rmvnorm(1,mean=mu.n,sigma=Lambda.n)

  ## Update Sigma
  S.theta <- (t(y)-c(THETA.curr))%*% t(t(y)-c(THETA.curr))
  SIGMA.curr <- riwish(v=nu.n,S=S.theta)

  ## Impute the missing data
  for (i in 1:n) {
    var.obs = which(I[i,]) ## which variables are observed
    var.mis = which(!I[i,]) ## which variables are missing

    if (length(var.mis) > 0) { ## if there are missing values

      SIGMA.obs <- SIGMA.curr[var.obs,var.obs] # Sigma11
      SIGMA.mis <- SIGMA.curr[var.mis,var.mis] # Sigma22

      SIGMA.mis.obs <- SIGMA.curr[var.mis,var.obs] # Sigma21
      SIGMA.obs.mis <- t(SIGMA.mis.obs) # Sigma12

      y[i,var.mis] <- rnorm(1, mean=THETA.curr[var.mis]+
      SIGMA.mis.obs%*%solve(SIGMA.obs,y[i,var.obs]-THETA.curr[var.obs]),
      sd=sqrt(SIGMA.mis-SIGMA.mis.obs%*%solve(SIGMA.obs,SIGMA.obs.mis)))
    }
  }
  ybar <- apply(y,2,mean)

  ## Save the current iteration
  THETA[t,] <- THETA.curr
  SIGMA[t,] <- SIGMA.curr
}
```

Now computing values for the rest of the question. We know first off that simply imputing values statically ignores the uncertainty in the missing data, so in that sense, we could probably ignore those results in favor of the Gibb's sampling approach because this takes the uncertainty

into account with each iteration.

However, in this case, it looks like the two approaches agree, so maybe that gives us a bit more confidence in our approach. The Gibb's sampler approach yields a posterior mean and confidence interval that imply that the true difference, $\theta_A - \theta_B \neq 0$.

```
tA.minus.tB = mean(THETA[,1] - THETA[,2])
cat("Posterior mean for theta A - theta B : ", tA.minus.tB)
```

```
Posterior mean for theta A - theta B :  -0.6144591
```

```
CI = quantile(THETA[,1] - THETA[,2], probs = c(.025,0.975))
cat("Posterior 95% CI for theta A - theta B : ", CI)
```

```
Posterior 95% CI for theta A - theta B :  -1.308303 0.05118288
```

7.6

a)

First, separate the data and set up priors, then run the Gibb's sampler. It seems like every single variable is higher in the diabetes setting, which looked like an error at first but may actually make sense. If a patient has diabetes, all of the measured factors would be higher. In addition, visually examining the data, it looks like all of the measured factors were higher in the diabetes set, so the posteriors would be higher as well.

```
# yes diabetes
az_d = az[az$diabetes == "Yes",]
# no diabetes
az_n =az[az$diabetes == "No",]

# priors for diabetes
y.d = az_d[,-8]
mu0.d = c(apply(y.d, 2, mean))
L0.d = cov(y.d)
S0.d = L0.d
nu0.d = 9
n.d = dim(y.d)[1]
y_bar.d = mu0.d
```

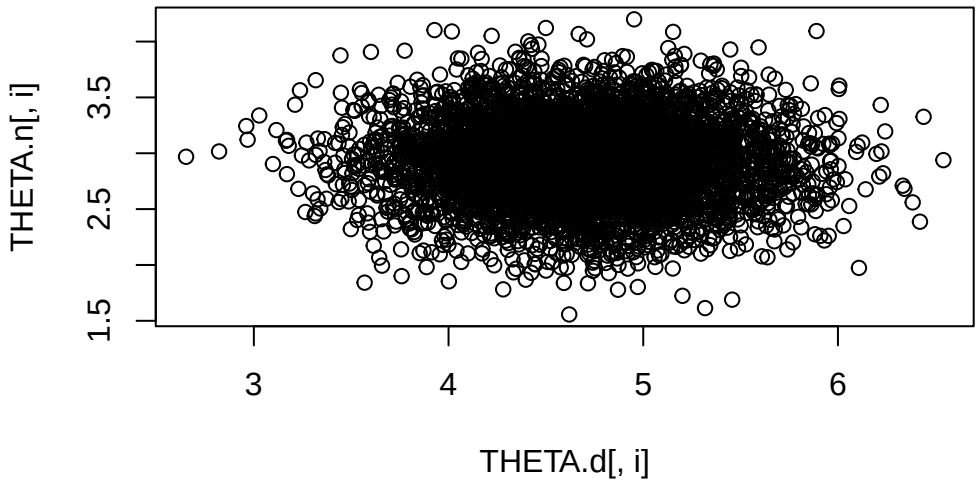
```r
# priors for no diabetes
y.n = az_n[,-8]
mu0.n = c(apply(y.n, 2, mean))
L0.n = cov(y.n)
S0.n = L0.n
nu0.n = 9
n.n = dim(y.n)[1]
y_bar.n = mu0.n
```

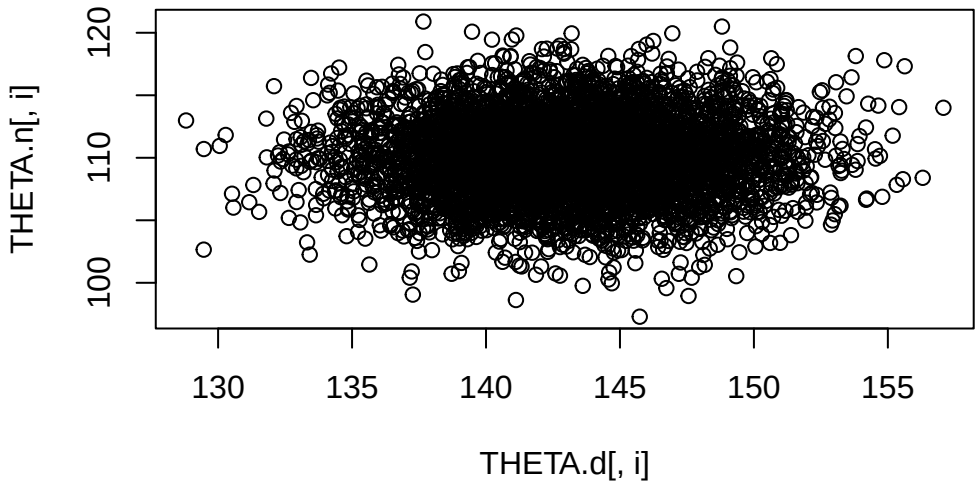
```r
# starting point
sigma.n = L0.n
sigma.d = L0.d
THETA.n = THETA.d = NULL
SIG.n = SIG.d = array(c(0), dim = c(2,2,1))

S = 5000

for (i in 1:S) {

  # draw thetas
  Ln.d = solve(solve(L0.d) + n.d * solve(sigma.d))
  mu.d = Ln.d %*% (solve(L0.d) %*% mu0.d + n.d * solve(sigma.d) %*% y_bar.d)
  theta.d = rmvnorm(1, mean = mu.d, sigma = Ln.d)

  Ln.n = solve(solve(L0.n) + n.n * solve(sigma.n))
  mu.n = Ln.n %*% (solve(L0.n) %*% mu0.n + n.n * solve(sigma.n) %*% y_bar.n)
  theta.n = rmvnorm(1, mean = mu.n, sigma = Ln.n)


  # draw sigmas
  # need the ,,1 element because of how rWishart stores the matrices
  S.d = S0.d + (t(y.d) - c(theta.d)) %*% t(t(y.d) - c(theta.d))
  sigma.d = solve(rWishart(1, nu0.d + n, solve(S.d))[,,1])

  S.n = S0.n + (t(y.n) - c(theta.n)) %*% t(t(y.n) - c(theta.n))
  sigma.n = solve(rWishart(1, nu0.n + n, solve(S.n))[,,1])

  # storing results
  THETA.d = rbind(THETA.d,theta.d)
  THETA.n = rbind(THETA.n,theta.n)
```

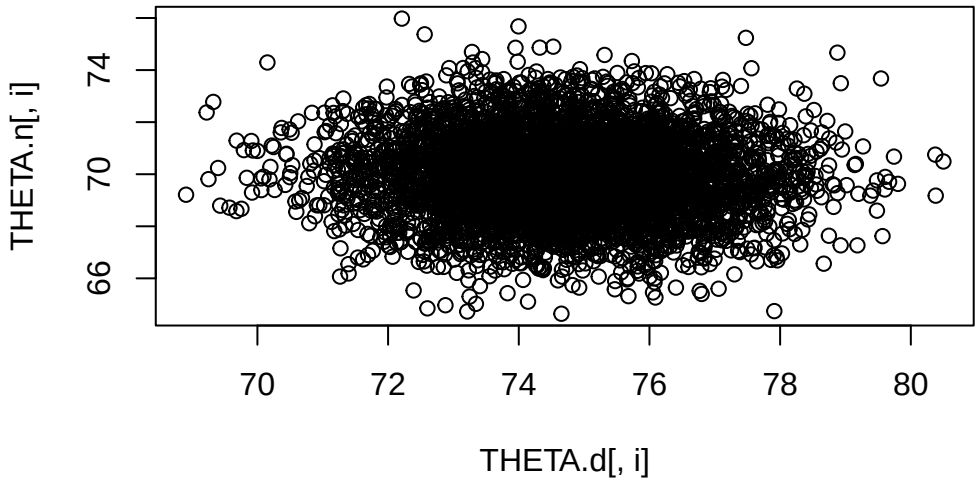
```
  # same thing with the ,,1 element because of rWishart
  SIG.d = array(c(SIG.d, sigma.d), dim=c(7,7,i))
  SIG.n = array(c(SIG.n, sigma.n), dim=c(7,7,i))

}
THETA.d = data.frame(THETA.d)
THETA.n = data.frame(THETA.n)
names(THETA.d) = names(THETA.n) = names(az_d[,-8])
```

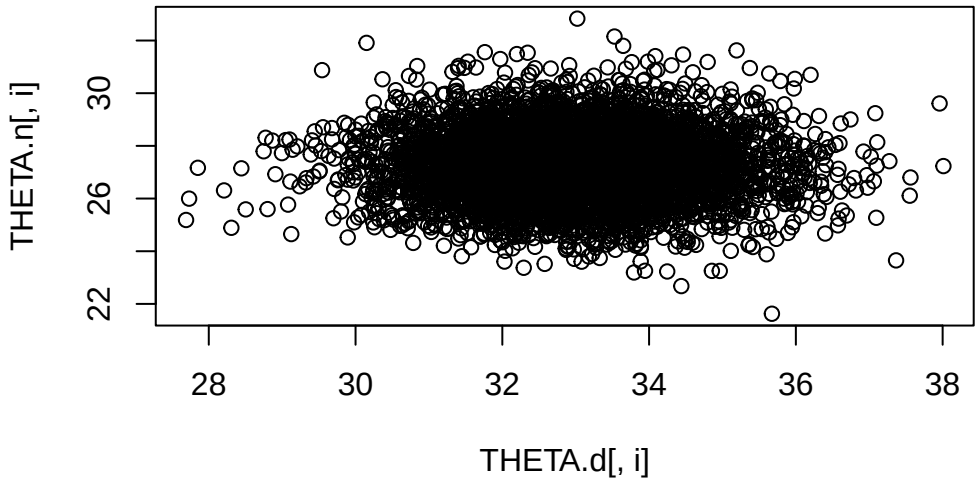
Here, computing some plots to visually look at the variables and computing $P(\theta_{d,j} > \theta_{n,j}|Y)$

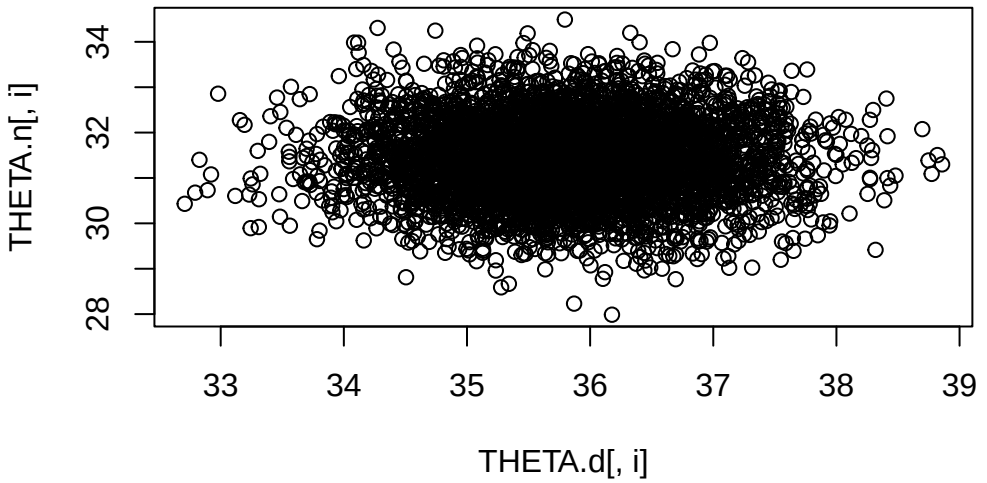
```
for (i in 1:7) {
  plot(THETA.d[,i], THETA.n[,i])
}
```

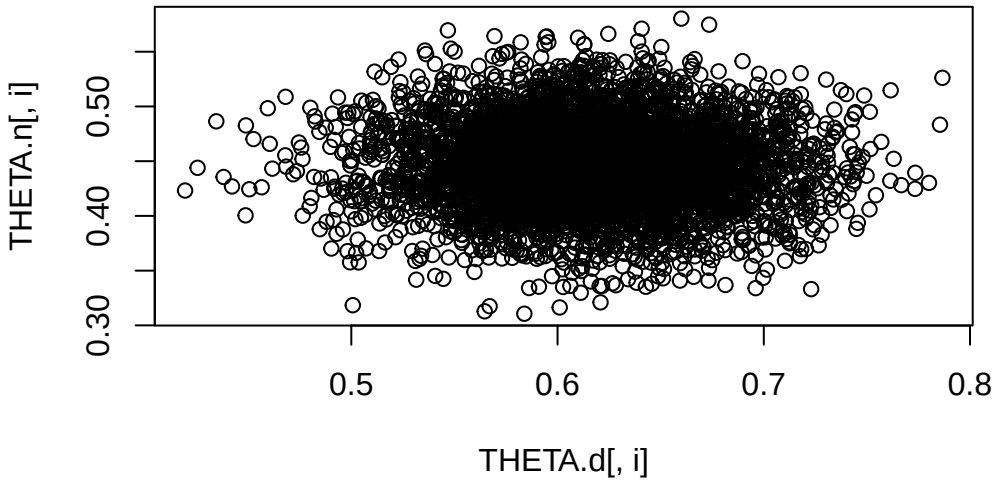


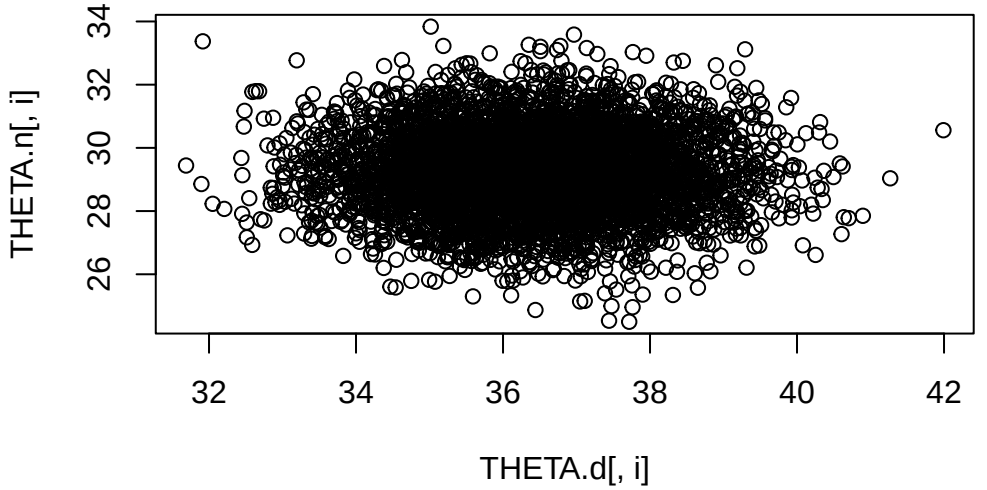

THETA.n[, i]
100
110
120
130
135
140
145
150
155
THETA.d[, i]


THETA.n[, i]
66
70
74
70
72
74
76
78
80
THETA.d[, i]


THETA.n[, i]
22
26
30
28
30
32
34
36
38
THETA.d[, i]


THETA.n[, i]
28
30
32
34
33
34
35
36
37
38
39
THETA.d[, i]


THETA.n[, i]
0.30
0.40
0.50
0.5
0.6
0.7
0.8
THETA.d[, i]


THETA.n[, i]
26
28
30
32
34
32
34
36
38
40
42
THETA.d[, i]

```
for (i in 1:7) {
  print(mean(THETA.d[,i] > THETA.n[,i]))
}
```

```
[1] 0.9968
[1] 1
[1] 0.9808
[1] 0.9992
[1] 0.9998
[1] 0.9956
[1] 0.9998
```

b)

First, calculating the average posterior of the matrices by summing them element-wise then dividing by the sample size.

```
# initialize the right shapes
out.d = array(c(0), dim = c(7,7,1))
out.n = array(c(0), dim = c(7,7,1))

# adding them up
for (i in 1:5000) {

  a = SIG.d[,,i]
  out.d[,,1] = out.d[,,1] + a

  b = SIG.n[,,i]
  out.n[,,1] = out.n[,,1] + b
}

# averaging over S iterations
sig.d.average = out.d / S
sig.n.average = out.n / S
```

Now, plotting the entries of the means of Σ_d and Σ_n, I removed the 9th value of to make the plot look a little better. It doesn't look like there are huge differences among the two covariance matrices since the relationship is fairly linear and the elements of each covariance matrix appear to have strong correlation.

```r
sigma.means = data.frame(c(sig.d.average), c(sig.n.average))
colnames(sigma.means) = c("sig.d", "sig.n")
sigma.means = sigma.means[-9,]

plot(sigma.means$sig.d, sigma.means$sig.n)
```