Assignment #7

Yutae Lee (626005947)

2022-10-31

Problem 7.1

a)

The p_J is uniform respect to θ. Thus integral would be $\infty \neq 1$

b) Knowing that $p_J(\theta, \Sigma|y_1, \dots, y_n) \propto p(\theta, \Sigma)$

We can get:

$$\begin{aligned} p_J(\theta, \Sigma \mid y_1, \dots, y_n) &\propto p(\theta, \Sigma) \times p(y_1, \dots, y_n \mid \theta, \Sigma) \\ &\propto \left[|\Sigma|^{-(p+2)/2}\right] \times \left[|\Sigma|^{-n/2} \exp\big(-\mathrm{tr}(\mathbf{S}_\theta \Sigma^{-1})\big)\right] \\ &\propto |\Sigma|^{-(p+n+2)/2} \exp(-\sum_{i=1}^{n} (y_i - \theta)^T \Sigma^{-1} (y_i - \theta)/2) \end{aligned}$$

From that we can get $p_J(\theta \mid y_1, \dots, y_n, \Sigma)$:

$$\begin{aligned} p_J(\theta \mid y_1, \dots, y_n, \Sigma) &\propto \exp(-\sum_{i=1}^{n} (y_i - \theta)^T \Sigma^{-1} (y_i - \theta)/2) \\ &= \exp(-n(\bar{y} - \theta)^T \Sigma^{-1} (\bar{y} - \theta)/2) \\ &= \mathrm{dnormal}(\bar{y}, \Sigma/n) \end{aligned}$$

Lastly getting the $p_J(\Sigma \mid y_1, \dots, y_n, \theta)$:

$$\begin{aligned} p_J(\Sigma \mid y_1, \dots, y_n, \theta) &\propto |\Sigma|^{-(p+n+2)/2} \exp(-\sum_{i=1}^{n} (y_i - \theta)^T \Sigma^{-1} (y_i - \theta)/2) \\ &\propto \text{dinverse-wishart}\left(n+1, \left((y_i - \theta)^T (y_i - \theta)\right)^{-1}\right) \end{aligned}$$

Problem 7.3

Let’s first read the two datas as matrix.

```
bluecrab = as.matrix(read.table(url('https://www2.stat.duke.edu/courses/Fall09/sta290/datasets/Hoffdata/bluecrab.dat')))
orangecrab = as.matrix(read.table(url('https://www2.stat.duke.edu/courses/Fall09/sta290/datasets/Hoffdata/orangecrab.dat')))
```

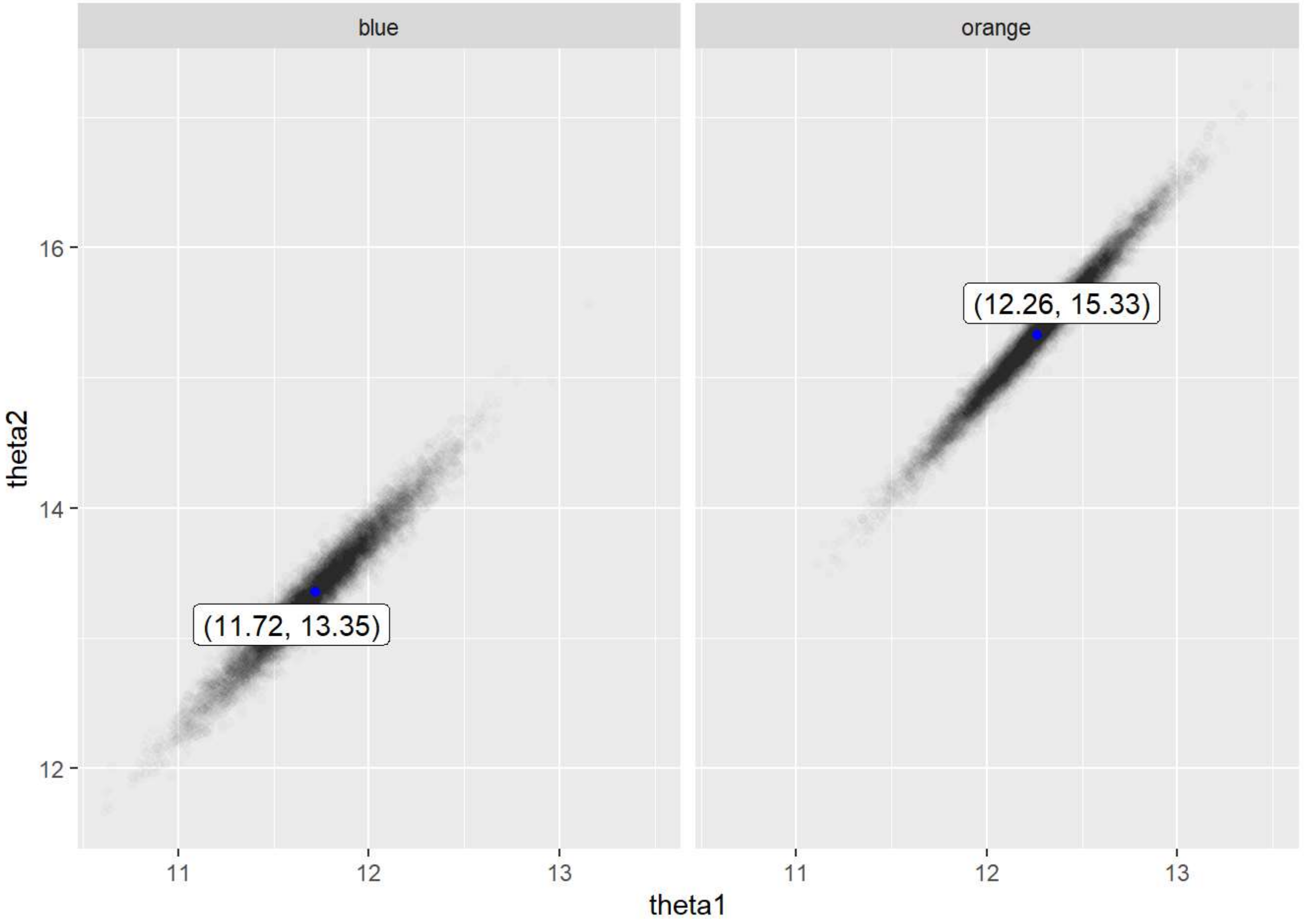
a)

```
set.seed(32)
crab.mcmc = lapply(list('bluecrab' = bluecrab, 'orangecrab' = orangecrab), function(crab) {
  p = ncol(crab)
  n = nrow(crab)
  ybar = colMeans(crab)

  # Prior parameters

  mu0 = ybar
  lambda0 = s0 = cov(crab)
  nu0 = 4

  S = 10000
  Theta = matrix(nrow = S, ncol = p)
  Sigma = array(dim = c(p, p, S))

  sigma = s0
  library(MASS)
  inv = solve

  for (s in 1:S) {
    lambdan = inv(inv(lambda0) + n * inv(sigma))
    mun = lambdan %*% (inv(lambda0) %*% mu0 + n * inv(sigma) %*% ybar)
    theta = mvrnorm(n = 1, mun, lambdan)

    resid = t(crab) - c(theta)
    stheta = resid %*% t(resid)
    sn = s0 + stheta
    sigma = inv(rWishart(1, nu0 + n, inv(sn))[, , 1])

    Theta[s, ] = theta
    Sigma[, , s] = sigma
  }

  list(theta = Theta, sigma = Sigma)
})
```

The list crab.mcmc is the list of our posteriors.

b)

```
## 필요한 패키지를 로딩중입니다: ggplot2
```



Using these plots we can clearly see that Orange is generally bigger than the Blue crabs.

```
mean(orangecrab.df$theta1 > bluecrab.df$theta1)
```

```
## [1] 0.896
```

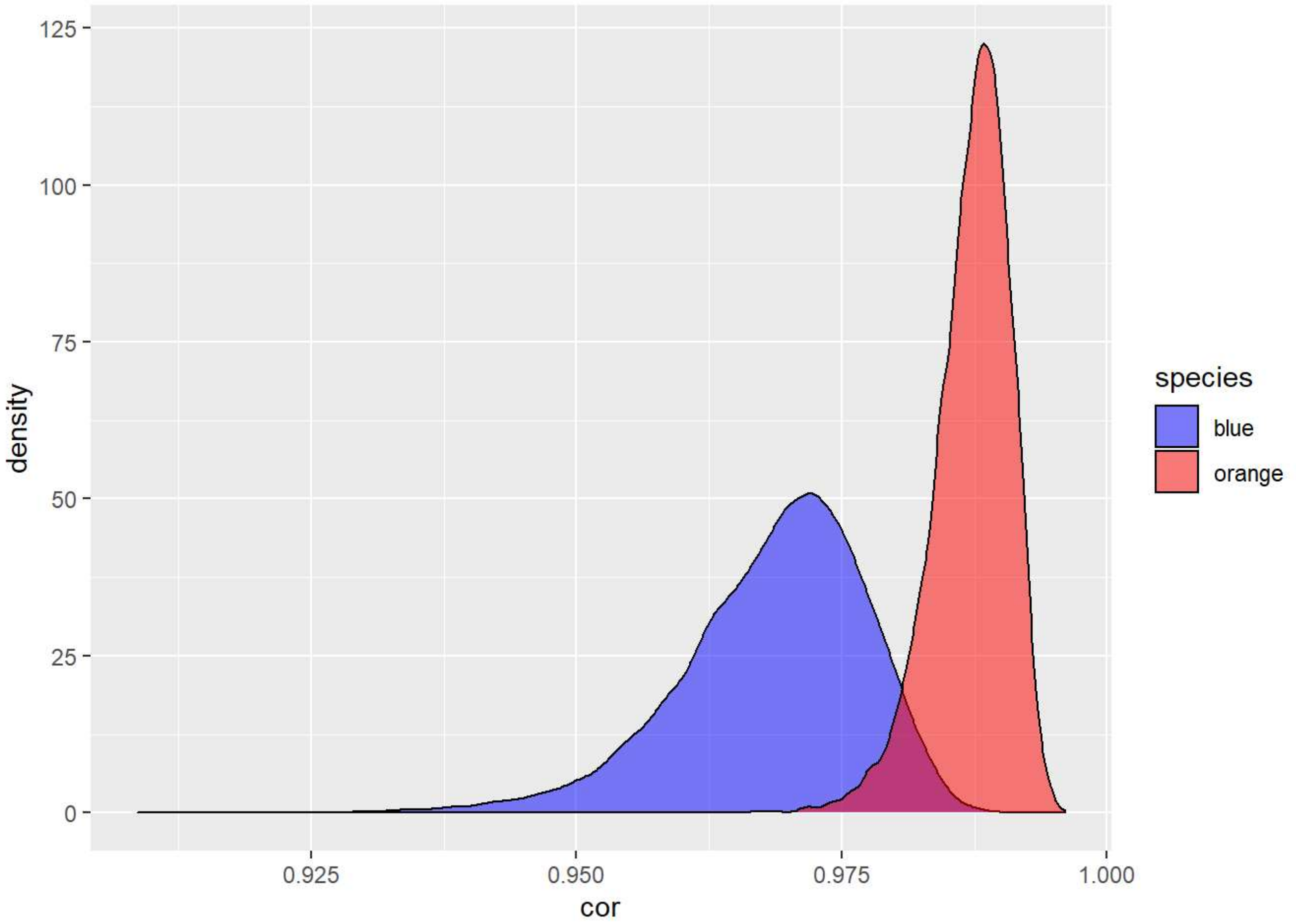
```
mean(orangecrab.df$theta2 > bluecrab.df$theta2)
```

```
## [1] 0.9983
```

We can see that for the first measurement orange crab is 90% of times greater than blue crab. For the second measurement it is nearly 99.8% of times greater.

c)

```
bluecrab.cor = apply(crab.mcmc$bluecrab$sigma, MARGIN = 3, FUN = function(covmat) {
  covmat[1, 2] / (sqrt(covmat[1, 1] * covmat[2, 2]))
})
orangecrab.cor = apply(crab.mcmc$orangecrab$sigma, MARGIN = 3, FUN = function(covmat) {
  covmat[1, 2] / (sqrt(covmat[1, 1] * covmat[2, 2]))
})
cor.df = data.frame(species = c(rep('blue', length(bluecrab.cor)), rep('orange', length(orangec
rab.cor))),
                    cor = c(bluecrab.cor, orangecrab.cor))
ggplot(cor.df, aes(x = cor, fill = species)) +
  geom_density(alpha = 0.5) +
  scale_fill_manual(values = c('blue', 'red'))
```



We can also see that Orange crab generally has higher correlation between the two measure.

```
mean(bluecrab.cor < orangecrab.cor)
```

```
## [1] 0.9883
```

7.4

Let’s get the data as matrix just like previous question.

```
agehw = as.matrix(read.table(url('https://www2.stat.duke.edu/courses/Fall09/sta290/datasets/Hoffdata/agehw.dat')))
colnames(agehw) = agehw[1, ]
agehw = agehw[-1, ]
agehw = matrix(as.numeric(agehw), nrow = 100)
```

a)

I am going to set $\boldsymbol{\mu}_0 = ((20+100)/2, (20+100)/2) = (60, 60)^T$. Because with general knowledge, I believe that married couple will have an age greater 20 but less than 100.

I know that 20 will be thee minimum age for most people to get married and 100 is the maximum age for most people to live, I expect that there would be more married people centered around μ_0 = 60 with variance of $20^2 = 400$. This means that 95% of the prior is going to be within the range 20 to 100.

Also I believe that there would be high correlation between the ages of each couple and I am going to set that as .80.

Which I get

$$0.80 = \frac{\sigma_{1,2}}{400}$$

$$\sigma_{1,2} = 320$$

So I am going to set

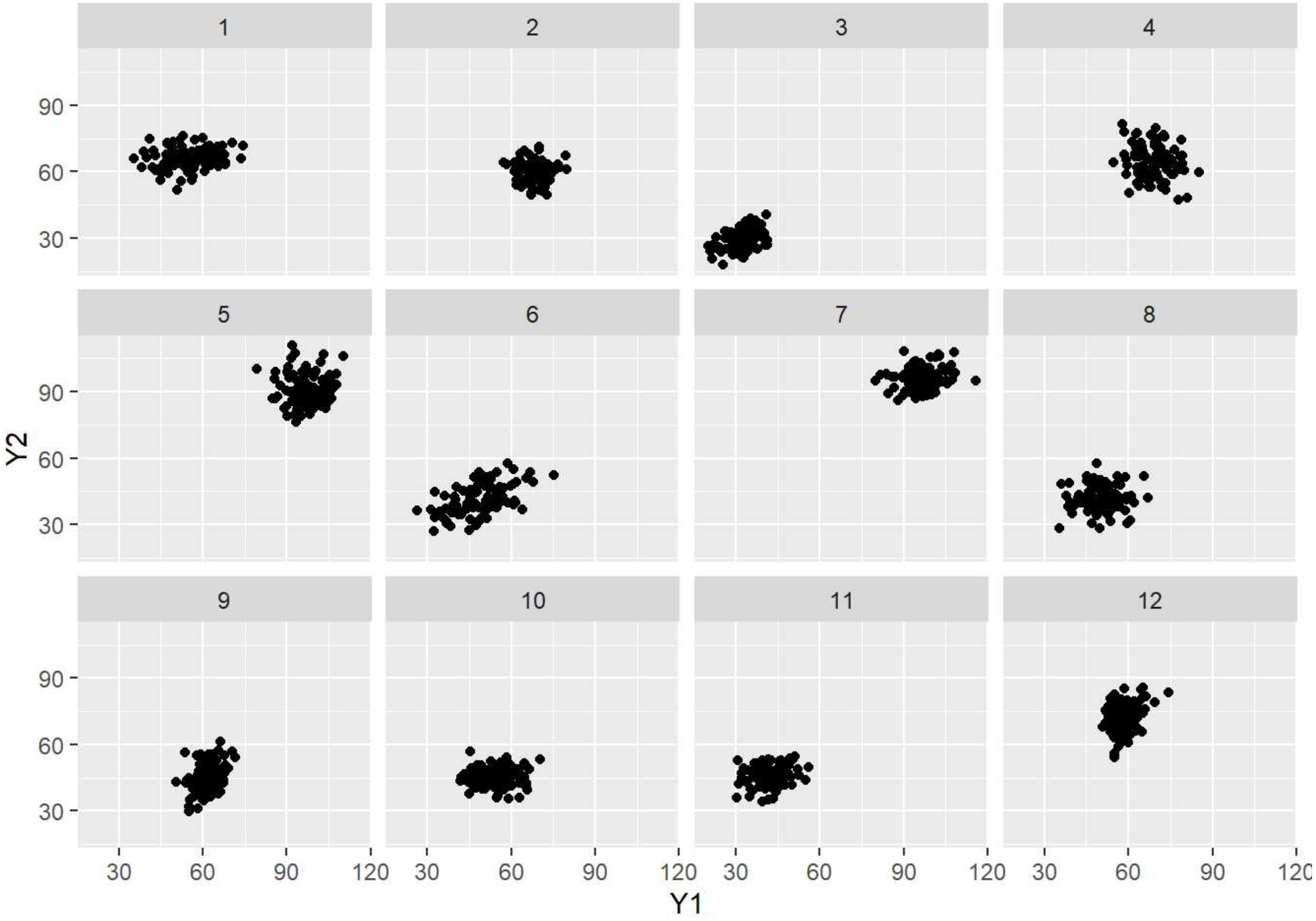
$$\Lambda_0 = \begin{bmatrix} 400 & 320 \\ 320 & 400 \end{bmatrix}$$

For the variance, I will set $\mathbf{S}_0^{-1} = \Lambda_0$ and $\nu_0 = p + 2 = 4$.

b)

```
set.seed(32)
Y = agehw
p = ncol(agehw)
n = nrow(agehw)
ybar = colMeans(agehw)
mu0 = rep(60, p)
lambda0 = s0 = rbind(c(400, 20), c(320, 400))
# nu0 = p + 2
nu0 = p + 2 + 10
```

```
N = 100
S = 12
Y_preds = lapply(1:S, function(s) {
  theta = mvrnorm(n = 1, mu0, lambda0)
  sigma = solve(rWishart(1, nu0, solve(s0))[, , 1])
  Y_s = mvrnorm(n = 100, theta, sigma)
  data.frame(Y1 = Y_s[, 1], Y2 = Y_s[, 2], dataset = s)
})
Y_comb = do.call(rbind, Y_preds)
ggplot(Y_comb, aes(x = Y1, y = Y2)) +
  geom_point() +
  facet_wrap(~ dataset)
```



c

```
set.seed(32)
S = 10000
do_mcmc = function(Y, mu0, lambda0, s0, nu0) {
  ybar = colMeans(Y)
  p = ncol(Y)
  n = nrow(Y)
  THETA = matrix(nrow = S, ncol = p)
  SIGMA = array(dim = c(p, p, S))

  sigma = cov(Y)


  for (s in 1:S) {
    lambdan = solve(solve(lambda0) + n * solve(sigma))
    mun = lambdan %*% (solve(lambda0) %*% mu0 + n * solve(sigma) %*% ybar)
    theta = mvrnorm(n = 1, mun, lambdan)

    # Update sigma
    resid = t(Y) - c(theta)
    stheta = resid %*% t(resid)
    sn = s0 + stheta
    sigma = solve(rWishart(1, nu0 + n, solve(sn))[, , 1])

    THETA[s, ] = theta
    SIGMA[, , s] = sigma
  }

  list(theta = THETA, sigma = SIGMA)
}
my_prior_mcmc = do_mcmc(agehw, mu0, lambda0, s0, nu0)
THETA = my_prior_mcmc$theta
SIGMA = my_prior_mcmc$sigma
print_quantiles = function(THETA, SIGMA) {
  print("For Husband")
  print(quantile(THETA[, 1], probs = c(0.025, 0.5, 0.975)))
  print("For Wife")
  print(quantile(THETA[, 2], probs = c(0.025, 0.5, 0.975)))
  cors = apply(SIGMA, MARGIN = 3, FUN = function(covmat) {
    covmat[1, 2] / (sqrt(covmat[1, 1] * covmat[2, 2]))
  })
  print("Correlation")
  print(quantile(cors, probs = c(0.025, 0.5, 0.975)))
}
print_quantiles(THETA, SIGMA)
```

```
## [1] "For Husband"
##     2.5%      50%    97.5%
## 42.03410 44.49388 46.99527
## [1] "For Wife"
##     2.5%      50%    97.5%
## 38.65348 40.95015 43.30816
## [1] "Correlation"
##      2.5%       50%     97.5%
## 0.8373819 0.8850628 0.9192012
```

d)

i.

```
set.seed(32)
Theta = matrix(nrow = S, ncol = p)
Sigma = array(dim = c(p, p, S))
sigma = cov(Y)

for (s in 1:S) {
  # Update theta
  theta = mvrnorm(n = 1, ybar, sigma / n)

  # Update sigma
  resid = t(Y) - c(theta)
  stheta = resid %*% t(resid)
  sigma = solve(rWishart(1, n + 1, solve(stheta))[, , 1])

  Theta[s, ] = theta
  Sigma[, , s] = sigma
}
print_quantiles(Theta, Sigma)
```

```
## [1] "For Husband"
##     2.5%      50%    97.5%
## 41.71029 44.40980 47.12925
## [1] "For Wife"
##     2.5%      50%    97.5%
## 38.36223 40.87941 43.43414
## [1] "Correlation"
##      2.5%       50%     97.5%
## 0.8612598 0.9043510 0.9346393
```

iii.

```
set.seed(32)
mu0 = rep(0, p)
lambda0 = 10^5 * diag(p)
s0 = 1000 * diag(p)
nu0 = 3
diffuse_mcmc = do_mcmc(agehw, mu0, lambda0, s0, nu0)
print_quantiles(diffuse_mcmc$theta, diffuse_mcmc$sigma)
```

```
## [1] "For Husband"
##     2.5%      50%    97.5%
## 41.67099 44.41281 47.18572
## [1] "For Wife"
##     2.5%      50%    97.5%
## 38.30760 40.87932 43.48216
## [1] "Correlation"
##      2.5%       50%     97.5%
## 0.7923628 0.8551593 0.8997521
```

e.

I can see that generally prior doesn’t have much influence in our confidence interval. I am assuming this is because our sample size is big enough.