Joseph High
Hopkins ID: 9E1FDC

EN.553.732; Homework 2
Problems 5 and 6

Problem 5

Part (a):
R Code

```
#Problem 5
#Part (a)

mu0=5
sigma0=4
v0=2
k0=1
schooldata=list()

schooldata[1]<-read.table("school1.txt")
schooldata[2]<-read.table("school2.txt")
schooldata[3]<-read.table("school3.txt")

n = sapply(schooldata, length)
ybar=sapply(schooldata, mean)
s=sapply(schooldata, var)

kn=k0+n
vn=v0+n
mun=(k0*mu0+n*ybar)/kn
sigman=(v0*sigma0+(n-1)*s+k0*n*(ybar-mu0)^2/kn)/(vn)
sigma=mu=matrix(0, 10000, 3, dimnames = list(NULL, c("school1", 
"school2", "school3")))
for (i in c(1, 2, 3)){
  sigma[,i]=1/rgamma(10000, vn[i]/2, vn[i]*sigman[i]/2)
  mu[,i]=rnorm(10000, mun[i], (sigma[,i]/kn[i])^0.5)
}

#Computing posterior means and 95% confidence interval for mu

colMeans(mu)
apply(mu, 2, function(x) {
  quantile(x, c(0.025, 0.975))
})

#Computing posterior means and 95% confidence interval for standard 
deviation

colMeans(sqrt(sigma))
apply(sqrt(sigma), 2, function(x) {
  quantile(x, c(0.025, 0.975))
})
```

Results:

Posterior Means:

```
school1  school2  school3 
9.290606 6.963136 7.814114
```

95% CI for mean:

```
       school1  school2  school3
2.5%   7.75762 5.150658 6.172948
97.5% 10.84183 8.787480 9.427163
```

Posterior Means for standard deviation:

```
school1  school2  school3 
3.905729 4.402176 3.741269
```

95% CI for standard deviation:

```
       school1  school2  school3
2.5%  3.000531 3.349973 2.800034
97.5% 5.157399 5.889208 5.110928
```

Problem 5, Part b

R Code

```
#Part b

#combinat package installed for permn function use. Used to generate 
all 6 permutations of {1,2,3}.

mu_ranks= t(apply(mu, 1, rank))
prob_ranks= list()
for (p in permn(3)) {
  index= apply(mu_ranks, 1, function(row) {
    all(row == p)
  })
  prob_ranks[[paste(p, collapse = ",")]] = length(mu_ranks[index, 
1])/10000
}

prob_ranks[["1,2,3"]]
prob_ranks[["1,3,2"]]
prob_ranks[["2,1,3"]]
prob_ranks[["3,1,2"]]
prob_ranks[["2,3,1"]]
prob_ranks[["3,2,1"]]
```

Results:

```
> prob_ranks[["1,2,3"]]
[1] 0.0066

> prob_ranks[["1,3,2"]]
```

```
[1] 0.0042

> prob_ranks[["2,1,3"]]
[1] 0.0846

> prob_ranks[["3,1,2"]]
[1] 0.6639

> prob_ranks[["2,3,1"]]
[1] 0.0154

> prob_ranks[["3,2,1"]]
[1] 0.2253
```

Problem 5, Part c

R Code

```
#Part c

#Posterior predictive distribution
predict = matrix(0, 10000, 3, dimnames = list(NULL,
c("school1","school2", "school3")))
for (i in c(1, 2, 3)) {
  predict[, i]= rnorm(10000, mun[i], sqrt(sigma[,i]*((kn[i]+1)/kn[i])))
}

#Computing ranks and probabilities

pred_rank= t(apply(predict, 1, rank))
pred_probrank = list()
for (p in permn(3)) {
  index = apply(pred_rank, 1, function(row) {all(row == p)
  })
  pred_probrank[[paste(p, collapse = ",")]]= length(pred_rank[index,
1])/10000
}

pred_probrank[["1,2,3"]]
pred_probrank[["1,3,2"]]
pred_probrank[["3,1,2"]]
pred_probrank[["2,1,3"]]
pred_probrank[["2,3,1"]]
pred_probrank[["3,2,1"]]
```

Results:

```
> pred_probrank[["1,2,3"]]
[1] 0.1092

> pred_probrank[["1,3,2"]]
[1] 0.1041

> pred_probrank[["3,1,2"]]
[1] 0.2699

> pred_probrank[["2,1,3"]]
[1] 0.1828
```

```
> pred_probrank[["2,3,1"]]
[1] 0.1402

> pred_probrank[["3,2,1"]]
[1] 0.1938
```

Problem 5, part d

R Code and Results

```
> #Part d
> prob_ranks[["2,3,1"]]+prob_ranks[["3,2,1"]]
[1] 0.2407

> pred_probrank[["2,3,1"]]+pred_probrank[["3,2,1"]]
[1] 0.334
```