Implementing Metropolis Algorithm and Gibbs Sampler

Rory Michelen

October 6, 2019

Question 1

Part A

To implement the metropolis algorithm for the given question, we must first calculate $f(Y \mid \theta)\pi(\theta)$ where Y is the set of y_i for i on 1 to n. We are given:

(1) $f(y_i \mid \theta) \; \alpha \; \tau^{\frac{1}{2}} exp\{-\frac{\tau}{2}(y_i - \theta)^2\}$

(2) $\pi(\theta) \; \alpha \; cos^2(\frac{\pi\theta}{2m})$

First, we need to derive a likelihood function when there are multiple oberservations for Y.

(3) $f(Y \mid \theta) = \prod_{i=1}^{n} f(y_i \mid \theta)$

(4) $f(Y \mid \theta) \; \alpha \; \tau^{\frac{n}{2}} exp\{-\frac{\tau}{2}\sum_{i=1}^{n}(y_i - \theta)^2\}$

Using (2) and (4), we can create a formula for $f(Y \mid \theta)\pi(\theta)$ which will be used in the metrpolis ratio, rho.

(5) $f(Y \mid \theta)\pi(\theta) \; \alpha \; \tau^{\frac{n}{2}} exp\{-\frac{\tau}{2}\sum_{i=1}^{n}(y_i - \theta)^2\} cos^2(\frac{\pi\theta}{2m})$

The code below generates 10,500 samples from the posterior and plots a histogram of the results. Additionall, a plot is provided showing the results using the first x samples where x varies from 500 to 10,000 in increments of 500

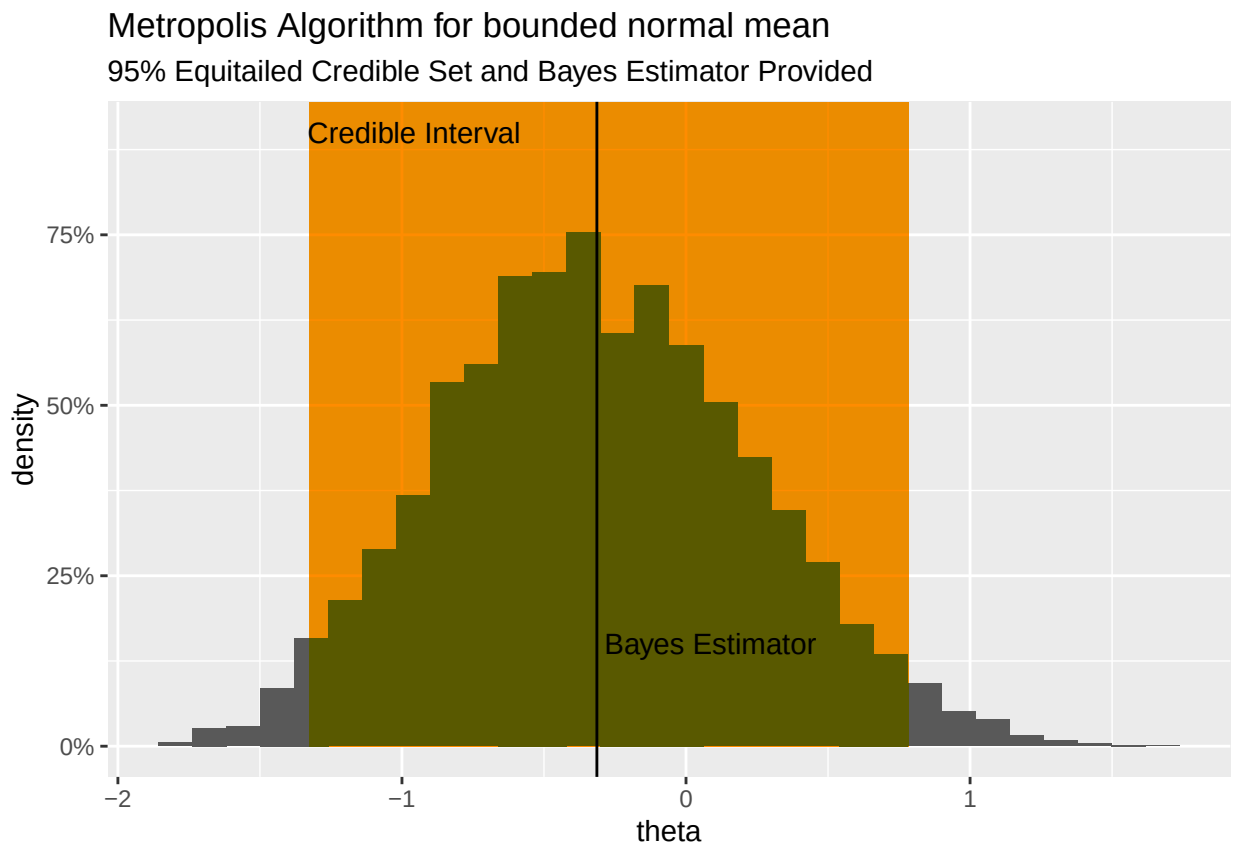
```
# Instantiating the metropolis algorithm
set.seed(1)

posterior.prob<-function(tau,Y,theta,m){
  sum.sq.y<-sum((Y-theta)^2)
  n.y=length(Y)
  return(tau^(n.y*0.5)*exp(-0.5*tau*sum.sq.y)*cos(pi*theta/(2*m))^2)
}

metropolis.iteration<-function(tau,Y,theta,m){
  theta.prop<-runif(1,-m,m)
  rho<-posterior.prob(tau,Y,theta.prop,m)/posterior.prob(tau,Y,theta,m)
  rand<-runif(1)
  if(rho>rand){
    theta=theta.prop
  }
  return(data.frame(theta,rho,rand))
}

metropolis<-function(tau,Y,m,sample.size,sample.burn){
  output<-data.frame()
  theta<-0
  for(i in 1:(sample.size+sample.burn)){
    iteration.output<-cbind(i,sample.type=ifelse(i<=sample.burn,'burn','sample'),
                            metropolis.iteration(tau,Y,theta,m))
    output<-rbind(output,iteration.output)
```

```
    theta<-output[nrow(output),]$theta
  }
  return(output)
}
```

```
# Plugging in the values provided

params<-data.frame(tau=1/4,m=2,sample.size=10000,sample.burn=500)
Y<-c(-2,-3, 4,-7,0,4)


output<-with(metropolis(tau,Y,m,sample.size,sample.burn),data=params)

credible.set<-output%>%
  dplyr::filter(sample.type!='burn')%>%
  summarise(bayes.estimate=mean(theta),lb=quantile(theta,0.025),ub=quantile(theta,0.975))
```

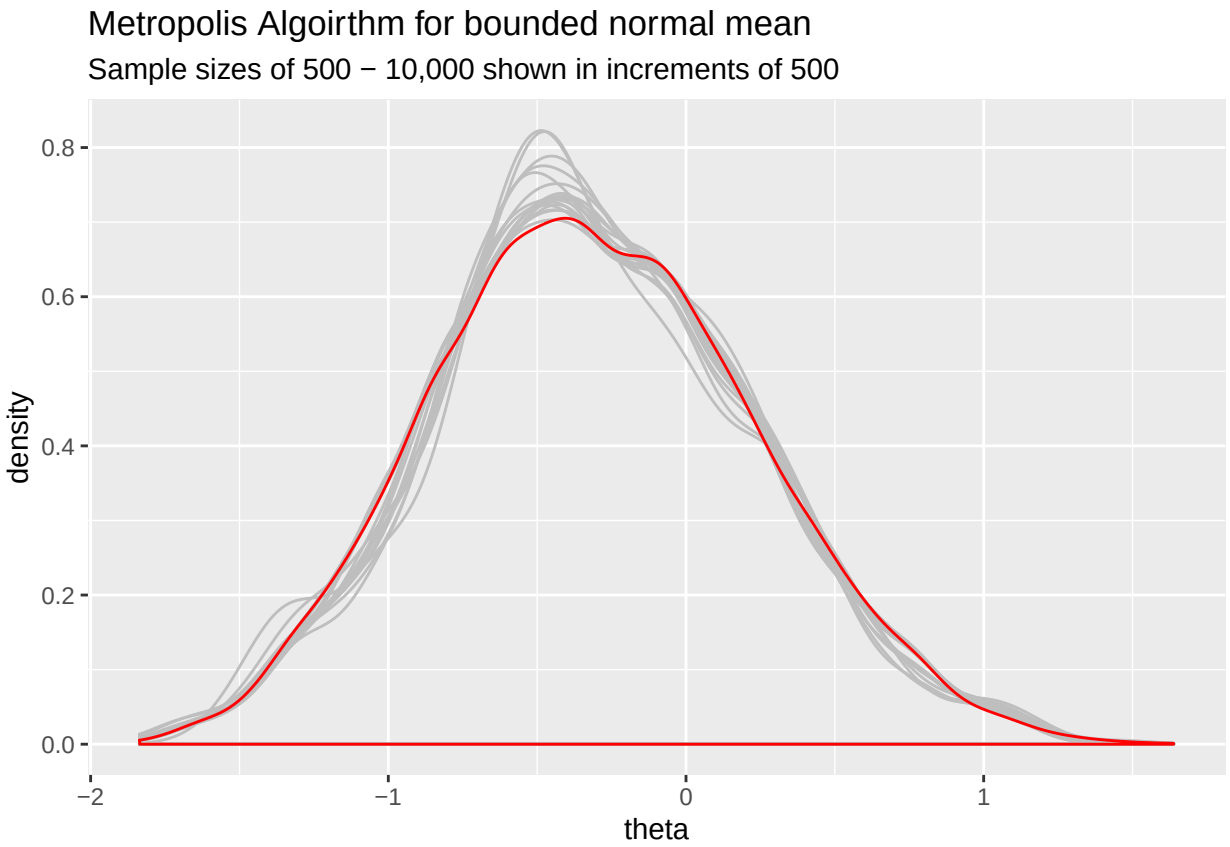
```
## Warning: package 'bindrcpp' was built under R version 3.5.1
```

```
output%>%
  dplyr::filter(sample.type!='burn')%>%
  ggplot(aes(x=theta))+geom_histogram(aes(y=..density..))+
  scale_y_continuous(labels = scales::percent)+
  geom_rect(aes(xmin=credible.set$lb,xmax=credible.set$ub,ymin=0,ymax=Inf),
            fill='darkorange',alpha=0.005)+
  annotate("text",label='Credible Interval',x=credible.set$lb+0.37,y=0.9)+
  geom_vline(xintercept = credible.set$bayes.estimate)+
  annotate("text",label='Bayes Estimator',x=credible.set$bayes.estimate+0.40,y=0.15)+
  labs(title='Metropolis Algorithm for bounded normal mean',
       subtitle='95% Equitailed Credible Set and Bayes Estimator Provided')
```




```
data.frame(i=seq(500,10000,500),sample.type='sample')%>%
  inner_join(.,output,by=c('sample.type'))%>%
  dplyr::filter(i.x>i.y)%>%
  mutate(color2=ifelse(i.x<10000,'grey','red'))%>%
  ggplot(aes(x=theta,col=color2))+geom_density(aes(group=i.x))+
  scale_color_identity()+
  labs(title='Metropolis Algoirthm for bounded normal mean',
       subtitle='Sample sizes of 500 - 10,000 shown in increments of 500')
```

```
## Warning: Column `sample.type` joining factors with different levels,
## coercing to character vector
```



Part B

The code above was used to find the 95% credible set of [-1.33,0.782] and bayes estimator of -0.314 (presumably the true value of the parameter is pi).

Question 2

Part A

We are given a lot of information to preface this problem. It is listed below, with some slight modifications to the notation.

(1) $f(y_{j,i} \mid \theta_j, \tau_j) \; \alpha \; \tau_j^{\frac{1}{2}} exp\{-\frac{\tau_j}{2}(y_{j,i} - \theta)^2\}$

Where j=1 corresponds to the high protein diet and j=2 correponds to the low protein diet. And i is the index of the observations for y.

(2) $\theta_j \sim Normal(\theta_{j,0}, \frac{1}{\tau_{j,0}})$

(3) $\tau_j \sim Gamma(a_j, b_j)$

Since there are multiple observations for Y_1 and Y_2, let's derive a likelihood function that is valid for multiple observations.

(4) $f(Y_j \mid \theta_j, \tau_j) = \prod_{i=1}^{n_j} f(y_{j,i} \mid \theta_j, \tau_j)$

(5) $f(Y_j \mid \theta_j, \tau_j) \; \alpha \; \tau_j^{\frac{n_j}{2}} exp\{-\frac{\tau_j}{2} \sum_i^{n_j} (y_{j,i} - \theta_j)^2\}$

Where $n_1 = 12$ and $n_2 = 7$

Finding the joint probability

There are 4 parameters of interest. If they were all dependent on one another, we would have to multiply all four priors and 2 likelikehoods together to get the joint probability. However, since both samples (and therefore both posterior distributions) are independent of one another, this problem can be thought of as two separate Gibbs samplers, one for j=1 and another for j=2.

(6) $f(\theta_j, \tau_j, Y_j) \; \alpha \; \tau_j^{\frac{n_j}{2}} exp\{-\frac{\tau_j}{2}\sum_i^{n_j}(y_{j,i}-\theta_j)^2\}\tau_{j,0}^{\frac{1}{2}} exp\{-\frac{\tau_{j,0}}{2}(\theta_j-\theta_{j,0})^2\}\tau_j^{a_j-1} exp\{-b_j\tau_j\}$

Finding the conditional probabilities

Let's start with conitionals for τ_1 and τ_2 (because it's easier).

(7) $f(\tau_j \mid \theta_j, Y_j) \; \alpha \; \tau_j^{\frac{n}{2}}\tau_j^{a_j-1} exp\{-\frac{\tau_j}{2}\sum_{i=1}^{n_j}(y_{j,i}-\theta_j)^2\} exp\{-b_j\tau_j\}$

(8) $f(\tau_j \mid \theta_j, Y_j) \; \alpha \; \tau_j^{\frac{n_j}{2}+a_j-1} exp\{-\tau_j(b+\frac{1}{2}\sum_{i=1}^{n_j}(y_{j,i}-\theta_j)^2\}$

From (8), it is clear that the posterior for τ_j follows a $Gamma(\frac{n_j}{2}+a_j, b_j+\frac{1}{2}\sum_{i=1}^{n_j}(y_{j,i}-\theta_j)^2)$ distribution

Now, for the fun part: θ_1 and θ_2. Using (6) we can derive the conditional probability:

(9) $f(\theta_j \mid \tau_j, Y_j) \; \alpha \; exp\{-\frac{\tau_j}{2}\sum_{i=1}^{n_j}(y_{j,i}-\theta_j)^2\} exp\{-\frac{\tau_{j,0}}{2}(\theta-\theta_{j,0}^2)\}$

Using the guidance in the Gibbs Handout pdf, we know that this is equivalent to

(10) $f(\theta_j \mid \tau_j, Y_j) \; \alpha \; exp\{-\frac{1}{2}(\tau_{j,0}+n_j\tau_j)(\theta_j-\frac{\tau_j\sum_{i=1}^{n_j}y_{j,i}+\tau_{j,0}\theta_{j,0}}{\tau_{j,0}+n_j\tau_j})^2\}$

From (10) we see that θ_j follows a distribution of the form: $Normal(\frac{\tau_j\sum_{i=1}^{n_j}y_{j,i}+\tau_{j,0}\theta_{j,0}}{\tau_{j,0}+n_j\tau_j}, \tau_{j,0}+n_j\tau_j)$ where the second parameter represets precision (as opposed to variance).

Now with the fun part out of the way, let's code!

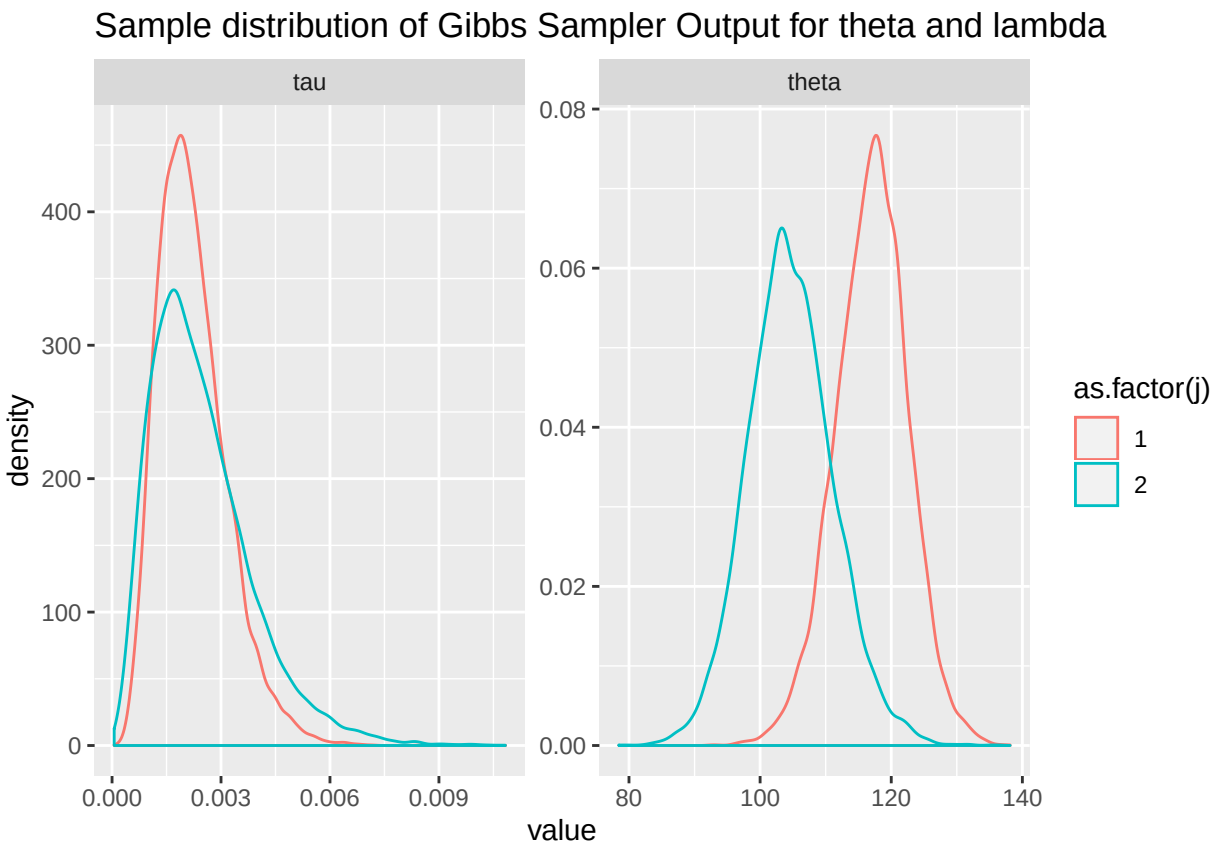
```
# Instantiating the gibbs sampler algorithm
tau.sample<-function(a,b,Y,theta){
  n<-length(Y)
  alpha<-a+(n/2)
  beta<-b+(0.5*sum((Y-theta)^2))
  return(rgamma(1,alpha,rate=beta))
}

theta.sample<-function(theta0,tau0,Y,tau){
  n<-length(Y)
  mu<-((tau*sum(Y))+(tau0*theta0))/(tau0+(n*tau))
  precision<-tau0+(n*tau)
  return(rnorm(1,mean=mu,sd=(1/precision)^0.5))
}

# I've arbitrarily chosen to sample theta first in each iteration
gibbs.iteration<-function(a,b,theta0,tau0,Y,theta,tau){
  theta.new<-theta.sample(theta0,tau0,Y,tau)
  tau.new<-tau.sample(a,b,Y,theta.new)
  return(data.frame(theta=theta.new,tau=tau.new))
}
```

```
# Plugging in the provided values
params<-rbind(data.frame(j=1,a=0.01,b=4,tau0=1/100,theta0=110,n.sample=10500),
              data.frame(j=2,a=0.01,b=4,tau0=1/100,theta0=110,n.sample=10500))
```

```
Y<-list(c(134,146,104,119,124,161,107,83,113,129,97,123),
        c(70,118,101,85,107,132,94))

output<-data.frame()
# Initialize our parameters
theta=1
tau=1
for(j in 1:nrow(params)){
  for(i in 1:params[j,]$n.sample){
    new.row<-data.frame(with(cbind(j,i,gibbs.iteration(a,b,theta0,tau0,Y[[j]],theta,tau)),data=params[j
    theta<-new.row$theta
    tau<-new.row$tau
    output<-rbind(output,new.row)
  }
}
```

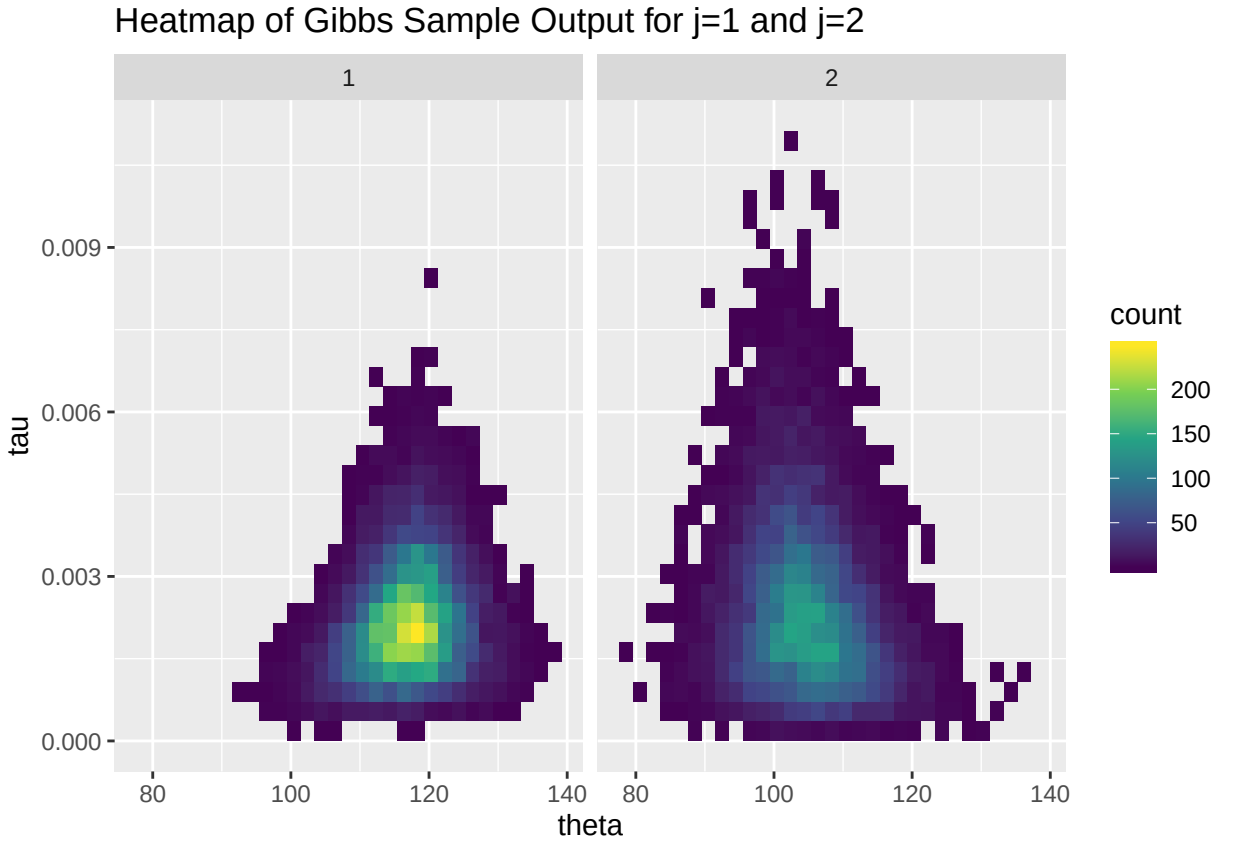
The plot below shows the distribution of samples for all 4 parameters. As expected, $theta_1$ is shifted slightly right since this data set had a higher average weight.

```
output%>%
  dplyr::filter(i>500)%>%
  gather(key='parameter',value='value',theta,tau)%>%
  ggplot(aes(x=value,col=as.factor(j)))+
    geom_density(aes(group=j))+
    facet_wrap(~parameter,scales = 'free')+
    labs(title='Sample distribution of Gibbs Sampler Output for theta and lambda')
```



Here is an alternative view showing the 2d distribution for each participant group.

```
output%>%
  dplyr::filter(i>500)%>%
  ggplot(aes(x=theta,y=tau))+
    geom_bin2d()+facet_wrap(~j)+
    scale_fill_continuous(type='viridis')+
    labs(title='Heatmap of Gibbs Sample Output for j=1 and j=2')
```



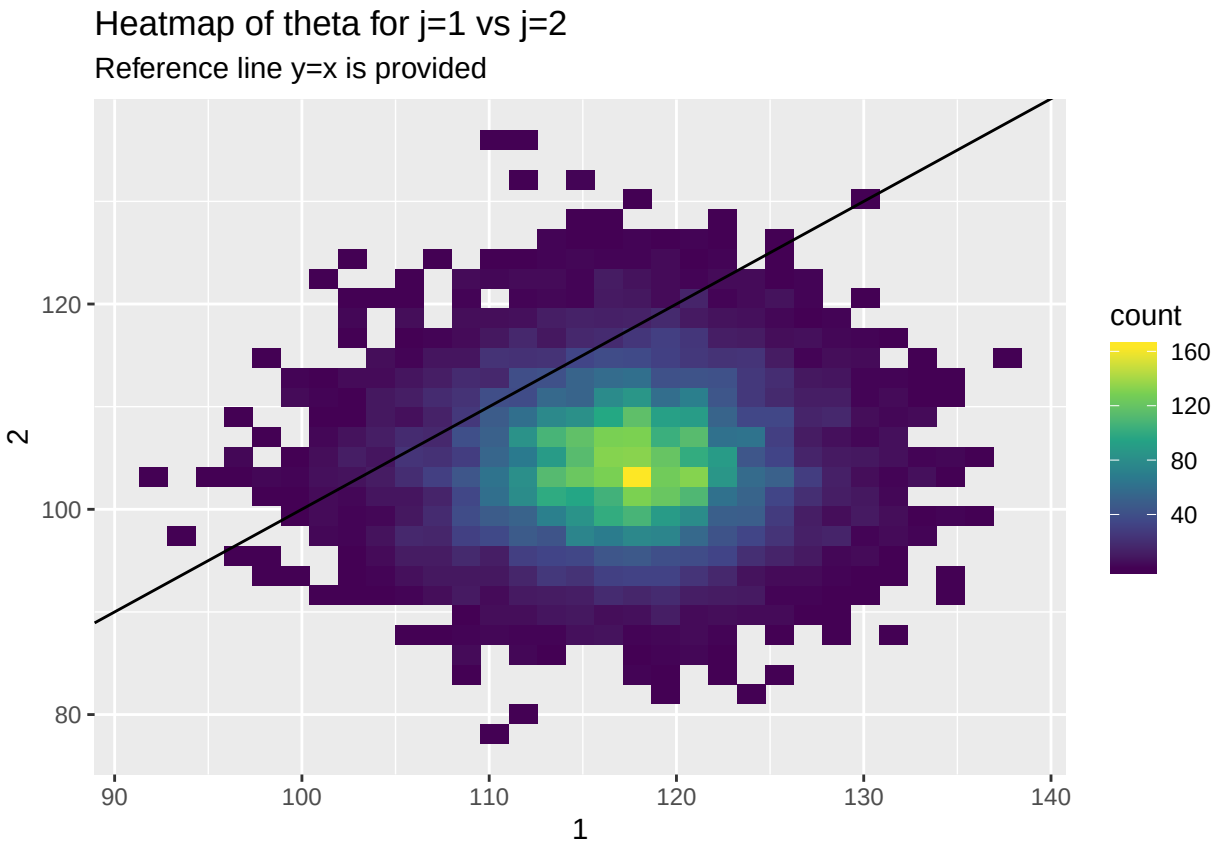
Part B

The code below demonstrates that in 92.5% of the 10,000 samples, $\theta 1 > \theta_1$. Therefore it should come as no surprise that our 95% credible interval of [-5.33,29.08] contains 0.

```
output.diff<-output%>%
  dplyr::filter(i>500)%>%
  select(-tau)%>%
  spread(key=j,value=theta)%>%
  mutate(sample.diff=`1`-`2`)

output.diff%>%
  mutate(h0.accept=ifelse(sample.diff>0,1,0))%>%
  group_by(h0.accept)%>%
  summarise(num.samples=n())%>%
  ungroup()%>%
  mutate(output=num.samples/sum(num.samples))%>%
  kable()
```

h0.accept	num.samples	output
0	746	0.0746
1	9254	0.9254

```
credible.set<-output.diff%>%
  summarise(lb=quantile(sample.diff,0.025),
            ub=quantile(sample.diff,0.975))

output.diff%>%
  ggplot(aes(x=`1`,y=`2`))+
  geom_bin2d()+
  scale_fill_continuous(type='viridis')+
  geom_abline(slope=1)+
  labs(title='Heatmap of theta for j=1 vs j=2',subtitle='Reference line y=x is provided')
```



While intuition may lead us to believe that this sample difference would be more significant. Our prior distributions had a fairly low variance (high precision) and therefore was not overwhelmed by the data.