Normal Mixture Model Gibbs Sampler

Jordan Aron

October 15, 2017

$z_i \sim Bern(\frac{a}{a+b})$ where $a = (p * f(x_i|\mu_1, \sigma_1))$ and $b = ((1-p) * f(x_i|\mu_2, \sigma_2))$

$p \sim beta(\sum z, n - \sum z)$

$\sigma_1 \sim inverse-gamma(\frac{\kappa_{n_1}}{2}, \frac{\kappa_{n_1}}{2}\sigma^2_{n_1})$ where $\kappa_{n_1} = \kappa_{0_1} + n_1$, κ_{0_1} is the prior sample size for population 1, and n_1 is the sample size of the data for population 1

$\sigma_2 \sim inverse-gamma(\frac{\kappa_{n_2}}{2}, \frac{\kappa_{n_2}}{2}\sigma^2_{n_2})$ where $\kappa_{n_2} = \kappa_{0_2} + n_2$, κ_{0_2} is the prior sample size for population 2, and n_2 is the sample size of the data for population 2

$\theta_1 \sim normal(\mu_{n_1}, \frac{\sigma^2_1}{\kappa_{n_1}})$ where $\mu_{n_1} = \frac{\kappa_{0_1}}{\kappa_{n_1}}\mu_{0_1} + \frac{n_1}{\kappa_{n_1}}\bar{y}_1$, μ_{0_1} is the prior mean for population 1, and $\bar{y}_1$ is the mean of the data for population 1

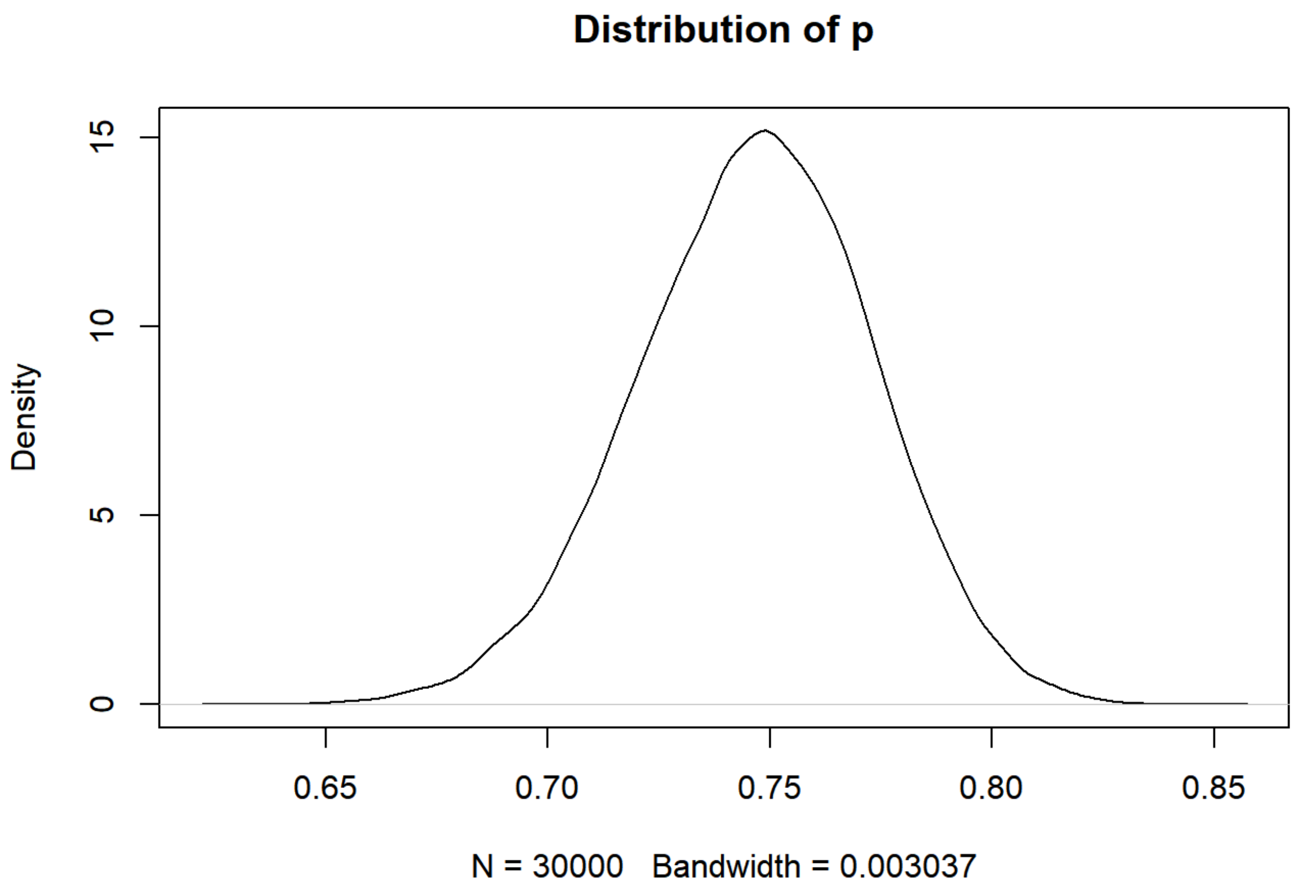
$\theta_2 \sim normal(\mu_{n_2}, \frac{\sigma^2_2}{\kappa_{n_2}})$ where $\mu_{n_2} = \frac{\kappa_{0_2}}{\kappa_{n_2}}\mu_{0_2} + \frac{n_2}{\kappa_{n_2}}\bar{y}_2$, μ_{0_2} is the prior mean for population 1, and $\bar{y}_2$ is the mean of the data for population 1

```r
#initialize values
#balance of obs1 vs. obs2 determins mixing of the normal distribution
obs1 <- 300
obs2 <- 100
#total amount of observations
n <- obs1+obs2

#initialize values for population 1
pop1.trueMean <- 0
pop1.trueSD <- 10

#initialize values for population 2
pop2.trueMean <- 40
pop2.trueSD <- 10

#initialize values for prior for population 1
pop1.prior.sampleSize <- 30
pop1.prior.trueMean <- 5
pop1.prior.trueSD <- 20

#initialize values for prior for population 2
pop2.prior.sampleSize <- 20
pop2.prior.trueMean <- 35
pop2.prior.trueSD <- 20


#Determines number of iterations for gibbs sampler
trials <- 30000
```

```r
#####Don't Alter Anything Below#####

#creates data, then binds together to create normal mixture model
samp1 <- rnorm(obs1,pop1.trueMean,pop1.trueSD)
samp2 <- rnorm(obs2,pop2.trueMean,pop2.trueSD)
data <- c(samp1,samp2)

#creates priors
prior.pop1 <- rnorm(pop1.prior.sampleSize, pop1.prior.trueMean, pop1.prior.trueSD)
prior.pop2 <- rnorm(pop2.prior.sampleSize, pop2.prior.trueMean, pop2.prior.trueSD)

#creates matrix where gibbs will be done
simlist <- matrix(rep(0,5*trials), ncol = 5)
colnames(simlist) <- c("p", "SD of Pop1", "SD of Pop2", "Mean of Pop1", "Mean of Po
p2")

#saves value for later usage
pop1.prior.mean <- mean(prior.pop1)
pop1.prior.variance <- var(prior.pop1)
pop2.prior.mean <- mean(prior.pop2)
pop2.prior.variance <- var(prior.pop2)

#initialize first entry in gibbs so it will run correctly
simlist[1,1] <- obs1/n
simlist[1,2] <- sqrt(pop1.prior.variance)
simlist[1,3] <- sqrt(pop2.prior.variance)
simlist[1,4] <- pop1.prior.mean
simlist[1,5] <- pop2.prior.mean


#The actual gibbs sampler
#starts at 2 since we initialized values for when i=1
for(i in 2:trials){

  #max size will be z
  z <- numeric(n)
  a <- numeric(n)
  b <- numeric(n)



  j = 1
  #calculates a_j and b_j for each data_j
  #then calculates z_i's
  for (j in 1:n){
    a[j] = simlist[i-1,1]*dnorm(data[j],simlist[i-1,4],simlist[i-1,2])
    b[j] = (1-simlist[i-1,1])*dnorm(data[j],simlist[i-1,5],simlist[i-1,3])
    z[j] = rbinom(1,1,(a[j]/(a[j]+b[j])))
  }

  #calculates p using beta distrivution
  simlist[i,1] <- rbeta(1,sum(z), n-sum(z))


  pop1 <- numeric(n)
```

```
  pop2 <- numeric(n)

  j = 1
  #dividing data into pop1 and pop2 as best as we can at the moment
  for (j in 1:n){
    pop1[j] = z[j]*data[j]
    pop2[j] = ((1-z[j])*-1)*data[j]
    pop2[j] = -1*pop2[j]
  }

  #gets rid of zeros in list
  pop1 <- pop1[pop1!=0]
  pop2 <- pop2[pop2!=0]

  #calculates length, used later
  n1 <- length(pop1)
  n2 <- length(pop2)

  #calculates SD for population 1
  #first calculated precision with gamma, then variance, finally SD
  pop1.Vn <- pop1.prior.sampleSize + n1
  pop1.sigmaN <- ((pop1.prior.sampleSize*pop1.prior.variance) + ((n1 - 1)*var(pop1
)) + (((pop1.prior.sampleSize*n1)/pop1.Vn)*(mean(pop1) - pop1.prior.mean))) / pop1
.Vn
  prec1 <- rgamma(1,pop1.Vn/2,(pop1.Vn * pop1.sigmaN)/2)
  variance1 = 1/prec1
  simlist[i,2] <- sqrt(variance1)

  #similar to above except uses population 2 parameters
  pop2.Vn <- pop2.prior.sampleSize + n2
  pop2.sigmaN <- ((pop2.prior.sampleSize*pop2.prior.variance) + ((n2 - 1)*var(pop2
)) + (((pop2.prior.sampleSize*n2)/pop2.Vn)*(mean(pop2) - pop2.prior.mean))) / pop2
.Vn
  prec2 <- rgamma(1,pop2.Vn/2,(pop2.Vn * pop2.sigmaN)/2)
  variance2 = 1/prec2
  simlist[i,3] <- sqrt(variance2)

  #calculates mean for population 1
  #uses normal distribution
  pop1.denom <- (simlist[i-1,2] + (sum(z)*var(pop1)))
  theta1 <- ((simlist[i-1,2] / pop1.denom)*simlist[i-1,4]) + (((sum(z)*var(pop1))/
pop1.denom) * mean(pop1))
  simlist[i,4] <- rnorm(1,theta1, simlist[i,2]/pop1.Vn)

  #same as above except uses population 2 parameters
  pop2.denom <- simlist[i-1,3] + ((n-sum(z))*var(pop2))
  theta2 <- ((simlist[i-1,3] / pop2.denom)*simlist[i-1,5]) + ((((n-sum(z))*var(pop
2))/pop2.denom) * mean(pop2))
  simlist[i,5] <- rnorm(1,theta2,simlist[i,3]/pop2.Vn)



}
```

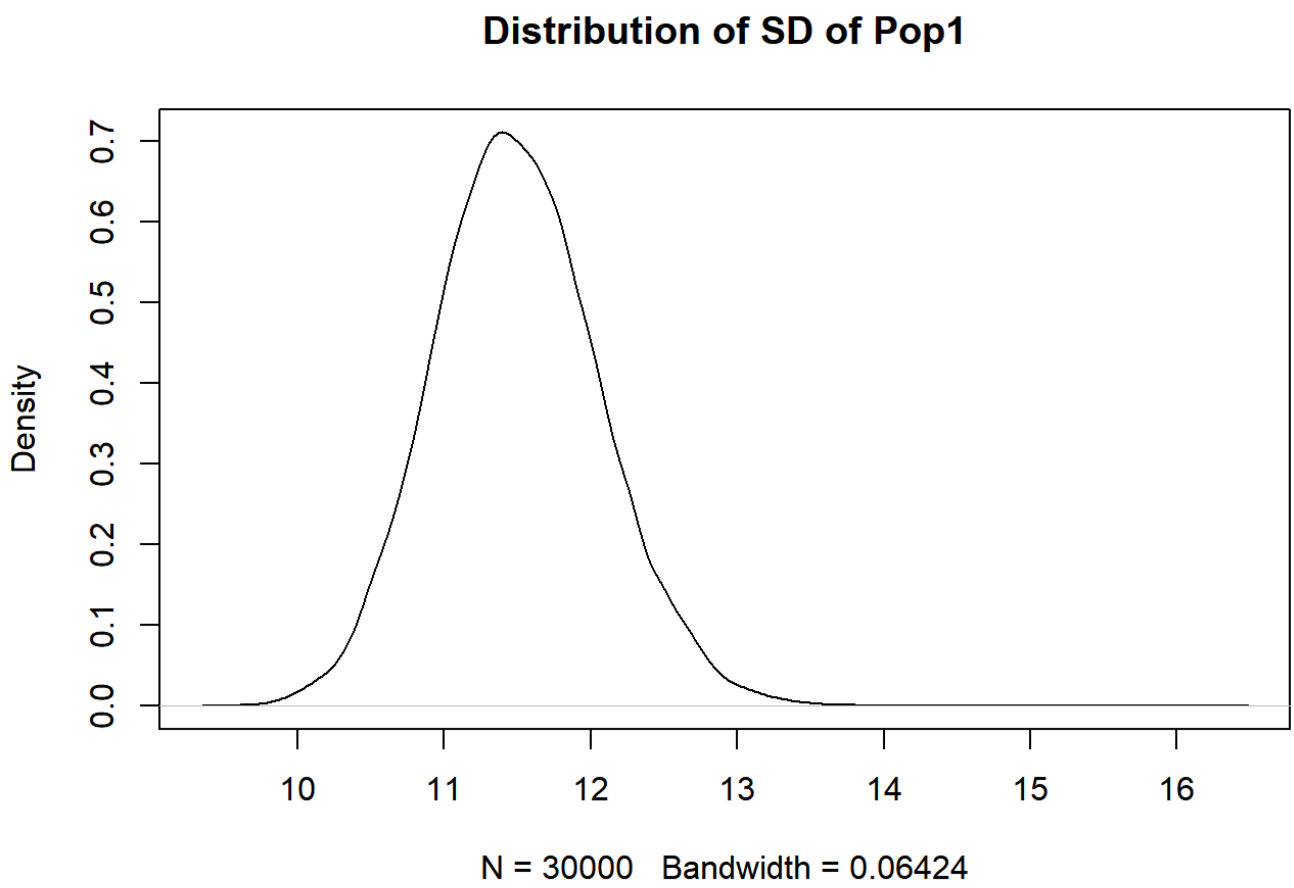
```r
#plots and gives means of parameters
plot(density(simlist[,1]), main = "Distribution of p")
```




```r
mean(simlist[,1])
```

```
## [1] 0.7464147
```

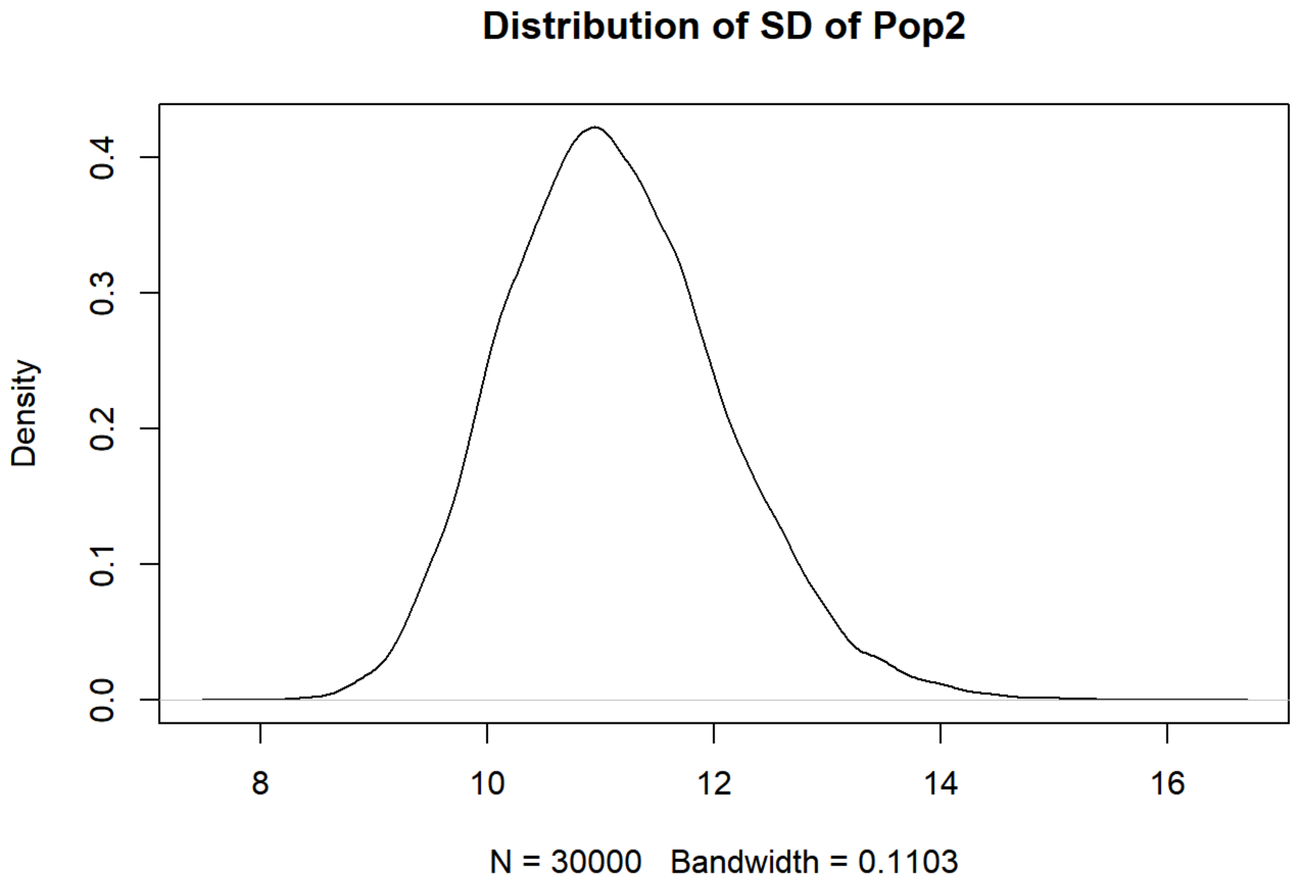
```r
plot(density(simlist[,2]),main = "Distribution of SD of Pop1")
```




```r
mean(simlist[,2])
```

```
## [1] 11.49618
```

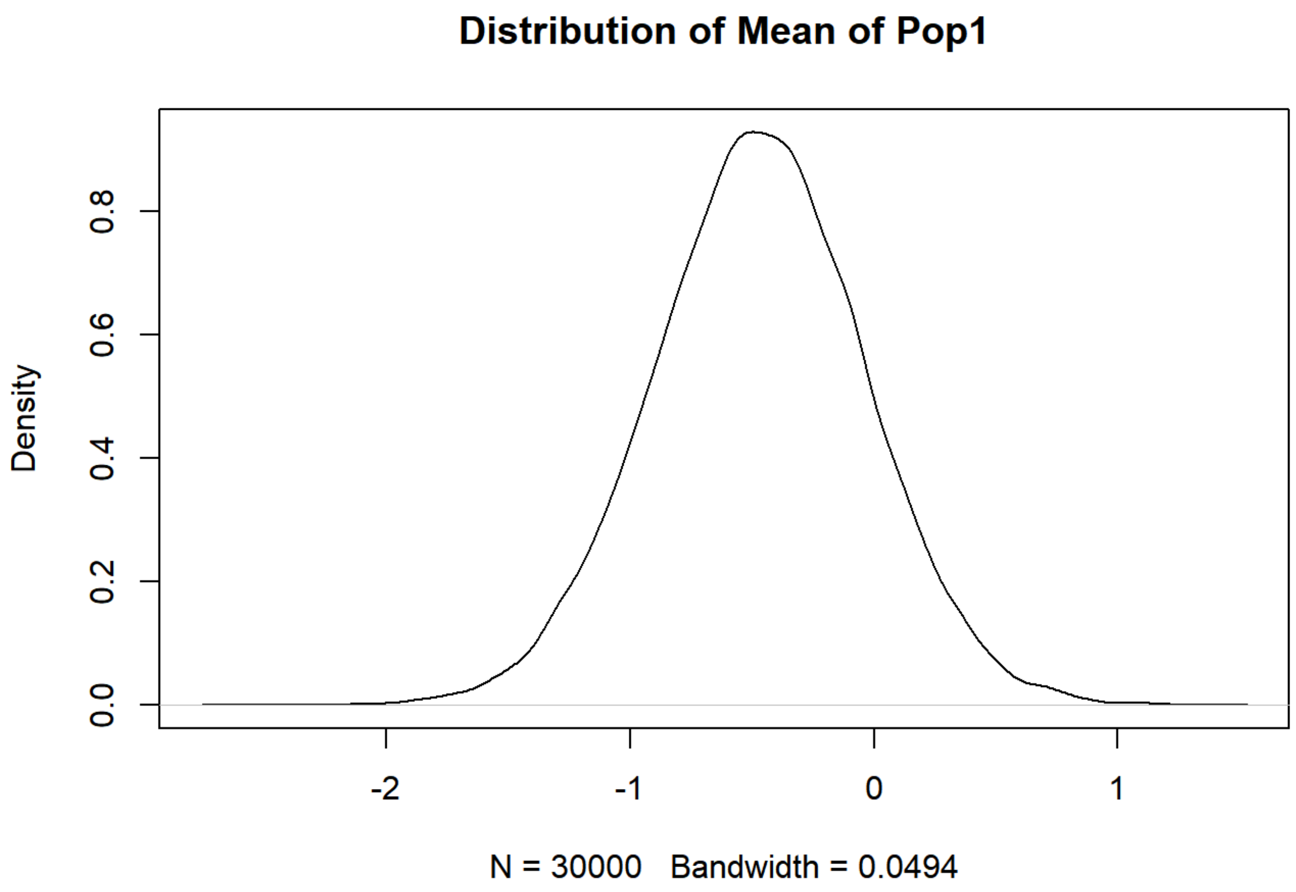
```r
plot(density(simlist[,3]),main = "Distribution of SD of Pop2")
```




```
mean(simlist[,3])
```

```
## [1] 11.1242
```

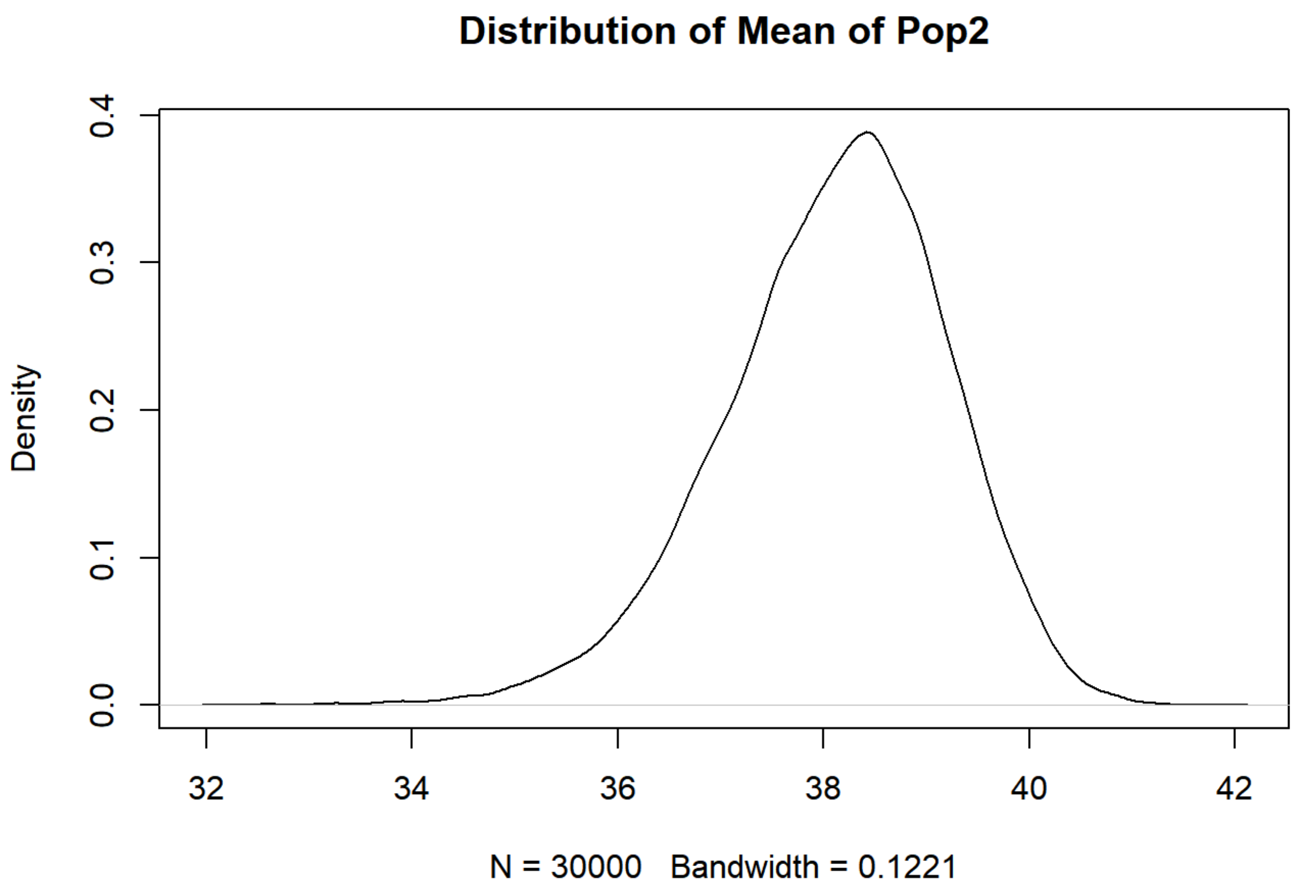
```
plot(density(simlist[,4]),main = "Distribution of Mean of Pop1")
```




```
mean(simlist[,4])
```

```
## [1] -0.4722706
```

```
plot(density(simlist[,5]),main = "Distribution of Mean of Pop2")
```




```
mean(simlist[,5])
```

```
## [1] 38.0994
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js