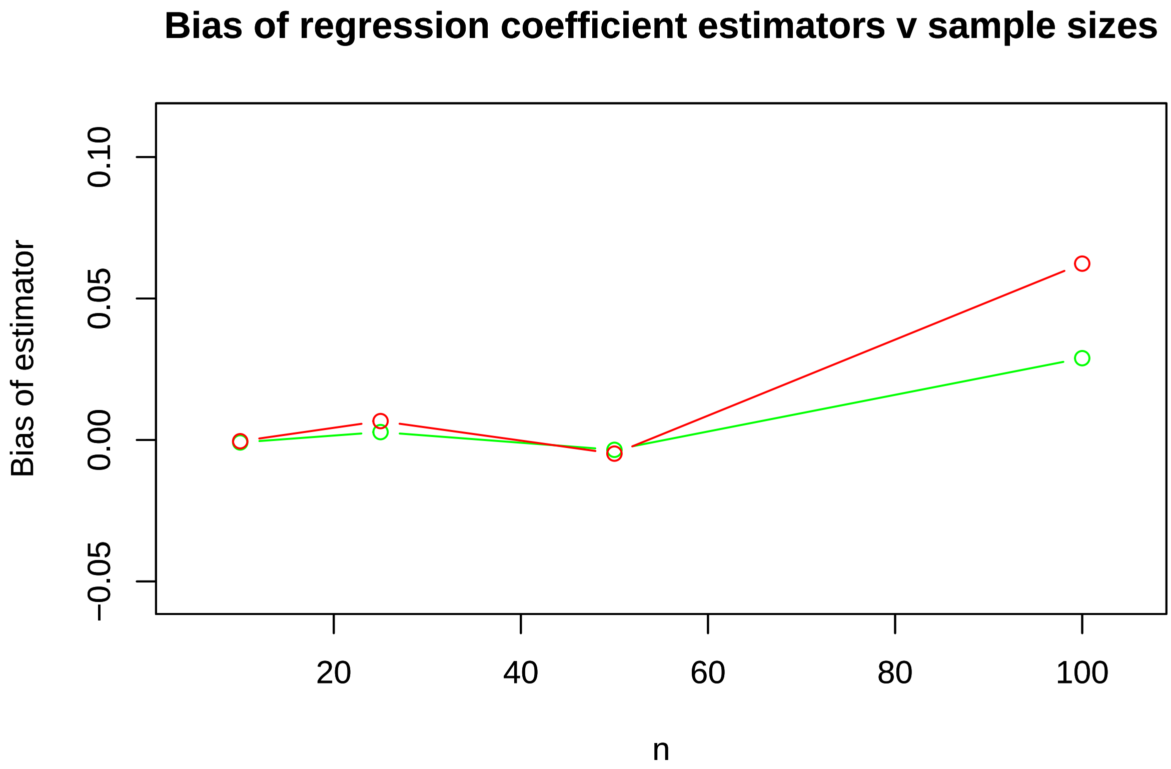
HW 2

Ian Douglas

2/12/2019

Question 1

```
### define constants
set.seed(142)
gammas <- c(0,.5,1,2)
slopes <- c(0,.5,1,2)
Beta0 <- 0
n <- c(10,25,50,100)
R <- 10000

### Define function
regSim <- function(size, intercept, slope, gamma, replications) {
  ### Create empty matrix for storing output
  output <- matrix(0, replications, 3)
  ### sample x (size n) from N(0,1)
  x <- rnorm(n = size, 0, 1)
  ### generate y using linear model with non-constant variance
  for (i in 1:replications) {
    y <- intercept + slope*x + rnorm(size, 0, exp(gamma*x))
    ### estimate Beta parameters and fill output matrix
    output[i,] <- c(lm(y ~ x)$coef, size)
  }

  output

}

### Run 4 times (first time: n = 10, Beta1 = 0, and gamma = 0, and so on)
out1 <- regSim(size = n[1], intercept = Beta0, slope = slopes[1],
               gamma = gammas[1], replications = R)
out2 <- regSim(size = n[2], intercept = Beta0, slope = slopes[2],
               gamma = gammas[2], replications = R)
out3 <- regSim(size = n[3], intercept = Beta0, slope = slopes[3],
               gamma = gammas[3], replications = R)
out4 <- regSim(size = n[4], intercept = Beta0, slope = slopes[4],
               gamma = gammas[4], replications = R)
b0.bias1 <- mean(out1[,1]) - Beta0
b0.bias2 <- mean(out2[,1]) - Beta0
b0.bias3 <- mean(out3[,1]) - Beta0
b0.bias4 <- mean(out4[,1]) - Beta0
b1.bias1 <- mean(out1[,2]) - slopes[1]
b1.bias2 <- mean(out2[,2]) - slopes[2]
b1.bias3 <- mean(out3[,2]) - slopes[3]
b1.bias4 <- mean(out4[,2]) - slopes[4]
plot(x = c(n[1],n[2],n[3],n[4]),
           y=c(b0.bias1,b0.bias2,b0.bias3,b0.bias4), col = "green",
     xlim=c(5,105), ylim=c(min(c(b1.bias1,b1.bias2,b1.bias3,b1.bias4))-.05,max(c(b1.bias1,b1.bias2,b1.bi
     type = "b",
     main = "Bias of regression coefficient estimators v sample sizes",
     xlab = "n", ylab = "Bias of estimator")
par(new=TRUE)
plot(x = c(n[1],n[2],n[3],n[4]),
           y=c(b1.bias1,b1.bias2,b1.bias3,b1.bias4), col = "red",
     xlim=c(5,105), ylim=c(min(c(b1.bias1,b1.bias2,b1.bias3,b1.bias4))-.05,max(c(b1.bias1,b1.bias2,b1.bi
     type = "b",
     main = "Bias of regression coefficient estimators v sample sizes",
     xlab = "n", ylab = "Bias of estimator")
```



Interpretation:

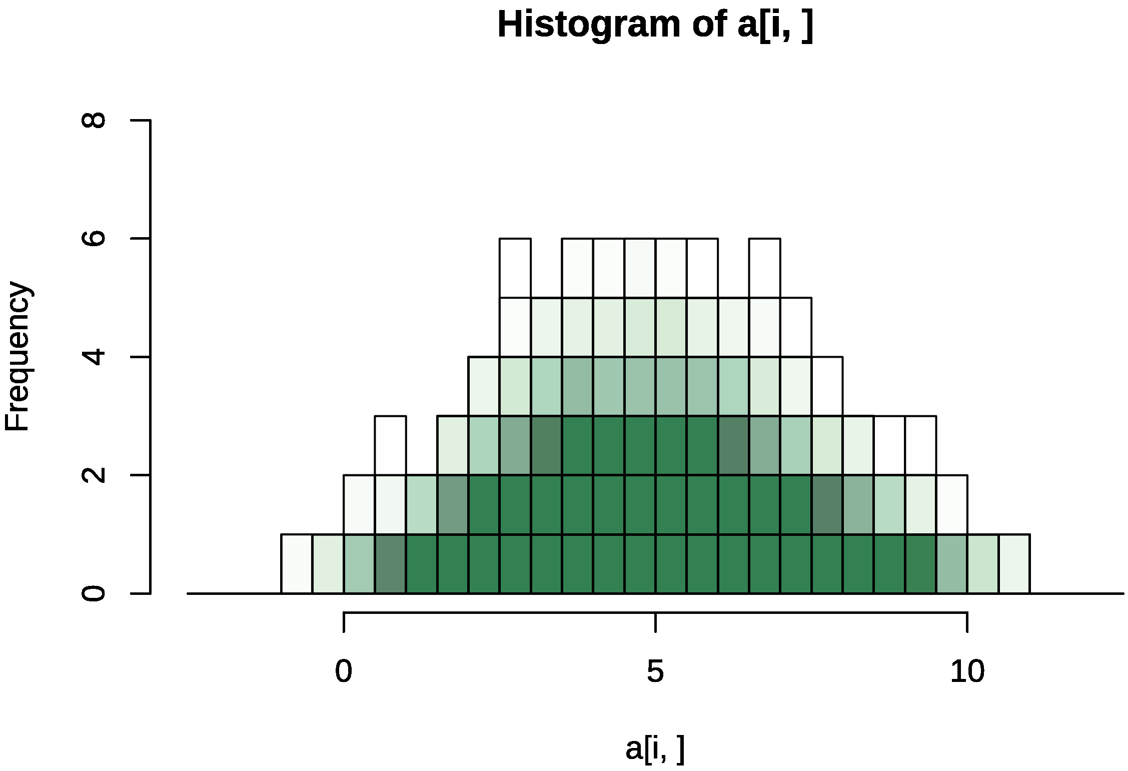
It looks like breaking the assumption of non-constant variance will not noticably affect estimations of Beta0 nor Beta1, as the bias associated with each respective estimate is near zero. However, after sample size 50, it looks like there may be evidence that a cumulative effect may take hold. This would need to be verified by repeating the procedure with sample sizes larger than 100. Still, even at n = 100, the largest bias is smaller than .1

Question 2

Part A)

```
### Generate x and fix constants
x1 <- runif(20,0,1)
x2 <- runif(20,0,2)
B0 <- 1
B1 <- 2
B2 <- 3
```

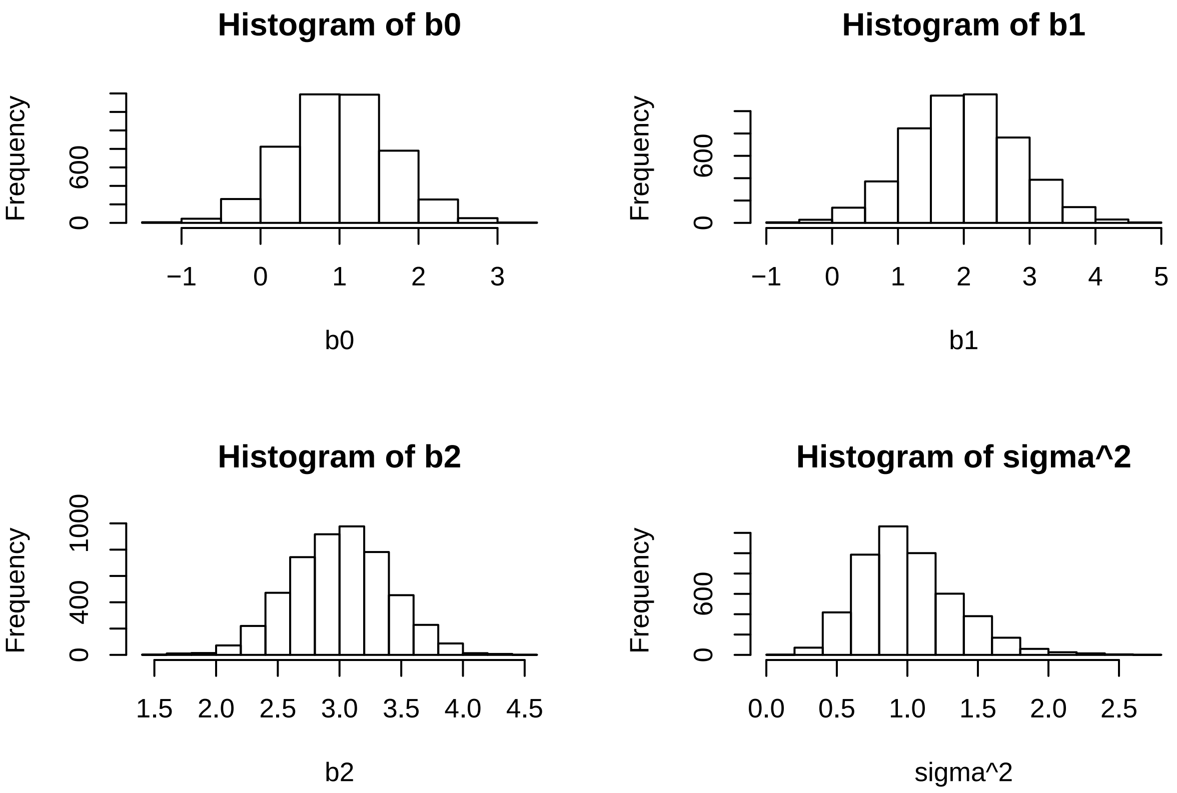
```
r <- 5000
### Write function with established parameters
newRegSim <- function(n,replications) {
  y.output <- matrix(0, replications, n)
  for (i in 1:replications) {
    y.output[i,] <- B0 + B1*x1 + B2*x2 + rnorm(n,0,1)
  }
  y.output
}
### Run with established parameters
a <- newRegSim(n = 20, replications = r)

### Check work with histograms
for (i in sample(1:5000, size = 700, replace = FALSE)) {
  hist(a[i,], breaks = seq(-2.5,12.5,.5),
       col = rgb(.2,.5,.32,
                 alpha = .008),
       ylim=c(0,8)
       )
par(new=TRUE)
}
```



This looks like a composition of two normals, almost bimodal which would make sense for a linear combination of two x variables that have different means.

Part B)

```
### generate regression coefficient estimators
regParams <- function(data) {
  b <- matrix(0,nrow(data),4, dimnames = list(NULL,c("b0", "b1", "b2", "sigma^2")))
  for (i in 1:nrow(data)) {
    b[i,] <- c(coef(lm(a[i,] ~ x1 + x2)), (sigma(lm(a[i,] ~ x1 + x2)))^2)

  }
  b
}
### Run
b <- regParams(a)
### Plot histograms
par(mfrow=c(2,2))
for (i in 1:4) {
  hist(b[,i], main = paste("Histogram of",colnames(b)[i]), xlab = colnames(b)[i])
}
```



Part C)

```
### Compute means and SD for each estimator
c<- apply(b, 2, function(x) c(mean(x),sd(x)))
### Theoretical standard error of b1 and b2:
SE.resid <- mean(sqrt(b[,4]))
R12 <- summary(lm(x1 ~ x2))$r.squared
vx1 <- var(x1)
vx2 <- var(x2)
sb1 <- (SE.resid)/(sqrt((1-R12)*vx1*(20 - 1)))
sb2 <- (SE.resid)/(sqrt((1-R12)*vx2*(20 - 1)))
c
```

```
##              b0        b1        b2   sigma^2
## [1,] 0.9970512 1.9996036 3.0027737 1.0043615
## [2,] 0.6486187 0.8171555 0.3980669 0.3439271
```

```r
print("theoretical standard error of b1:")
```

```
## [1] "theoretical standard error of b1:"
```

```r
sb1
```

```
## [1] 0.7973735
```

```r
print("theoretical standard error of b1:")
```

```
## [1] "theoretical standard error of b1:"
```

```r
sb2
```

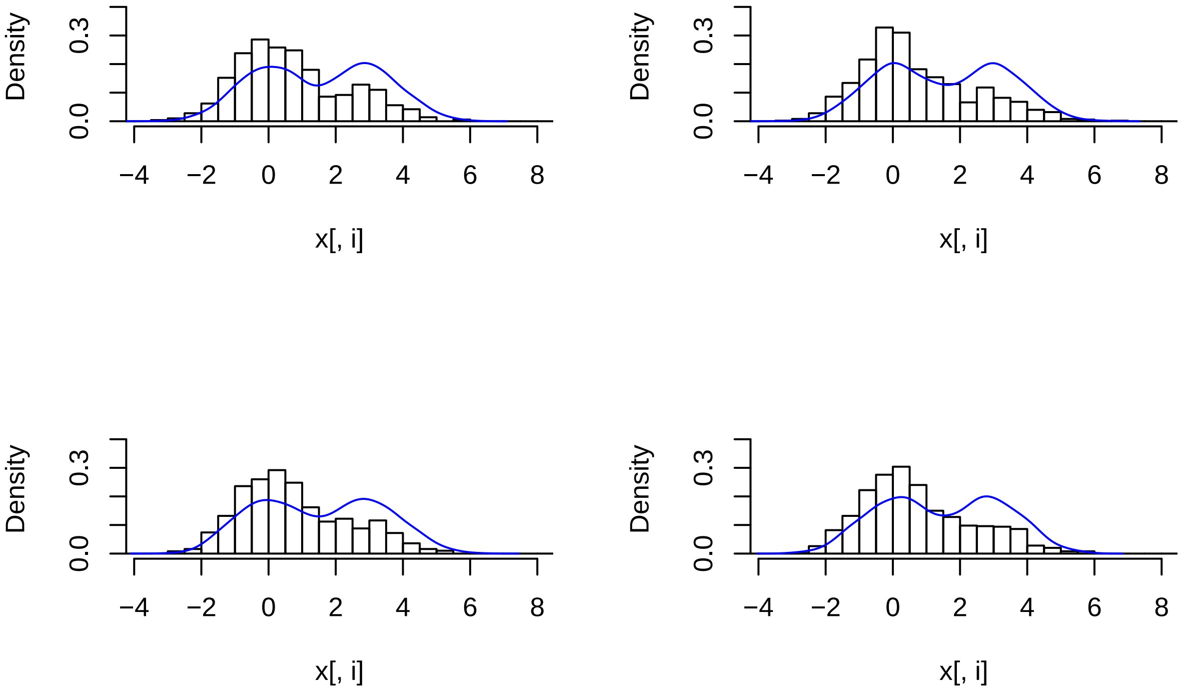
```
## [1] 0.3999263
```

This output is consistent with our theory in two ways. First, the regression coefficients we obtaind were on average very close to the provided values of B0 = 1, B1 = 2, and B2 = 3. And second, the theoretical value for the standard errors of the estimators of the regression coefficients (SE.b1 = .72, observed = .717; SE.b2 = .400, observed = .403) were very close to the observed simulated values, despite the smaller sample size of 20.

Question 3

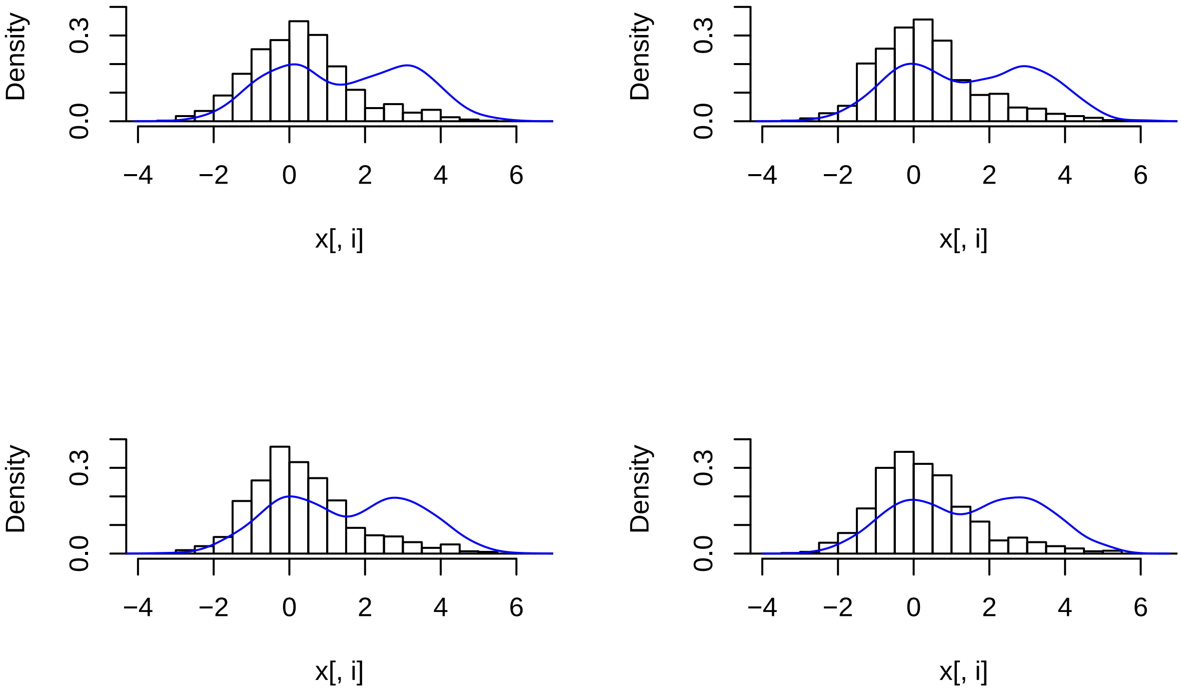
```r
library(MASS)
loc.mix <- function(n, p, mu1, mu2, Sigma) {

n1 <- rbinom(1, size = n, prob = p)
n2 <- n - n1
x1 <- mvrnorm(n1, mu = mu1, Sigma)
x2 <- mvrnorm(n2, mu = mu2, Sigma)
X <- rbind(x1, x2)
return(X[sample(1:n), ])
}
```

```r
#Generate 1000 values from two multivariate normal components:
x <- loc.mix(1000,
             p = .75, # p1 = .75
             mu1 = rep(0, 4), #First component from N(0,1)
             mu2 = rep(3, 4), #Second component from N(3,1)
             Sigma = diag(4))
r <- range(x) * 1.2
par(mfrow = c(2, 2))
for (i in 1:4){
    hist(x[ , i], xlim = r, ylim = c(0, .45), freq = FALSE,
         main = "", breaks = seq(-5, 10, .5))
  #superimpose multivariate normal with two components from N(0,1) and N(3,1)
    lines(density(mvrnorm(1000,mu=c(0,3),diag(2))), col = "blue")
}
```



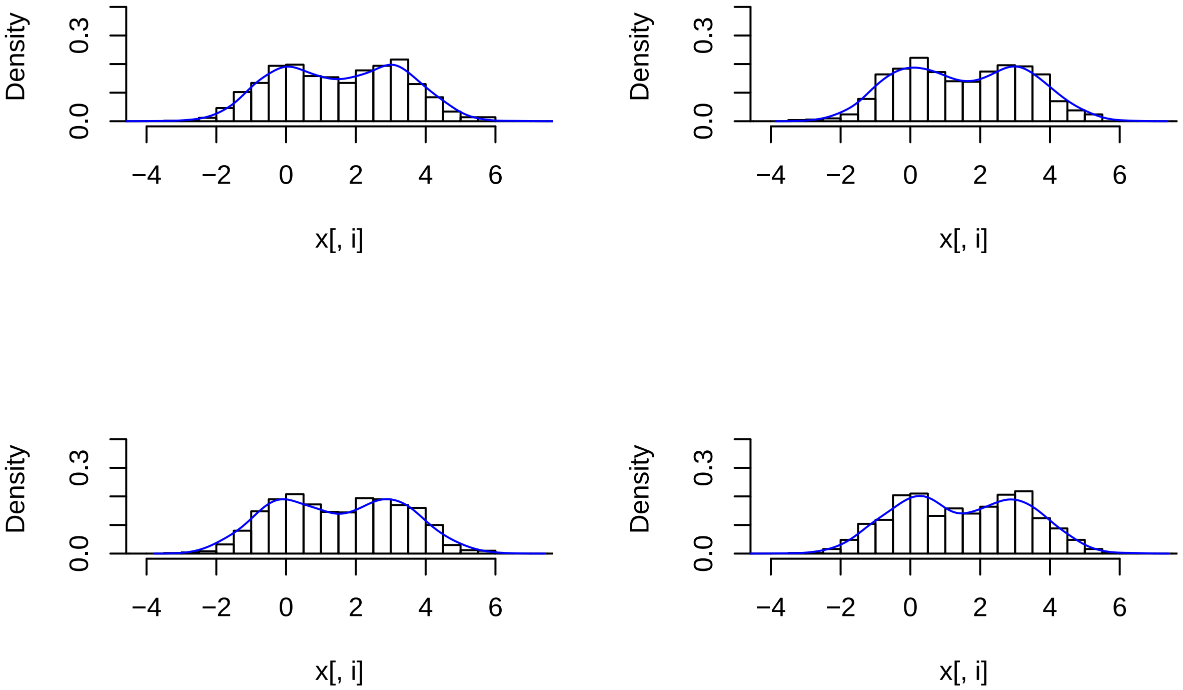
Repeat with varying levels for p1:

```
### p1 = .9
x <- loc.mix(1000,
             p = .9,
             mu1 = rep(0, 4),
             mu2 = rep(3, 4),
             Sigma = diag(4))
r <- range(x) * 1.2
par(mfrow = c(2, 2))
for (i in 1:4){
    hist(x[ , i], xlim = r, ylim = c(0, .45), freq = FALSE,
         main = "", breaks = seq(-5, 10, .5))
  #superimpose multivariate normal with two components from N(0,1) and N(3,1)
    lines(density(mvrnorm(1000,mu=c(0,3),diag(2))), col = "blue")
}
```



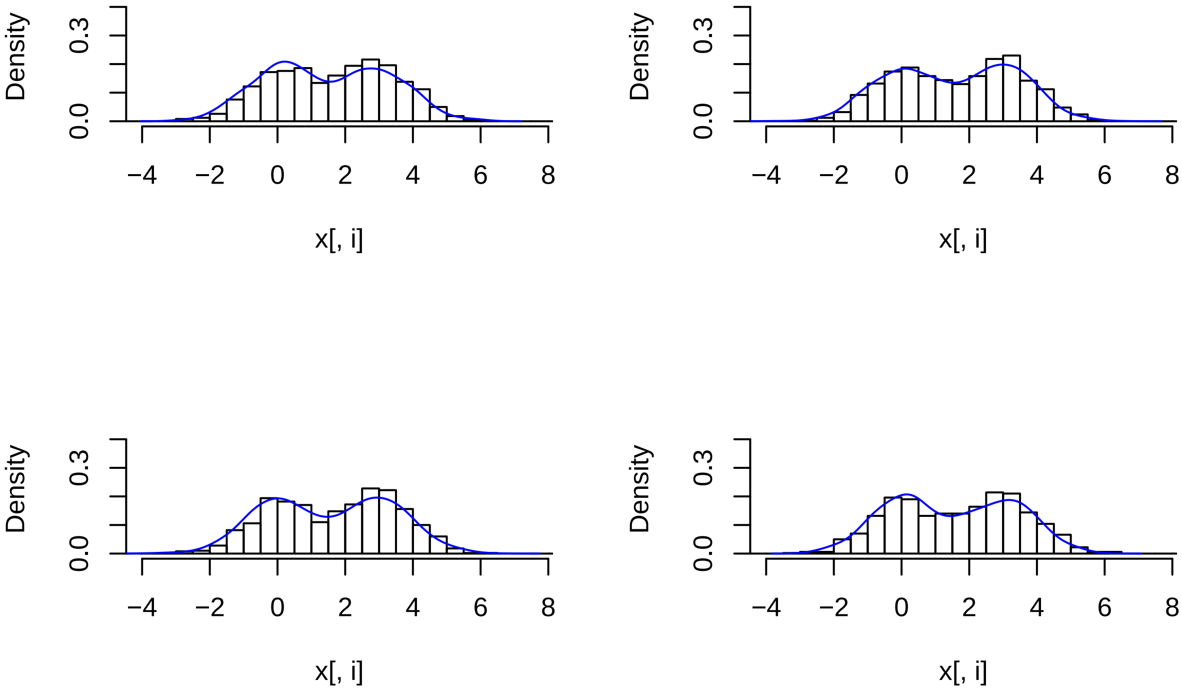
p1 = .5

```
# p1 = .5
x <- loc.mix(1000,
             p = .5,
             mu1 = rep(0, 4),
             mu2 = rep(3, 4),
             Sigma = diag(4))
r <- range(x) * 1.2
par(mfrow = c(2, 2))
for (i in 1:4){
    hist(x[ , i], xlim = r, ylim = c(0, .45), freq = FALSE,
         main = "", breaks = seq(-5, 10, .5))
  #superimpose multivariate normal with two components from N(0,1) and N(3,1)
    lines(density(mvrnorm(1000,mu=c(0,3),diag(2))), col = "blue")
}
```

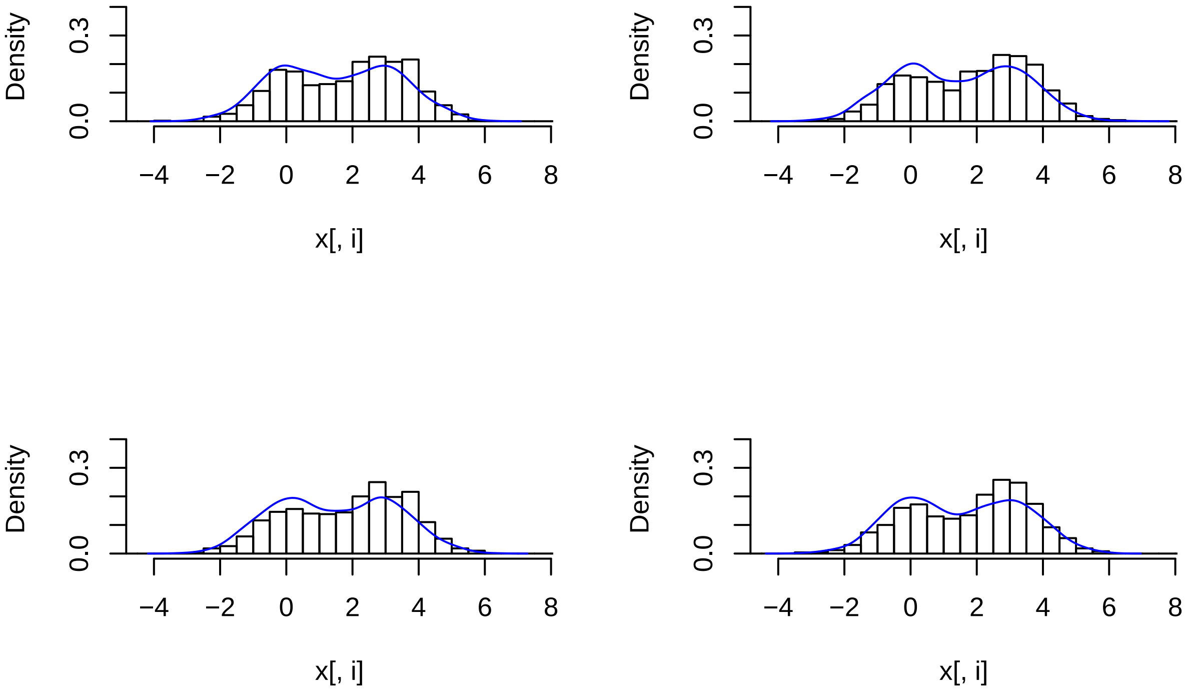


To obtain a true bimodal distribution, the probability of sampling from each component must be .5. However, the following illustrations, with p1 = .45, .4, .35, and .3 respectively, show that the histogram will look fairly bimodal up until around .3, where it looks skewed with a large tail in the direction of the distribution with the smaller sampling probability.

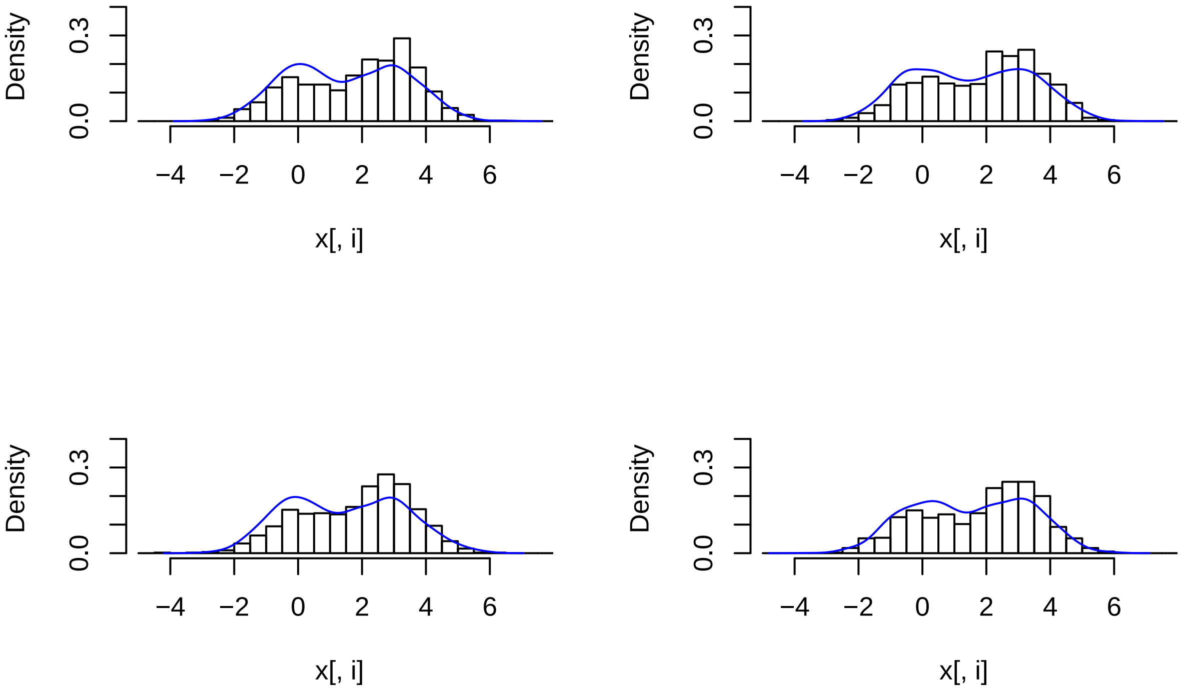
```
# p1 = .45
x <- loc.mix(1000,
             p = .45,
             mu1 = rep(0, 4),
             mu2 = rep(3, 4),
             Sigma = diag(4))
r <- range(x) * 1.2
par(mfrow = c(2, 2))
for (i in 1:4){
    hist(x[ , i], xlim = r, ylim = c(0, .45), freq = FALSE,
         main = "", breaks = seq(-5, 10, .5))
  #superimpose multivariate normal with two components from N(0,1) and N(3,1)
    lines(density(mvrnorm(1000,mu=c(0,3),diag(2))), col = "blue")
}
```



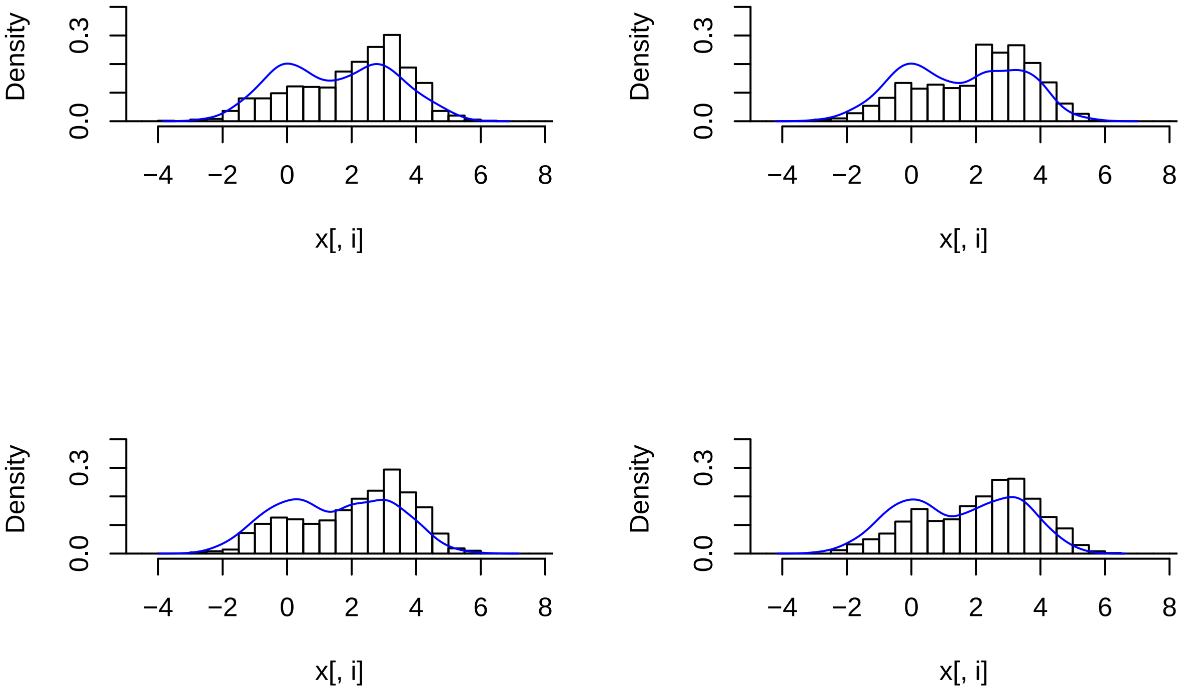
```r
# p1 = .4
x <- loc.mix(1000,
             p = .4,
             mu1 = rep(0, 4),
             mu2 = rep(3, 4),
             Sigma = diag(4))
r <- range(x) * 1.2
par(mfrow = c(2, 2))
for (i in 1:4){
    hist(x[ , i], xlim = r, ylim = c(0, .45), freq = FALSE,
         main = "", breaks = seq(-5, 10, .5))
  #superimpose multivariate normal with two components from N(0,1) and N(3,1)
    lines(density(mvrnorm(1000,mu=c(0,3),diag(2))), col = "blue")
}
```



```r
# p1 = .35
x <- loc.mix(1000,
             p = .35,
             mu1 = rep(0, 4),
             mu2 = rep(3, 4),
             Sigma = diag(4))
r <- range(x) * 1.2
par(mfrow = c(2, 2))
for (i in 1:4){
    hist(x[ , i], xlim = r, ylim = c(0, .45), freq = FALSE,
         main = "", breaks = seq(-5, 10, .5))
  #superimpose multivariate normal with two components from N(0,1) and N(3,1)
    lines(density(mvrnorm(1000,mu=c(0,3),diag(2))), col = "blue")
}
```



```
# p1 = .3
x <- loc.mix(1000,
             p = .3,
             mu1 = rep(0, 4),
             mu2 = rep(3, 4),
             Sigma = diag(4))
r <- range(x) * 1.2
par(mfrow = c(2, 2))
for (i in 1:4){
    hist(x[ , i], xlim = r, ylim = c(0, .45), freq = FALSE,
         main = "", breaks = seq(-5, 10, .5))
  #superimpose multivariate normal with two components from N(0,1) and N(3,1)
    lines(density(mvrnorm(1000,mu=c(0,3),diag(2))), col = "blue")
}
```


Density
0.0
0.3
−4
−2
0
2
4
6
8
x[, i]
Density
0.0
0.3
−4
−2
0
2
4
6
8
x[, i]
Density
0.0
0.3
−4
−2
0
2
4
6
8
x[, i]
Density
0.0
0.3
−4
−2
0
2
4
6
8
x[, i]