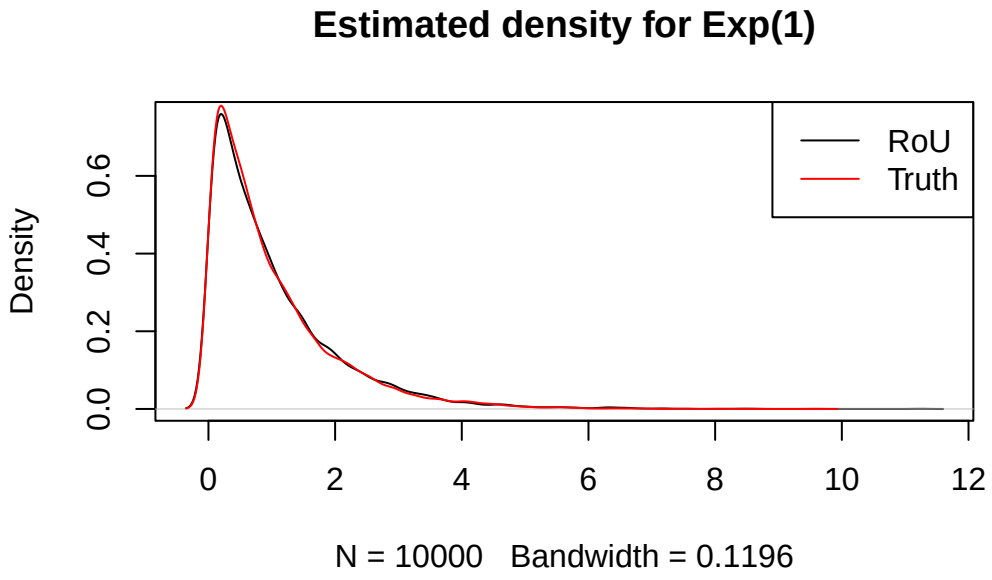
MTH210: Lab 5 Solutions

Ratio-of-Uniforms and Multivariate Normal

1. **Complete the code in the file `RoU.R` that has the code for RoU for the Exponential(1) distribution. This generates 10^4 samples from Exp(1) distribution (in a way that's different from before).**

```
#####################################################
### Ratio of Uniforms for Exp(1)
#####################################################
set.seed(1)
# function to sample from the rectangle
drawFromRect <- function(a, b, c)
{
  u <- runif(1, min = 0, max = a)
  v <- runif(1, min = b, max = c)
  return(c(u,v))
}

# sqrt f function
sqrt.f <- function(x) exp(-x/2)

# Starting the process for Exp(1)
a <-  1
b <-  0
c <-  2/exp(1)
prob.of.acceptance <- 1/(2*a*(c-b))  # true prob. of acceptance for AR

N <- 1e4 # number of samples
samp <- numeric(length = N)
i <- 1
counter <- 0  # to check acceptance
while(i <= N)
{
  counter <- counter + 1
```

```
  prop <- drawFromRect(a = a, b = b, c = c)
  vbyu <- prop[2]/prop[1]
  if( prop[1] <  exp(-prop[2]/(2*prop[1])) )
  {
    samp[i] <- prop[2]/prop[1]
    i <- i + 1
  }
}


plot(density(samp), main = "Estimated density for Exp(1)")
lines(density(rexp(1e4, 1)), col = "red")
legend("topright", col = c("black", "red"), lty = 1, legend = c("RoU", "Truth"))
```




```
(prob.of.acceptance)
```

```
[1] 0.6795705
```

```
# [1] 0.6795705

N/counter  # very close
```

```
[1] 0.6796248
```

```
# [1] 0.6796248
```

2. **Consider using RoU method to sample from $N(\theta, \sigma^2)$, where the pdf is**

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\theta)^2/(2\sigma^2)}$$

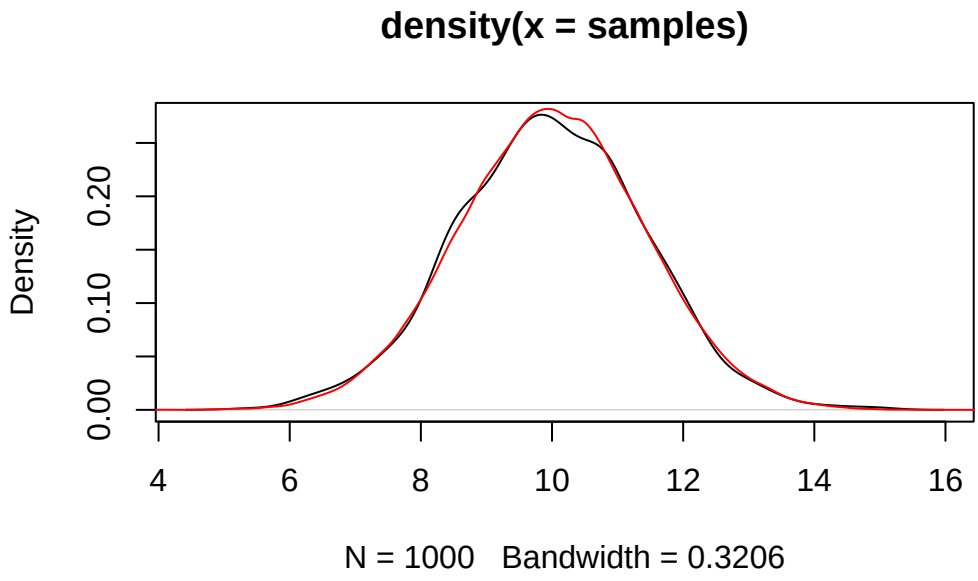
Note that the set D is

$$D = \left\{ (u, v) : 0 \leq u \leq \left(\frac{1}{2\pi\sigma^2} \right)^{1/4} e^{-(v-u\theta)^2/4\sigma^2 u^2} \right\}$$

Go through the example in the notes to find a, b, c, and then draw 10^4 samples using RoU method.

```
################################################
#### RoU for for N(theta, s2)
#### We will choose values of theta, s2
################################################

RoUNorm <- function(N, theta, s2)
{
  # a
  a <- 1/(2*pi*s2)^(.25)

  # b
  xb <- (theta - sqrt(theta^2 + 8*s2))/(2)
  b <- xb * sqrt(dnorm(xb, mean = theta, sd = sqrt(s2)))

  # c
  xc <- (theta + sqrt(theta^2 + 8*s2))/(2)
  c <- xc * sqrt(dnorm(xc, mean = theta, sd = sqrt(s2)))

  samples <- numeric(length = N)
  n <- 0
  while(n < N)
  {
    prop.u <- runif(1, min = 0, max = a)
    prop.v <- runif(1, min = b, max = c)
    if(prop.u < sqrt(dnorm(prop.v/prop.u, mean = theta, sd = sqrt(s2)) ))
    {
      n <- n+1
      samples[n] <- prop.v/prop.u
    }
  }
```

```
  return(samples)
}

samples <- RoUNorm(N = 1e3, 10, 2)

plot(density(samples), type = 'l')
lines(density(rnorm(1e5, mean = 10, sd = sqrt(2))), col = "red")
```

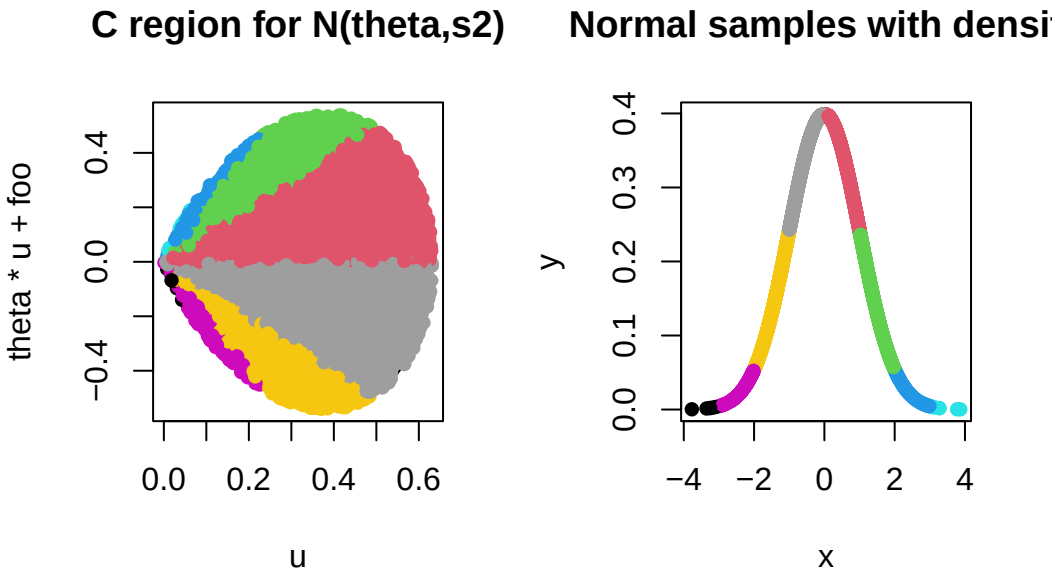


3. **Run the code in `RoURegion.R` to visualize the RoU region for Exp(1) and $N(\theta, \sigma^2)$. The code is complete, but carefully understand all the steps.**

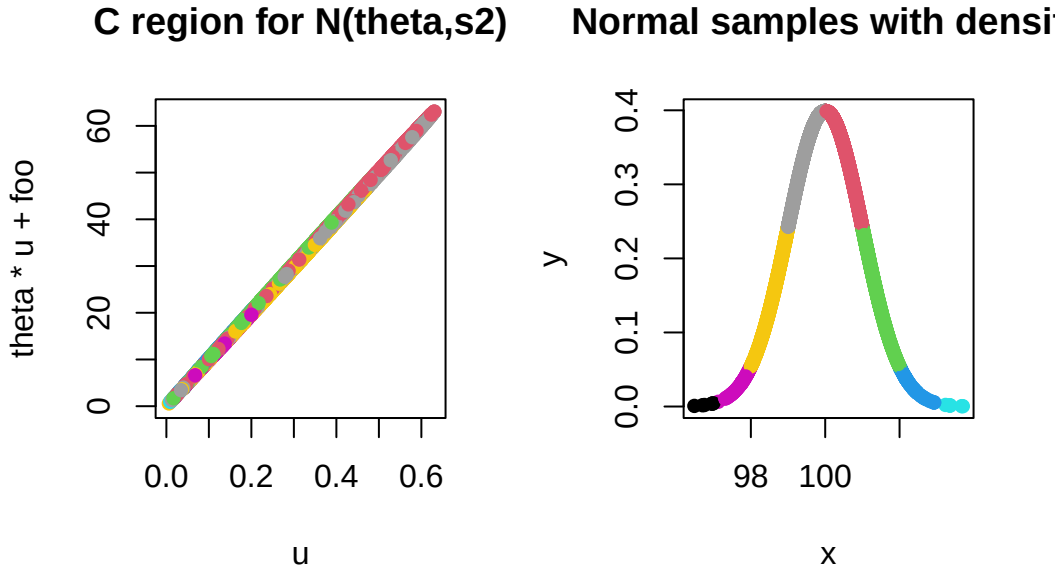
   a. **Change the values of θ and σ^2 to see what the shape turns out to be.**

```
################################################
#### RoU region for N(theta, s2)
################################################

theta <- 0
s2 <- 1

a <- 1/(2*pi*s2)^(.25)
xb <- (theta - sqrt(theta^2 + 8*s2))/(2)
b <- xb * sqrt(dnorm(xb, mean = theta, sd = sqrt(s2)))

xc <- (theta + sqrt(theta^2 + 8*s2))/(2)
c <- xc * sqrt(dnorm(xc, mean = theta, sd = sqrt(s2)))
```

```r
N <- 5e3
samples <- matrix(0, nrow = N, ncol = 2)
n <- 0
while(n < N)
{
  prop.u <- runif(1, min = 0, max = a)
  prop.v <- runif(1, min = b, max = c)
  if(abs(prop.v - theta*prop.u) <= sqrt(-4* s2* prop.u^2 *(log(prop.u) + log(2*pi*s2)/4) ) )
  {
    n <- n+1
    samples[n, ] <- c(prop.u,prop.v)
  }
}

x <- samples[,2]/samples[,1]
z <- (x - theta)/sqrt(s2)
# define color based on regions
color <- 0
for(i in 0:3)
{
  color <- color + (i+1)*(z > i & z < (i+1) )
}
for(i in (-1:-3) )
{
  color <- color + (i+8)*(z > (i) & z < (i+1))
}



# to plot the regions
u <- seq(0.00001, (2*pi*s2)^(-.25), length = 1e3)
foo <-  sqrt(-4*s2*u^2 *(log(u) + log(2*pi*s2)/4))

par(mfrow = c(1,2))
plot(u, theta * u + foo, type = 'l', main = "C region for N(theta,s2)", ylim = range(c(theta
lines(u,theta * u - foo )

points(samples, col = color + 1, pch = 16)

y <- dnorm(x, mean = theta, sd = sqrt(s2))
plot(x, y, col = color + 1, pch = 16, main = "Normal samples with density")
```



Repeating this for $N(100, 1)$ we get



b. **For a fixed value of σ^2, what do you think will happen to the efficiency of the RoU algorithm as θ increases?**

For a fixed value of σ^2, as θ increases, the region D becomes increasingly thinner, yielding a larger encapsulating box. Thus RoU will get increasingly more inefficient.

4. **Implement the RoU algorithm for $f(x) = \mathbb{I}(2 < x < 3)$.**

First, let's do theory for this. Since the target density is of the simple form, notice that the region D is

$$D = \{(u, v) : u \leq \mathbb{I}(2 \leq v/u \leq 3)\}$$

It is natural to see in this case that $a = 1$. Now:

$$c = \sup_{x \in (2,3)} x\, \mathbb{I}(2 < x < 3) = 3$$

Further, the smallest value on the v axis is

$$b = \inf_{x \in (2,3)} x \mathbb{I}(2 < x < 3) = 0\,.$$

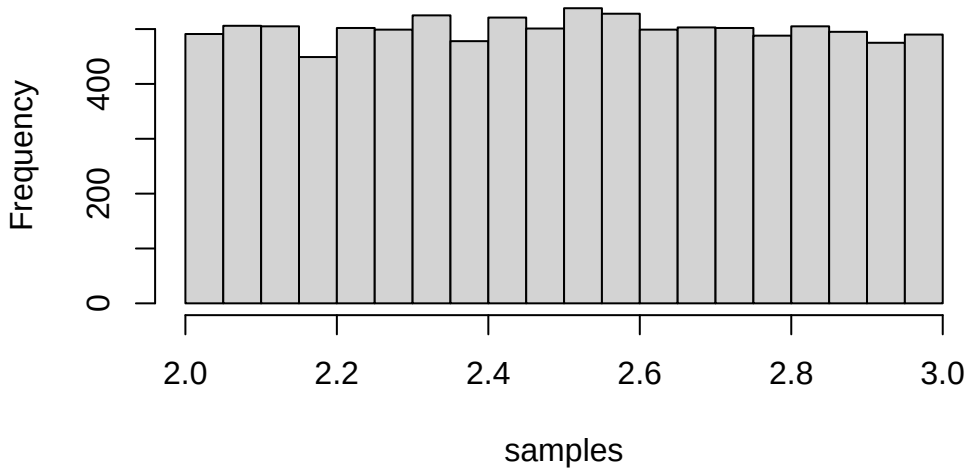
So now the AR step is:

a. Draw $(U, V) \sim Unif[0, 1] \times [0, 3]$.

b. If $2 \leq v/u \leq 3$, then set $X = V/U$.

c. Otherwise, repeat

```
################################################
#### Sample from Unif(2,3)
################################################
RoUunif <- function(N = 1e3)
{
  a <- 1
  b <- 0
  c <- 3

  samples <- numeric(length = N)
  n <- 0
  while(n < N)
  {
    prop.u <- runif(1)
    prop.v <- 3*runif(1)
    if( (2 < prop.v/prop.u) & (prop.v/prop.u < 3))
    {
      n <- n + 1
      samples[n] <- prop.v/prop.u
    }
  }
  return(samples)
}

samples <- RoUunif(1e4)
hist(samples) # checking if roughly uniform
```



5. **Multivariate Normal: In class we have learned about sampling from the multivariate normal. Suppose $\mu \in \mathbb{R}^p$ and $\Sigma \in \mathbb{R}^{p \times p}$ be a positive-definite matrix, so that we want to draw**

$$X \sim N_p(\mu, \Sigma)\,.$$

We will consider $p = 2$ and $\mu = (-5\ 10)^T$ and for $|\rho| < 1$

$$\Sigma = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}.$$

```
# defining mean and variance matrix
mu <- c(-5, 10)
Sigma <- matrix(c(1, .5, .5, 1), nrow = 2, ncol = 2)
```

Recall that drawing from this distribution involves first finding the eigenvalue decomposition of Σ

$$\Sigma = Q\Lambda Q^{-1}\,,$$

where Q is the matrix of eigenvectors and Λ is the diagonal matrix of eigenvalues $(\lambda_1, \lambda_2, \ldots, \lambda_p)$. An eigenvalue decomposition for a matrix can be done using `eigen()` function:

```
# eigen value decomposition
decomp <- eigen(Sigma)
```

```r
# eigenvectors
decomp$vectors
```

```
          [,1]       [,2]
[1,] 0.7071068 -0.7071068
[2,] 0.7071068  0.7071068
```

```r
# eigenvalues
decomp$values
```

```
[1] 1.5 0.5
```

```r
# Let's check if the decomposition is right
# yes it is!!
decomp$vectors %*% diag(decomp$values) %*% solve(decomp$vectors)
```

```
     [,1] [,2]
[1,]  1.0  0.5
[2,]  0.5  1.0
```

From this, we can calculate the "square-root" of the matrix

$$\Sigma^{1/2} = Q\Lambda^{1/2}Q^{-1}\,.$$

```r
# Finding matrix square-root
Sig.sq <- decomp$vectors %*% diag(decomp$values^(1/2)) %*% solve(decomp$vectors)
```

Now, in order to generate observations from $N_p(\mu, \Sigma)$, we obtain $Z = (Z_1, Z_2, \ldots, Z_p)^T$ and set

$$X = \mu + \Sigma^{1/2}Z\,.$$

```r
Z <- rnorm(2)  # Z
X = mu + Sig.sq %*% Z
X  # one draw from N(mu, Sigma)
```

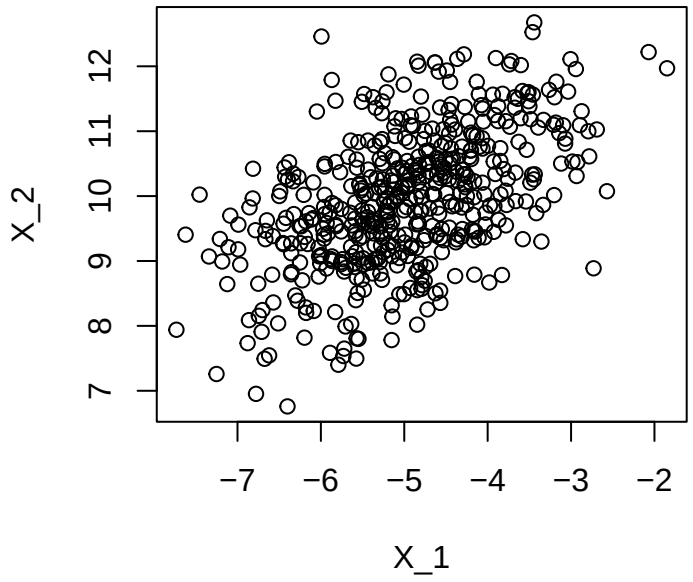
```
          [,1]
[1,] -3.570709
[2,] 10.642306
```

Using all of this information, we can now write a function `multinorm(mu, rho, N)` which takes arguments μ, ρ, and number of samples N, and returns the $N \times 2$ matrix of sampled values.

```
multinorm <- function(mu, rho, N = 5e2)
{
  Sigma <- matrix(c(1, rho, rho, 1), nrow = 2, ncol = 2)
  ...
  ...
  samples <- matrix(0, nrow = N, ncol = 2)
  for(i in 1:N)
  {
    .....
    samples[i, ] <-
  }
}
```

Use your to draw 500 samples from the bivariate normal with $\mu = (-5, 10)^T$ and $\rho = .5$.

```
samples <- multinorm(mu = c(-5, 10), rho = .5)
plot(samples, xlab = "X_1", ylab = "X_2")
```



a. **Make a similar plot for $\rho = -.9, -.5, 0, .5, .99$.**

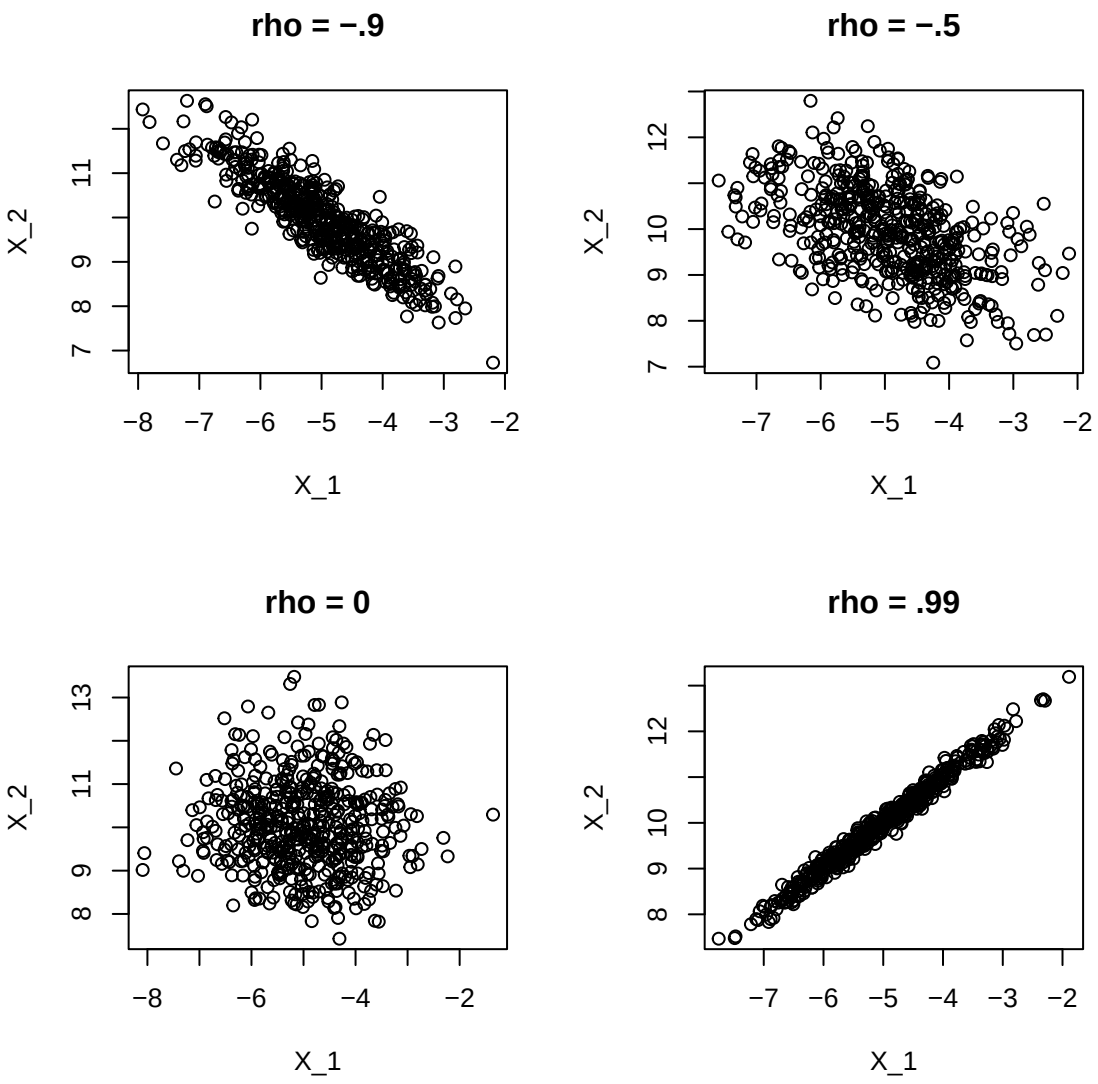
First, I complete the function so I can call it for different ρ.

```
multinorm <- function(mu, rho, N = 5e2)
{
  Sigma <- matrix(c(1, rho, rho, 1), nrow = 2, ncol = 2)
  # Eigenvalue (spectral) decomposition
  decomp <- eigen(Sigma)
```

```
  # Finding matrix square-root
  Sig.sq <- decomp$vectors %*% diag(decomp$values^(1/2)) %*% solve(decomp$vectors)

  samp <- matrix(0, nrow = N, ncol = 2)
  for(i in 1:N)
  {
    Z <- rnorm(2)
    samp[i, ] <- mu + Sig.sq %*% Z
  }
  return(samp)
}
```

Now, let's call this for different ρ.

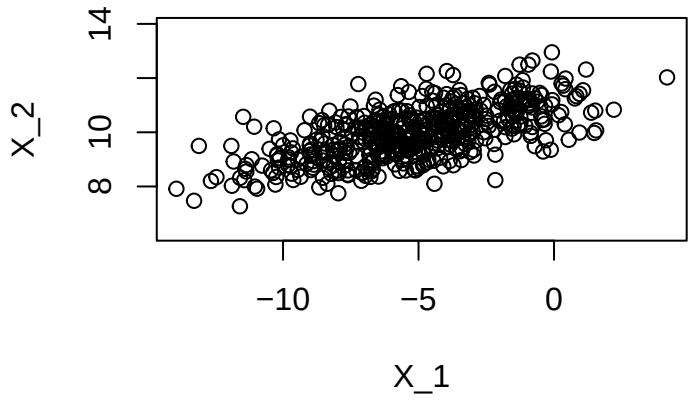
```
par(mfrow = c(2,2))
samp1 <- multinorm(mu = c(-5,10), rho = -.9)
samp2 <- multinorm(mu = c(-5,10), rho = -.5)
samp3 <- multinorm(mu = c(-5,10), rho = 0)
samp4 <- multinorm(mu = c(-5,10), rho = .99)

plot(samp1, xlab = "X_1", ylab = "X_2", main = "rho = -.9")
plot(samp2, xlab = "X_1", ylab = "X_2", main = "rho = -.5")
plot(samp3, xlab = "X_1", ylab = "X_2", main = "rho = 0")
plot(samp4, xlab = "X_1", ylab = "X_2", main = "rho = .99")
```



b. **Repeat the same where marginal variances are 10 and 1 and the off-diagonal elements are 2.**

```
multinorm2 <- function(mu, N = 5e2)
{
  Sigma <- matrix(c(10, 2, 2, 1), nrow = 2, ncol = 2)
  # Eigenvalue (spectral) decomposition
  decomp <- eigen(Sigma)

  # Finding matrix square-root
  Sig.sq <- decomp$vectors %*% diag(decomp$values^(1/2)) %*% solve(decomp$vectors)

  samp <- matrix(0, nrow = N, ncol = 2)
  for(i in 1:N)
  {
```

```
    Z <- rnorm(2)
    samp[i, ] <- mu + Sig.sq %*% Z
  }
  return(samp)
}

par(mfrow = c(1,1))
samp1 <- multinorm2(mu = c(-5,10))
plot(samp1, xlab = "X_1", ylab = "X_2", asp = 1)
```



c.