STAT822 HW1

Muxin Hua

1/21/2022

Problem 1

1.1 Function `sampler(n)`

```
'Function generates n random samples from standard normal using inverse CDF method'
```

```
## [1] "Function generates n random samples from standard normal using inverse CDF method"
```

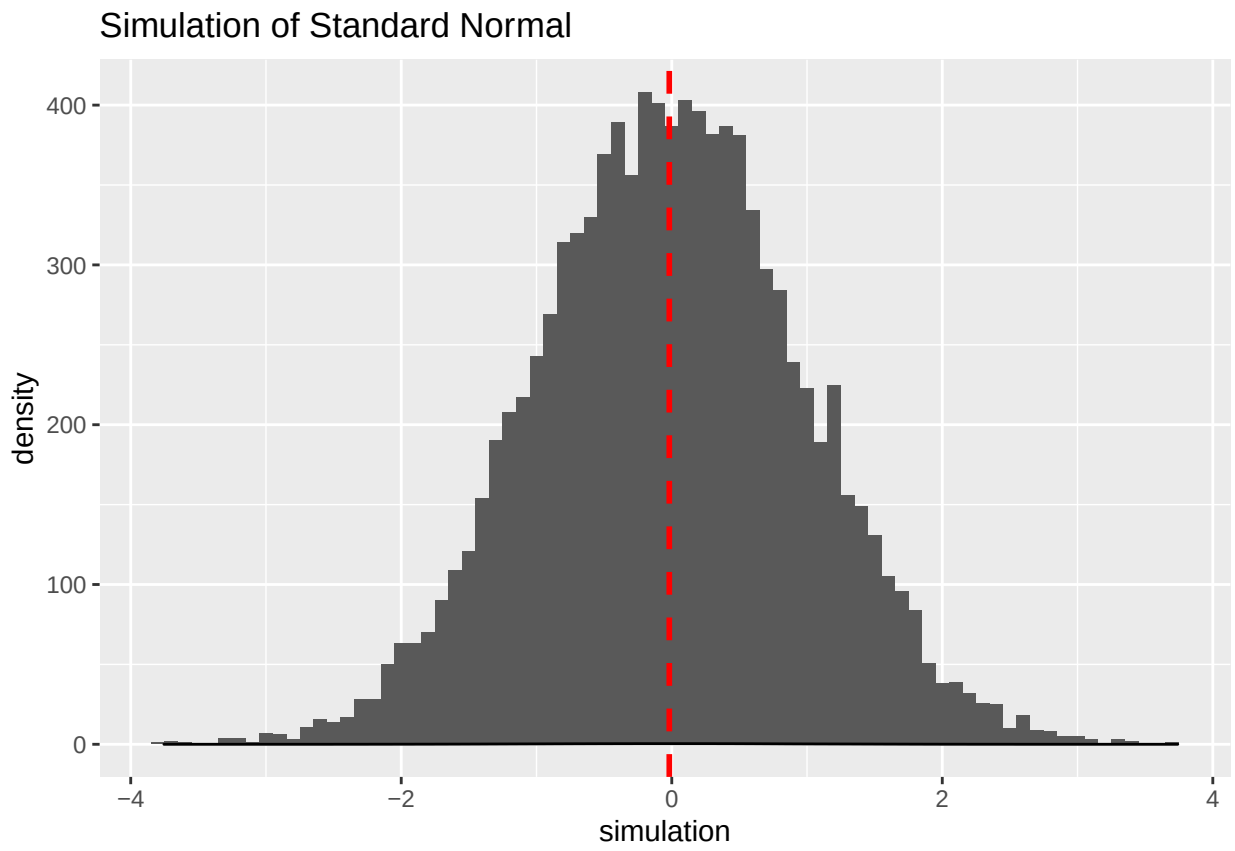
```
sampler <- function(n){ # n is the number of random variables
  uniform_sample <- runif(n, min = 0, max = 1)
  inversed <- qnorm(uniform_sample, 0, 1)
}
```

1.2 Generate 10,000 random variables

```
simulation <- sampler(10000)
```

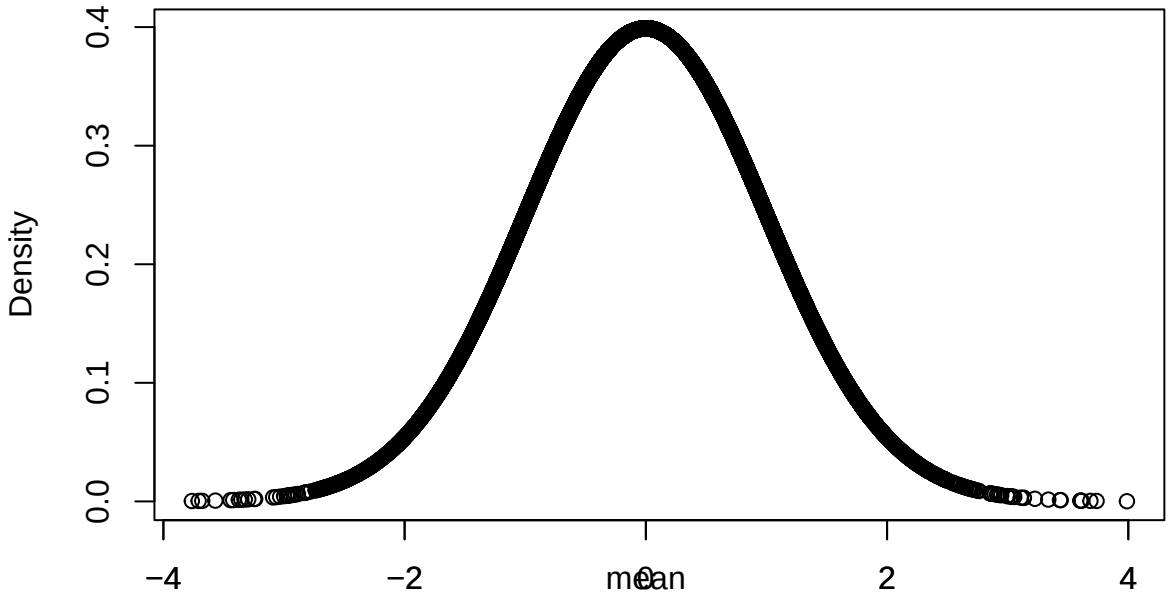
1.3 Create a histogram from result of `sampler(n)`

```
library(ggplot2)
df = data.frame()
p <-  ggplot(df, aes(x=simulation)) +
        geom_histogram(binwidth = 0.1)

p + geom_density(color = 'black') +
    geom_vline(aes(xintercept = mean(simulation)),
               color = 'red',
               linetype = 'dashed',
               size = 1) +
    ggtitle('Simulation of Standard Normal')
```



Direct sample from N(0,1) distribution

```
# Prepare x's and y's
direct_x <- rnorm(10000, 0, 1)
direct_y <- dnorm(direct_x)

# Plot PDF
plot(direct_x,direct_y, xlab='', ylab='Density')
axis(1, at = c(-4, -2, 0, 2, 4),
     labels = c('-4', '-2', 'mean', '2', '4'))
```



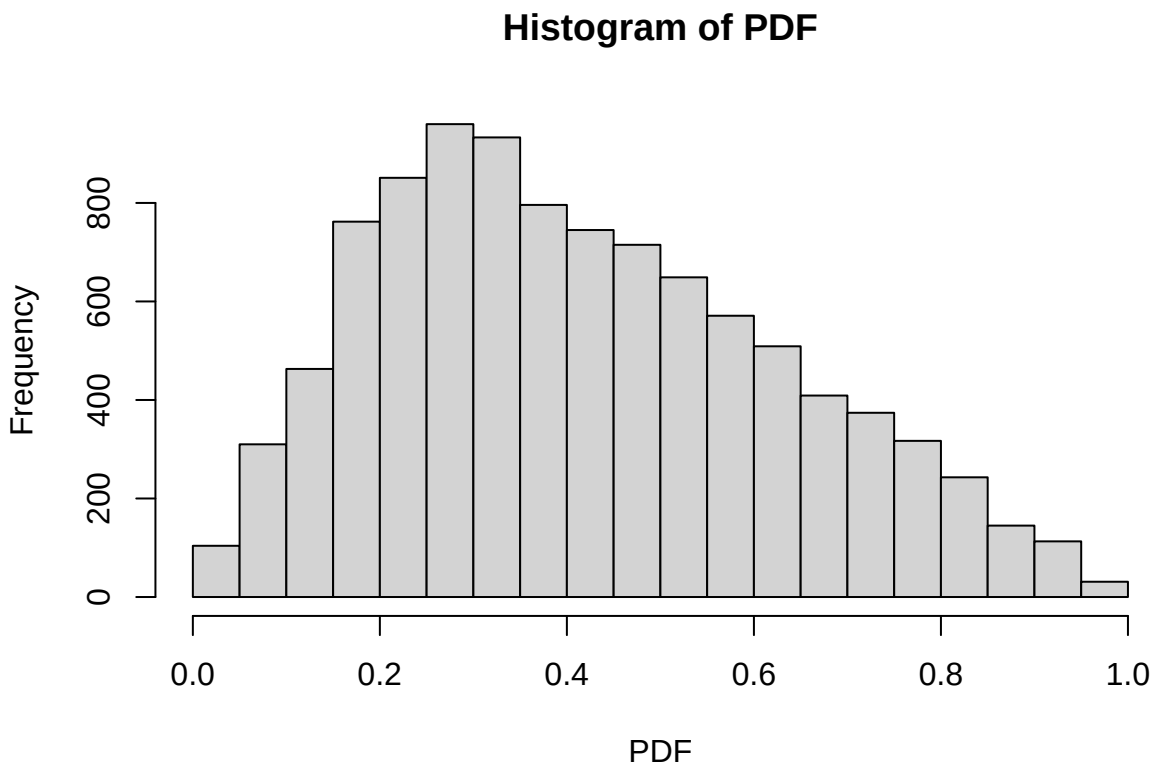
The simulated Normal distribution follows the bell-shape, and the mean is around 0, which is the theoretical mean of standard normal distribution.

Problem 2

Function `inv_triangular(n)` generates plot of triangular distribution random samples generated by inverse CDF method.

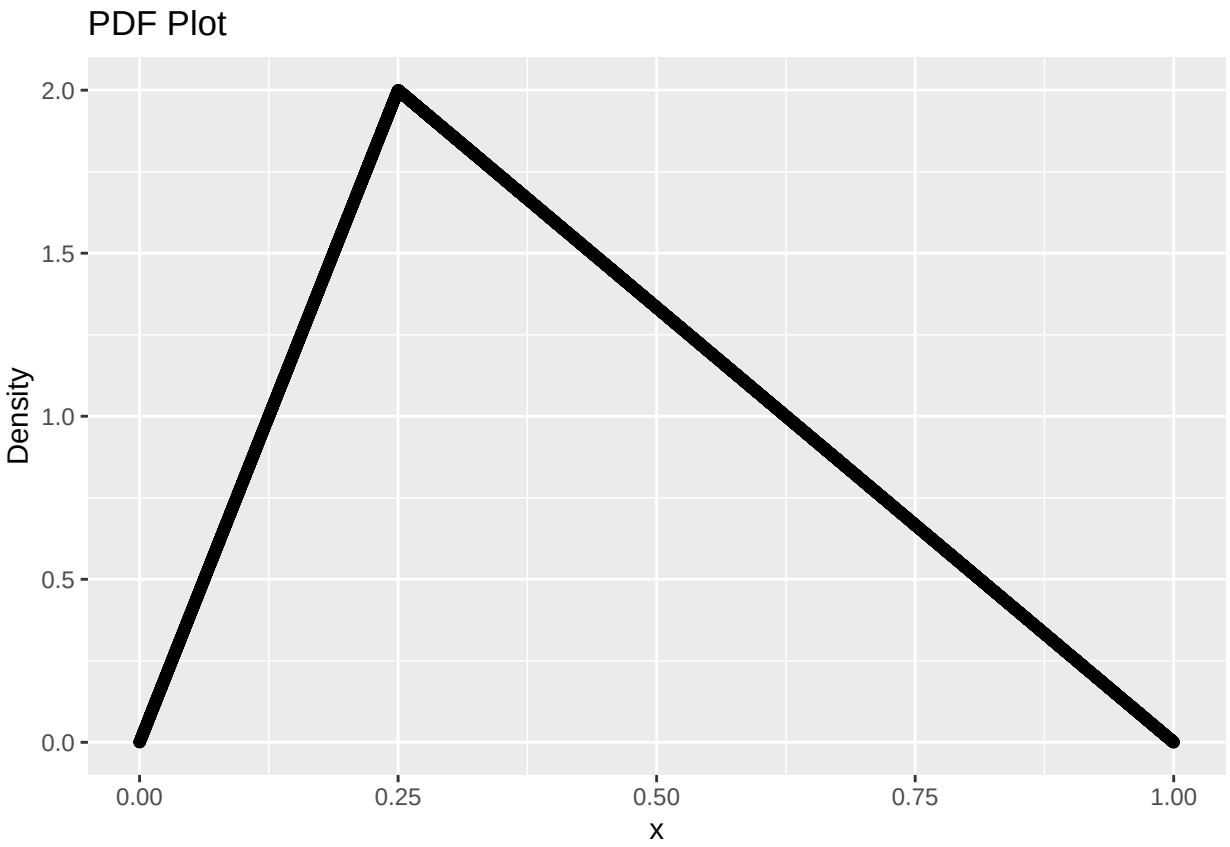
```
inv_triangular <- function(n){
  # Sample n random variables from uniform(0,1)
  uniform_sample <- runif(n)

  # Divide random samples by corresponding support of inverse PDF
  inv1 <- uniform_sample[uniform_sample < 0.25]
  inv2 <- uniform_sample[uniform_sample >= 0.25]

  # Calculate using corresponding inverse function
  u1 <- c(1/2 * sqrt(inv1))
  u2 <- 1/2 * (2 - sqrt(3 - 3*inv2))

  # Bind the simulated data together and ready to plot
  PDF <- append(u1,u2)

  hist(PDF)
}
```

```
inv_triangular(10000)
```



Plot PDF directly

```
# Sample random variables from uniform(0, 1)
s <- data.frame(runif(10000))

# Rename the column for plotting and compute corresponding pdf values
names(s)[1] <- 'x'
s['pdf'] <- ifelse(s['x'] < 0.25, 8 * s[,'x'], 8/3 - 8/3 * s[,'x'])

# Plot
ggplot(s, aes(x = x, y = pdf)) +
  geom_point() +
  ylab('Density') +
  ggtitle('PDF Plot')
```


PDF Plot
Density
2.0
1.5
1.0
0.5
0.0
0.00
0.25
0.50
0.75
1.00
x