LAPORAN TUGAS PRAKTIKUM KOMPUTASI STATISTIK
MODUL 3
"PEMBANGKITAN BILANGAN ACAK"

DISUSUN OLEH :
ANASTHASHYA RACHMAN
121450013

PROGRAM STUDI SAINS DATA
JURUSAN SAINS
INSTITUT TEKNOLOGI SUMATERA
2023

1. Pendahuluan

Pembangkitan bilangan acak merupakan suatu metode yang dilakukan serta sangat diperlukan dalam komputasi statistik untuk melakukan simulasi. Bilangan acak yang dibangkitkan merupakan pseudo random (acak semu). Pseudo random artinya jika dilakukan pengujian terhadap pengacakan, bilangan tersebut bersifat acak, tetapi sebenarnya dibangkitkan oleh sebuah fungsi. Bilangan acak yang dibangkitkan umumnya memenuhi distribusi statistik tertentu, sehingga kita memerlukan pdf/pmf dan cdf. . Pembangkit bilangan acak dapat diimplementasikan dalam komputasi statistik dapat dilakukan dengan R studio. R studio merupakan suatu bahasa pemrograman yang digunakan untuk keperluan statistik, khususnya untuk orang yang bertugas menangani dan bekerja dengan data. Pada praktikum ini bertujuan untuk memberi wawasan mengenai manfaat pembangkit bilangan acak pada analisis statistik dan implementasi pada R studio.

2. Tinjauan Pustaka

Pada praktikum kali ini mengenai pembangkitan bilangan acak, terdapat beberapa hal yang menjadi keterkaitan. Pembangkit bilangan acak merupakan algoritma yang digunakan untuk menghasilkan urutan-urutan dari angka-angka sebagai hasil perhitungan dengan komputer yang diketahui distribusinya sehingga angka-angka tersebut muncul secara random dan digunakan terus menerus. Terdapat beberapa teknik yang dapat digunakan, yaitu:

2.1 Invers-Transform Method

Inverse-Transform adalah salah satu teknik yang digunakan dalam statistik dan ilmu komputer untuk menghasilkan peubah acak dengan distribusi tertentu berdasarkan peubah acak yang berdistribusi seragam antara 0 dan 1. Metode ini sangat berguna ketika Anda ingin menghasilkan sampel dari suatu distribusi tertentu, seperti distribusi normal, eksponensial, atau poisson, dari peubah acak yang berdistribusi seragam.

Langkah-Langkah dalam pembangkitan inverse transformation untuk distribusi kontinu:
1. Tentukan CDF F(x) dari distribusi yang ingin dibangkitkan
2. Carilah invers dari CDF (quantile) $F-1(x)$
3. Bangkitkan u berdasarkan distribusi Uniform (0,1) atau u~(0,1)
4. Dapatkan bilangan acak x dengan menghitung $F-1(u)$

Langkah-Langkah dalam pembangkitan inverse transformation untuk distribusi diskrit:
1. Bangkitkan U(0,1)
2. Tentukan xi dimana $F(xi-1) < u < F(xi)$

2.2 Acceptance-rejection method

(Acceptance-Rejection Method) adalah salah satu teknik dalam statistika dan analisis probabilitas yang digunakan untuk menghasilkan sampel acak dari suatu distribusi probabilitas tertentu. Metode ini sering digunakan ketika sulit atau tidak mungkin menghasilkan sampel langsung dari distribusi yang diinginkan.

Langkah-Langkah dalam pembangkitan Acceptance - rejection method:

a. Tetapkan peubah acak Y sebagai target dari sebaran beserta PDFnya g(y)
b. Bangkitkan peubah acak Y tersebut.
c. Bangkitkan u berdasarkan distribusi Uniform (0,1) atau U(0,1)
d. Hitung nilai c dengan mencari nilai maksimum dari $f(t)/g(t)$
e. Jika $u \leq f(y)/cg(y)$, jadikan Y sebagai X
f. Jika poin 4 terpenuhi maka kembali ke poin 1

2.3 Direct Transformation

Direct transformation (transformasi langsung) adalah salah satu metode yang digunakan dalam statistika dan analisis probabilitas untuk menghasilkan sampel acak dari satu distribusi probabilitas ke distribusi probabilitas lainnya secara langsung, tanpa melalui tahap-tahap perantara.

1. Menggunakan teori dari sebaran. Seperti yang kita ketahui, sebaran dari peubah acak merupakan sebaran dari peubah acak lain. Contoh sebaran peubah acak khi-kuadrat berasal dari sebaran z dikuadratkan, sehingga untuk membangkitkan bilangan acak khi-kuadrat dengan db 1 dapat menggunakan z dikuadratkan.
2. Transformasi dari distribusi uniform ke distribusi normal dapat dilakukan dengan Transformasi Box-Muller.

3. Hasil dan Pembahasan

3.1 Peubah Acak Diskrit dengan Metode Inverse-Transform method

Dalam melakukan simulasi peubah acak diskrit dengan metode Inverse-Transform method didapatkan hasil sebagai berikut:

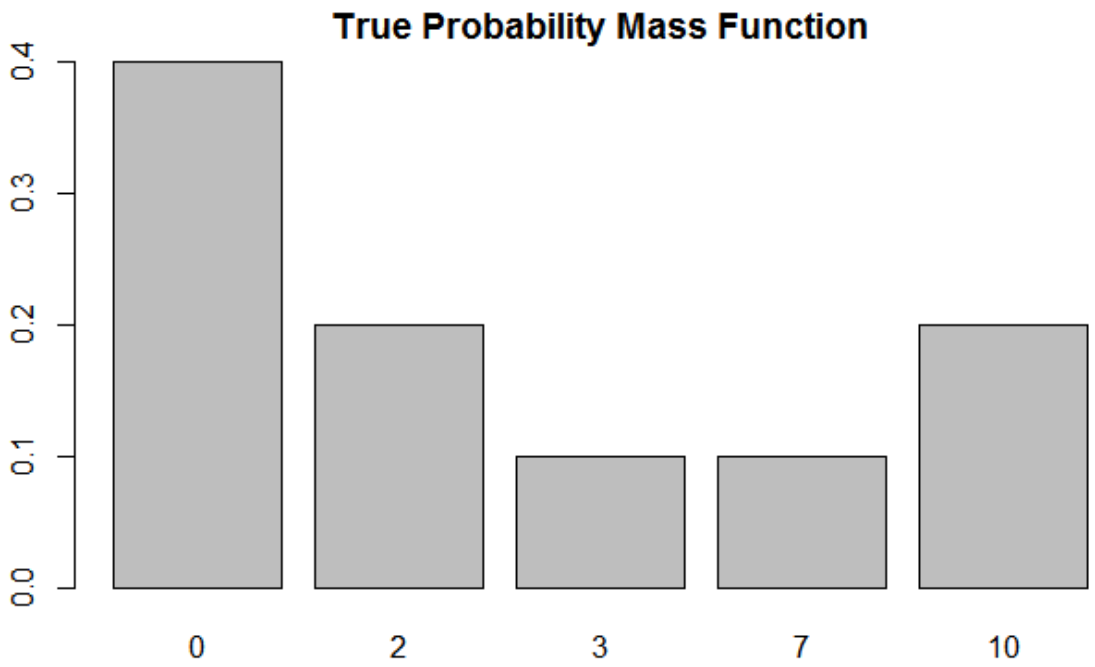
3.1.1. Distribusi dengan x_i(0, 2, 3, 7, 10) dan P(X = x_i)(0.4, 0.2, 0.1, 0.1, 0.2) dengan n=1000

Dengan menggunakan R untuk melakukan simulasi peubah acak diskrit dapat dilakukan dengan simulasi sebagai berikut:

```r
# Membuat fungsi untuk mengambil sampel dari distribusi diskrit dengan metode inverse transform
discrete.inv.transform.sample <- function(p) {
  # Menggunakan runif karena akan menghasilkan bilangan acak berdistribusi Uniform (Seragam)
  U <- runif(1)

  # Melakukan pemeriksaan dengan if atau perulangan dengan kondisi jika U kurang dari atau sama dengan probabilitas pertama
  if (U <= p[1]) {
    return(1)
  }

  # Melakukan iterasi dari peluang atau probabilitas
  for (status in 2:length(p)) {
    # Memeriksa apakah U berada dalam rentang probabilitas
    if (sum(p[1:(status-1)]) < U && U <= sum(p[1:status])) {
      return(status)
    }
  }
}
```

Analisis: Untuk melakukan pengolahan data pada kasus ini diperlukannya fungsi untuk pengambilan sampel diskrit dari distribusi uniform dengan menggunakan perulangan. Sehingga, pada kode ini digunakan untuk membantu dalam melakukan pembangkitan bilangan acak berdasarkan metode invers transform method untuk mensimulasikan peubah acak diskrit.

```r
# Menentukan jumlah sampel amatan yang akan dihasilkan
n <- 1000

# Menentukan distribusi probabilitas diskrit P(X=xi)
p <- c(0.4, 0.2, 0.1, 0.1, 0.2)

# Mendefinisikan nama pada nilai-nilai dalam distribusi probabilitas atau (xi)
names(p) <- c(0, 2, 3, 7, 10)

# Mendefinisikan samples sebagai numerik dari jumlah amatan untuk menyimpan sampel yang dihasilkan
samples <- numeric(n)

# Melakukan pengulangan dengan for sebanyak n kali untuk menghasilkan sampel
for (i in seq_len(n)) {
  # Menggunakan fungsi discrete.inv.transform.sample untuk menghasilkan sampel diskrit
  samples[i] <- discrete.inv.transform.sample(p)
}

# Membuat visualisasi barplot dari distribusi probabilitas yang didapatkan
barplot(p, main = 'True Probability Mass Function')
```



Analisis: Dalam proses yang dilakukan pada nomor terlihat bahwa, terdapat beberapa langkah, yaitu: menentukan jumlah sampel yang sesuai dengan soal, menentukan xi yang sesuai dengan soal, menentukan P(X=xi) yang sesuai dengan soal, membuat sampel numerik sesuai amatan, membuat sampel dengan iterasi dan fungsi diskrit, serta membuat hasil akhir dalam bentuk visualisasi. Pada grafik didapatkan nilai xi dengan probabilitas diskrit tertinggi pada xi = 0, yaitu P(X=xi) = 0.4

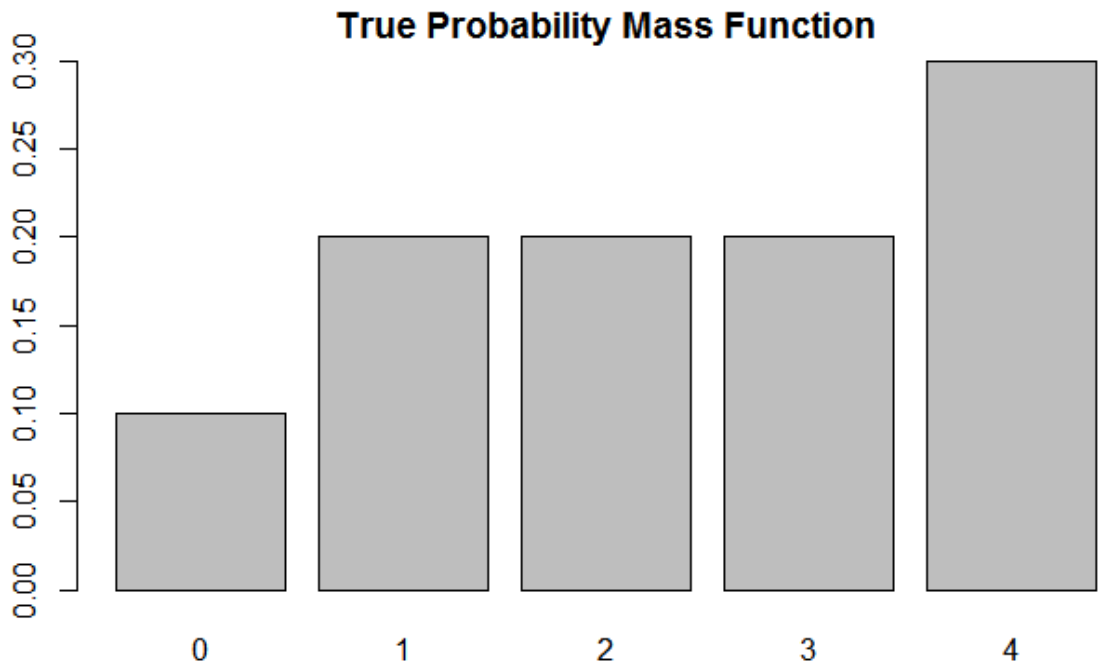
3.1.2 Distribusi dengan x_i(0, 1, 2, 3, 4) dan P(X = x_i) (0.1, 0.2, 0.2, 0.2, 0.3) dengan n=1000

Dengan menggunakan R untuk melakukan simulasi peubah acak diskrit dapat dilakukan dengan simulasi sebagai berikut:

```r
# Membuat fungsi untuk mengambil sampel dari distribusi diskrit dengan metode inverse transform
discrete.inv.transform.sample <- function(p) {
  # Menggunakan runif karena akan menghasilkan bilangan acak berdistribusi Uniform (Seragam)
  U <- runif(1)

  # Melakukan pemeriksaan dengan if atau perulangan dengan kondisi jika U kurang dari atau sama dengan probabilitas pertama
  if (U <= p[1]) {
    return(1)
  }

  # Melakukan iterasi dari peluang atau probabilitas
  for (status in 2:length(p)) {
    # Memeriksa apakah U berada dalam rentang probabilitas
    if (sum(p[1:(status-1)]) < U && U <= sum(p[1:status])) {
      return(status)
    }
  }
}
```

Analisis: Untuk melakukan pengolahan data pada kasus ini diperlukannya fungsi untuk pengambilan sampel diskrit dari distribusi uniform dengan menggunakan perulangan. Sehingga, pada kode ini digunakan untuk membantu dalam melakukan pembangkitan bilangan acak berdasarkan metode invers transform method untuk mensimulasikan peubah acak diskrit.

```r
# Menentukan jumlah sampel amatan yang akan dihasilkan
n <- 1000

# Menentukan distribusi probabilitas diskrit P(X=xi)
p<- c(0.1,0.2,0.2,0.2,0.3)

# Mendefinisikan nama pada nilai-nilai dalam distribusi probabilitas atau (xi)
names(p)<- c(0,1,2,3,4)

# Mendefinisikan samples sebagai numerik dari jumlah amatan untuk menyimpan sampel yang dihasilkan
samples <- numeric(n)

# Melakukan pengulangan dengan for sebanyak n kali untuk menghasilkan sampel
for(i in seq_len(n) ) {
  # Menggunakan fungsi discrete.inv.transform.sample untuk menghasilkan sampel diskrit
  samples[i] <- discrete.inv.transform.sample(p)
}

# Membuat visualisasi barplot dari distribusi probabilitas yang didapatkan
barplot(p, main='True Probability Mass Function')
```



Analisis : Dalam proses yang dilakukan pada nomor terlihat bahwa, terdapat beberapa langkah, yaitu: menentukan jumlah sampel yang sesuai dengan soal, menentukan xi yang sesuai dengan soal, menentukan P(X=xi) yang sesuai dengan soal, membuat sampel numerik sesuai amatan, membuat sampel dengan iterasi dan fungsi diskrit, serta membuat hasil akhir dalam bentuk visualisasi. Pada grafik didapatkan nilai xi dengan probabilitas diskrit tertinggi pada xi = 4, yaitu P(X=xi) = 0.3

3.2 Membangkitkan Bilangan Acak Menggunakan Inverse Transform method

3.2.1 Membangkitkan Bilangan Acak dengan $f(x) = 4e^{-4x}$, dengan $x \in \mathbf{R}$

Dalam membangkitkan bilangan acak diatas dapat dilakukan dengan cara manual dan menggunakan R.

3.2.1.1 Mencari CDF dan Invers CDF Untuk Pembangkitan Bilangan Acak

$$f(x) = 4e^{-4x}, \text{ dengan } x \in \mathrm{R}$$

Fungsi CDF:

$$F(X) = \int_0^x 4e^{-4x}\, dy$$

Misal u = $-4x$

du = $-4\, dx$

Sehingga, $\int_0^x 4e^{-4x}\,dx = 4\int_0^x e^{-4x}dx$

$= 4\int_0^x e^{u}\frac{-du}{4}$

$= -\int_0^x e^{u}\,du$

$= -e^{u}|_0^x$

$= -e^{-4x}|_0^x$

$= -e^{4x} + 1$

$= 1 - e^{-4x}$

Mencari Invers Fungsi CDF

$1 - e^{-4x} = y$

$e^{-4x} = 1 - y$

$ln\, e^{-4x} = ln\,(1 - y)$

$-4x = ln\,(1 - y)$

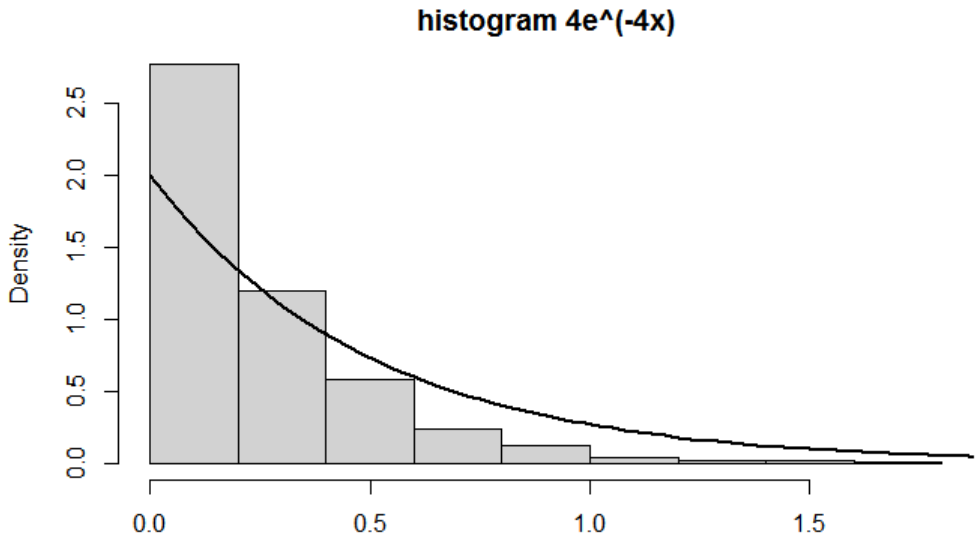
$x = \frac{-ln(1-y)}{4}$

3.2.1.2 Membangkitkan Bilangan Menggunakan R

Dalam membangkitkan bilangan acak dengan $f(x) = 4e^{-4x}$, dengan $x \in$ R dengan R dapat dilakukan sebagai berikut :

```
# Menentukan n sebagai jumlah amatan yang sesuai dengan soal
n<-1000

# Menentukan u sebagai pembangkitan bilangan acak yang berdistribusi uniform sesuai
dengan invers CDF
u<-runif(n)
u

# Menentukan x sebagai invers dari CDF
x =  -log(1-u)/4

# Membuat visualisasi dengan histogram
hist(x, freq=F, xlab = 'x', main = 'histogram 4e^(-4x)')
curve(dexp(x, rate=2) , 0, 3, lwd=2, xlab = "", ylab = "", add = T)
```

```
  [1] 9.728438e-01 1.938936e-01 8.569506e-01 8.731850e-01 2.325733e-01
  [6] 2.560602e-01 6.365446e-01 7.972325e-01 7.436824e-01 5.259849e-01
 [11] 4.441774e-01 8.828266e-01 1.787517e-01 3.577702e-01 4.985496e-01
 [16] 6.361588e-01 2.637754e-01 3.932682e-01 2.863512e-01 2.537067e-01
 [21] 2.610431e-01 4.590660e-01 9.243690e-01 6.325210e-01 8.684016e-01
 [26] 8.638310e-01 6.644303e-01 6.916871e-01 2.884816e-01 8.088382e-01
 [31] 2.683303e-01 3.367217e-01 8.089494e-01 8.657181e-01 5.765982e-01
 [36] 5.619114e-02 3.207610e-01 1.777431e-01 1.532827e-01 2.066539e-01
 [41] 9.491533e-01 9.049143e-01 1.857708e-01 7.456440e-01 8.096575e-01
 [46] 3.199174e-01 7.223521e-01 9.893780e-02 6.002157e-01 3.487182e-01
 [51] 2.143693e-01 9.323359e-01 6.974384e-01 2.138538e-01 4.452699e-01
 [56] 2.398825e-01 1.937233e-01 2.962153e-01 5.073672e-01 5.686466e-01
 [61] 2.950068e-01 5.105758e-01 9.794462e-01 2.983512e-01 6.807356e-01
 [66] 5.554894e-01 2.930200e-02 9.903802e-01 9.969612e-01 4.140649e-01
 [71] 9.375090e-01 2.951184e-01 7.481040e-01 9.435838e-01 1.578038e-01
 [76] 3.575976e-01 8.485803e-01 9.569364e-01 3.462351e-01 4.227227e-01
 [81] 2.675805e-01 5.292260e-02 7.766998e-02 9.991999e-01 9.200257e-01
 [86] 4.483424e-01 8.508332e-01 2.523103e-01 8.911883e-01 8.188325e-01
 [91] 9.228066e-01 1.916871e-02 6.944861e-01 3.606142e-01 8.949112e-01
 [96] 3.712095e-01 2.750590e-01 4.954306e-01 7.903338e-01 6.302536e-01
[101] 1.025924e-03 4.456388e-01 9.003361e-01 9.145443e-01 9.965270e-01
[106] 5.121484e-01 5.968419e-01 8.149583e-01 1.083849e-01 3.155641e-01
```



Analisis: Pada proses ini dilakukan dengan beberapa langkah, yaitu menentukan CDF dan sudah terdapat dalam soal f(x) = 4e\^(−4x), menentukan invers dan didapatkan hasil inversnya adalah -log(1-u)/4, membangkitkan u berdasarkan invers, membangkitkan bilangan acak dan mendapatkan hasil output bilangan acak dan histogram agar lebih jelas. Pada histogram terlihat bahwa, terdapat penurunan grafik pada data.

3.2.2. Bangkitkan Bilangan Acak dengan $f(x) = \frac{3}{32}x^5$ dengan $0 < x < 2$

Dalam membangkitkan bilangan acak diatas dapat dilakukan dengan cara manual dan menggunakan R.

3.2.2.1 Mencari CDF dan Invers CDF Untuk Pembangkitan Bilangan Acak

$$f(x) = \frac{3}{32}x^5 \text{ dengan } 0 < x < 2$$

Mencari Fungsi CDF

$$F(X) = \int_0^x \frac{3}{32}x^5 \, dt$$

$$= \frac{3}{32}\int_0^x x^5 \, dt$$

$$= \frac{3}{32} \times \frac{1}{6}x^6 \Big|_0^x$$

$$= \frac{1}{64}x^6 - 0$$

$$= \frac{1}{64}x^6$$

Mencari Fungsi Invers CDF

$$\frac{1}{64}x^6 = y$$

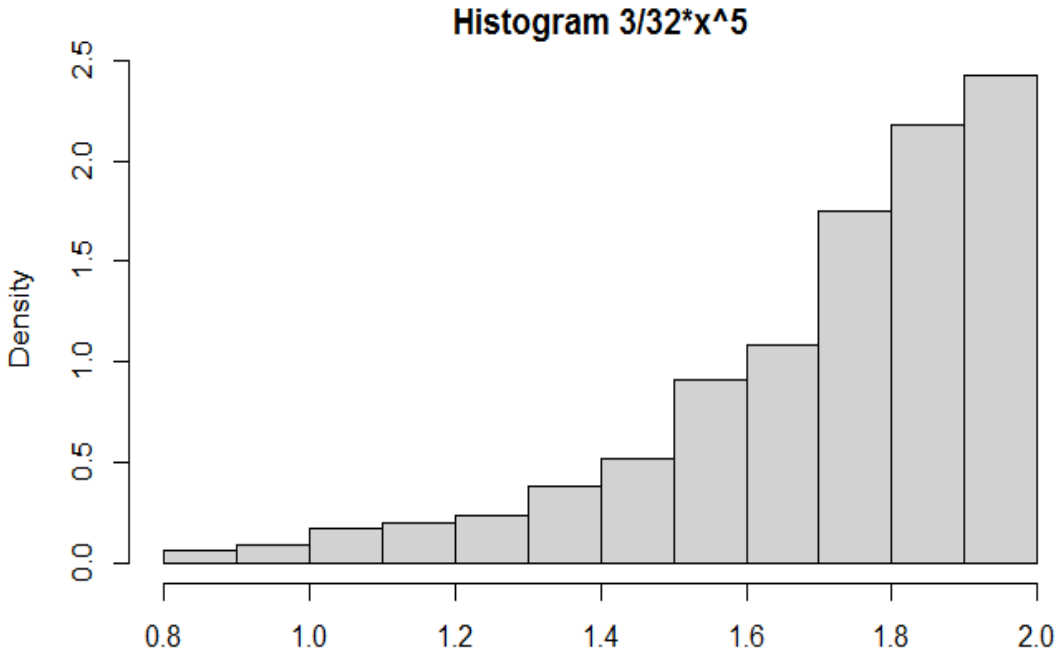
$$x^6 = 64y$$

$$x = (64)^{\frac{1}{6}}$$

3.2.1.2 Membangkitkan Bilangan Menggunakan R

Membangkitkan bilangan dengan f(x) = $\frac{3}{32}x^5$ dengan 0<x< 2 dengan R studio dapat dilakukan dengan membuat kode sebagai berikut :

```
# Menentukan n sebagai jumlah amatan yang tertera pada soal
n<-1000

# Menentukan u sebagai pembangkit bilangan acak uniform dan nilai min=0 serta nilai
max=1
u<-runif(n, min=0, max = 1)

# Menentukan x sebagai hasil dari persamaan fungsi yang telah diturunkan sebelumnya
x <- (64*u)**(1/6)#terlihat nilai x tidak ada yang di bawah 0 atau di atas 2

# Membuat histogram sebagai hasil akhir
hist(x, freq=F, xlab='x', main='Histogram 3/32*x^5')
```



Analisis: Penyelesaian pada kasus ini dilakukan dengan beberapa proses, yaitu: mendefinisikan jumlah amatan, lalu membangkitkan bilangan acak, mendapatkan persamaan x dari penurunan, dan yang terakhir dengan membuat visualisasi agar terlihat jelas hasil yang didapatkan. Pada grafik didapatkan grafik yang mengalami kenaikan yang cukup signifikan dari x pada 0,7

3.3 Membangkitkan Bilangan Acak dengan Metode Acceptance – Rejection

Fungsi PDF nya adalah :

$$f(x)=\frac{3}{2}x^3+\frac{11}{8}x^2+\frac{1}{6}x+\frac{1}{2}, \quad 0 \leq x \leq 1$$

Dalam membangkitkan bilangan acak dengan metode accepted-rejection dengan R studio dapat dilakukan sebagai berikut :

a. Tetapkan Peubah acak Y sebagai target dari sebaran beserta PDFnya g(y)

$$g(y) = 1,\ 0 \leq y \leq 1$$

b. Bangkitkan peubah acak Y tersebut.

```
set.seed(123)
X <- runif(1000)
head(X)
```

```
[1] 0.2875775 0.7883051 0.4089769 0.8830174 0.9404673 0.0455565
```

c. Bangkitkan u berdasarkan distribusi Uniform (0,1) atau U(0,1)

```
set.seed(333)
U <- runif(1000)
head(U)
```

```
[1] 0.46700066 0.08459815 0.97348527 0.57130558 0.02011937 0.72355739
```

d. Hitung nilai c dengan mencari nilai maksimum

```
h <- function(x) (3/2)*x**3 + (11/8)*x**2 + (1/6)*x + (1/2)
derivH <- optimize(f=h, interval = c(0,1), maximum = 1)
derivH$objective
```

```
[1] 3.541176
```

e. Jadikan Y sebagai X

```
f <- function(x) (3/2)*x**3 + (11/8)*x**2 + (1/6)*x + (1/2)
head(f(X))

g <- function(x) 1
head(g(x))

nilaic <- derivH$objective
```

```
[1] 0.6973176 2.2206523 0.9007578 2.7520433 3.1206377 0.5105882
[1] 1
```

f. Kriteria Penerimaan

```
criteria <- U < f(X)/(nilaic*g(X))
head(criteria)
```

```
[1] FALSE  TRUE FALSE  TRUE  TRUE FALSE
```

g. Hasil Akhir

```
sum(criteria)

Y2 <- X[criteria][1:1000]
head(Y2)

summary(Y2)
```

```
[1] 397
[1] 0.7883051 0.8830174 0.9404673 0.8924190 0.5514350 0.9568333
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
 0.0006  0.4785  0.7371  0.6623  0.8866  0.9994     603
```

Analisis : Pada penyelesaian permasalahan ini diselesaikan dengan langkah yang cukup panjang agar mendapatkan hasil dari pembangkitan bilangan acak dengan metode ARM. Langkah-langkah yang dilakukan adalah Tetapkan Peubah acak Y sebagai target dari sebaran beserta PDFnya g(y), Bangkitkan peubah acak Y

tersebut, Bangkitkan u berdasarkan distribusi Uniform (0,1) atau U(0,1), Hitung nilai c dengan mencari nilai maksimum, f. Kriteria Penerimaan, Hasil akhir. Pada semua proses yang dilakukan mendapatkan semua output dan pada hasil akhir menampilkan output berupa jumlah kriteria penerimaan dan kesimpulan summary dari data.

4. Kesimpulan

Pembangkitan bilangan acak sangat diperlukan pada proses pemodelan statistik yang berguna dalam melakukan simulasi. Penggunaan simulasi pembangkitan bilangan acak dapat dilakukan dan dianjurkan melalui software pengolahan data dan sangat disarankan untuk menggunakan R studio. Pada R studio terdapat banyak fungsi untuk membangkitkan bilangan acak berdasarkan distribusi yang sudah umum. Bilangan acak yang dibangkitkan merupakan pseudo random (acak semu). Dalam pembangkit bilangan acak terdapat 3 metode yaitu Inverse-Transform method, Acceptance-rejection method, dan Direct Transformation. Terdapat dua metode yang diterapkan pada praktikum kali ini, yaitu Inverse-Transform method dan Acceptance-rejection method. Dari metode-metode yang sudah dilakukan terlihat jelas bahwa terdapat langkah-langkah yang berbeda dalam penggunaannya. Kedua metode memiliki kelebihan dan ketertarikan sendiri serta dalam penggunaannya memiliki ciri khasnya. R studio sangat membantu dalam hal keperluan statistik dengan cara pemrograman tanpa pengerjaan manual.