Modul 3 Pembangkitan Bilangan Acak

2025-10-07

Pendahuluan

Bilangan acak (random) adalah bilangan yang kemunculannya terjadi secara acak dan diharapkan memenuhi sebaran statistik tertentu (PDF/PMF, CDF). Pembangkitan bilangan acak merupakan alat yang diperlukan dalam komputasi statistik, umumnya untuk simulasi. Semua metode pembangkitan bilangan acak tergantung dari pembangkitan bilangan acak uniform.

Pseudo-Random Number Generator merupakan algoritma untuk mendapatkan bilangan secara acak dan tidak dapat diprediksi kemunculan selanjutnya. Bilangan acak yang dibangkitkan oleh komputer disebut bilangan acak semu atau pseudo random number. Disebut pseudo karena kemunculan bilangan tersebut sangat bergantung pada seed atau nilai awal. Jika nilai seed yang digunakan adalah bilangan yang benar-benar acak, maka bilangan yang dihasilkan juga akan acak sempurna (tidak memiliki pola). Nilai seed yang sama akan menghasilkan bilangan acak yang sama pula. Untuk menetapkan nilai seed menggunakan fungsi “set.seed”.

```
#menetapkan bilangan seed
set.seed(123)
```

Membangkitkan bilangan acak pada software R dari suatu distribusi, nama fungsinya diawali dengan huruf r diikuti dengan nama distribusinya. Berikut beberapa perintah yang dapat digunakan dalam software R untuk membangkitkan suatu bilangan acak menggunakan distribusi.

Tabel 1. Random Generator Pada R

Distributionc	cdf	Generator	Parameters
Beta	pbeta	rbeta	shape1, shape2
Binomial	pbinom	rbinom	size, prob
Chi-squared	pchisq	rchisq	df
Exponential	pexp	rexp	rate
F	pf	rf	df1, df2
Gamma	pgamma	rgamma	shape, rate, scale
Geometric	pgeom	rgeom	prob
Lognormal	plnorm	rlnorm	meanlog, sdlog
Negative Binomial	pnbinom	rnbinom	size, prob
Normal	pnorm	rnorm	mean, sd
Poisson	ppois	rpois	lambda

Student's T	pt	rt	df
Uniform	punif	runif	min, max

Berikut merupakan contoh dari beberapa distribusi umum dan fungsinya pada R untuk menghasilkan bilangan acak.

1. Menggunakan distribusi uniform Fungsi yang digunakan untuk menghasilkan bilangan acak pada distribusi seragam yaitu runif(). Dibawah ini merupakan contoh penggunaan fungsi runif()

```
#contoh penggunaan runif pada pembangkitan 10 bilangan acak dari distribusi u
niform (0,1)
distribusi_uniform <- runif(10)
print(distribusi_uniform)
```

```
##  [1] 0.2875775 0.7883051 0.4089769 0.8830174 0.9404673 0.0455565 0.5281055
##  [8] 0.8924190 0.5514350 0.4566147
```

```
#contoh penggunaan runif pada pembangkitan 10 bilangan acak dari distribusi u
niform (5,20)
distribusi_uniform <- runif(10, 5, 20)
print(distribusi_uniform)
```

```
##  [1] 19.352500 11.800012 15.163560 13.589501  6.543870 18.497375  8.691316
##  [8]  5.630893  9.918811 19.317555
```

2. Menggunakan distribus normal Fungsi yang digunakan untuk menghasilkan bilangan acak pada distribusi seragam yaitu rnorm(). Dibawah ini merupakan contoh penggunaan fungsi rnomr()

```
#contoh penggunaan rnomr pada pembangkitan 10 bilangan acak dari distribusi n
ormal (0,1)
distribus_normal <- rnorm(10)
print(distribus_normal)
##  [1]  1.2240818  0.3598138  0.4007715  0.1106827 -0.5558411  1.7869131
##  [7]  0.4978505 -1.9666172  0.7013559 -0.4727914
```

```
#contoh penggunaan rnomr pada pembangkitan 10 bilangan acak dari distribusi n
ormal (10, 2)
distribus_normal <- rnorm(10, mean = 10, sd = 2)
print(distribus_normal)
```

```
##  [1]  7.864353  9.564050  7.947991  8.542218  8.749921  6.626613 11.675574
##  [8] 10.306746  7.723726 12.507630
```

3. Menggunakan distribusi binomial Fungsi yang digunakan untuk menghasilkan bilangan acak pada distribusi seragam yaitu rbinom(). Dibawah ini merupakan contoh penggunaan fungsi rbinom()

```
#contoh penggunaan rbinom pada pembangkitan 10 bilangan acak dari distribusi binomial dengan 15 percobaan dan probability 0.3
distribus_binomial <- rbinom(10, size = 15, prob = 0.3)
print(distribus_binomial)
```

```
##  [1] 5 2 4 3 6 4 6 6 6 4
```

Contoh kasus distribusi binomial

Sebuah Sekolah Menengah Atas (SMA) sedang melangsungkan ujian dengan probabilitas siswa yang menjawab pertanyaan secara benar 0,4 dari 6 siswa. Tentukan peluang paling sedikit 4 siswa menjawab benar, peluang 2 sampai 5 siswa menjawab benar, dan peluang tepat 3 siswa menjawab benar.
Jawaban :

```
# Parameter
n_siswa <- 6
p_benar <- 0.4

# Peluang paling sedikit 4 siswa menjawab benar
peluang1 <- pbinom(3, size=n_siswa, prob=p_benar, lower.tail = FALSE)
peluang1
```

```
## [1] 0.1792
```

```
# Peluang 2 sampai 5 siswa menjawab benar
peluang2 <- pbinom(5, size=n_siswa, prob=p_benar) - pbinom(1, size=n_siswa, prob=p_benar)
peluang2
```

```
## [1] 0.762624
```

```
# Peluang tepat 3 siswa menjawab benar
peluang3 <- dbinom(3, size=n_siswa, prob=p_benar)
peluang3
```

```
## [1] 0.27648
```

Teknik Pembangkitan Bilangan Acak

Jika fungsi pembangkitan bilangan acak belum tersedia akibat belum adanya sebaran yang ditentukan, maka pembangkitan bilangan acak harus didasarkan pada suatu sebaran acak terlebih dahulu. Untuk melakukannya, dapat digunakan teknik pembangkitan bilangan acak berikut:

1. Inverse-Transform Method

Inverse-Transform method adalah teknik yang digunakan untuk membangkitkan bilangan acak menggunakan CDF dari suatu peubah acak. Berdasarkan teori Probability Integral Transformation (transformasi integral peluang), teori tersebut menyatakan jika X adalah peubah acak kontinu dengan CDF $F(x)$, maka U = $F(X) \sim \text{Uniform}(0,1)$.
Tahapan Menggunakan Inverse Transform Method

1. Bangkitkan bilangan acak U yang berdistribusi Uniform di antara 0 dan 1, $U(0,1)$.
2. Misalkan $F(x)$ adalah CDF dari suatu distribusi acak.
3. Cari invers dari F, $yaitu\ F^{-1}(u)$
4. Maka bilangan acak $X = F^{-1}(U)$

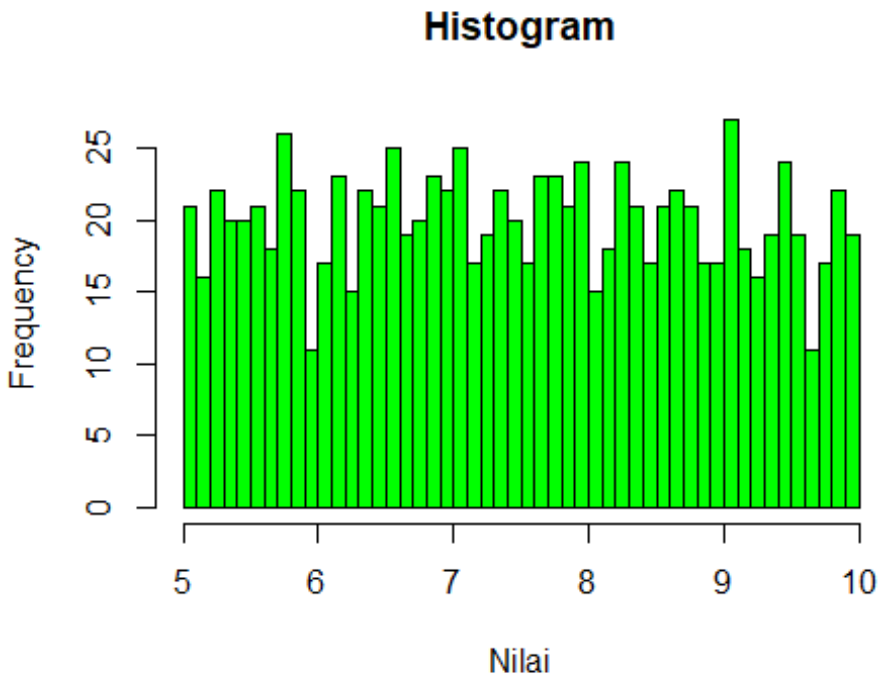
Kasus Kontinu

Misalkan suatu distribusi uniform pada interval $[a, b]$ memiliki fungsi distribusi kumulatif (CDF) : $\mathrm{F(x)} = \frac{\mathrm{x-a}}{\mathrm{b-a}}, \quad \mathrm{x} \in [\mathrm{a}, \mathrm{b}]$. Maka fungsi invers dari CDF tersebut adalah $F^{-1}(u) = a + u \cdot (b - a)$, dengan u adalah bilangan acak dari distribusi $Uniform[0,1]$.
Tentukan 100 sampel dari distribusi tersebut.

```
a <- 5
b <- 10
U <- runif (1000) #membangkitkan variabel acak unfirom dari 0 sampai 1
X <- a + U * (b - a) #apply the inverse CDF
head(X, 10)
```

```
##  [1] 8.772376 8.146106 8.550912 5.003124 7.376583 6.100594 6.899083 8.0638
55
##  [9] 6.758990 5.555677
```

```
#menampilkan histogram
hist(X, breaks = 50, main = "Histogram", xlab = "Nilai", col = "green")
```



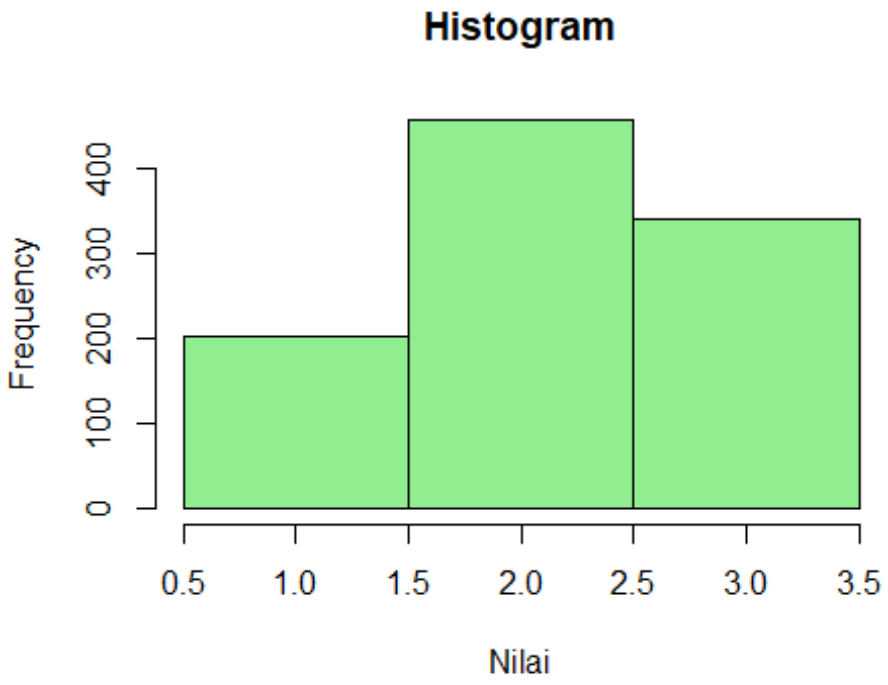
Kode R di atas menunjukkan bagaimana cara membangkitkan data acak dari distribusi uniform kontinu pada interval $[a, b] = [5,\ 10]$ menggunakan metode Inverse Transform. Pertama, ditetapkan nilai batas bawah distribusi $a\ =\ 5$ dan batas atas $b\ =\ 10$. Interval ini menjadi ruang sebaran bagi bilangan acak yang akan dibangkitkan. Selanjutnya, perintah runif(1000) digunakan untuk membangkitkan 1000 bilangan acak dari distribusi $Uniform(0{,}1)$. Bilangan acak tersebut disimpan dalam variabel U. Karena masih berada pada rentang $[0{,}1]$, nilai-nilai U perlu ditransformasikan agar sesuai dengan distribusi uniform pada interval $[5{,}10]$. Transformasi dilakukan dengan menerapkan rumus inverse CDF : $X = a + U \times (b - a)$ dalam kode hal ini ditulis sebagai X <- a + U * (b - a). Hasilnya adalah variabel X yang berisi 1000 bilangan acak yang tersebar merata antara 5 dan 10. Untuk memastikan hasil, fungsi head (X, 10) digunakan untuk menampilkan 10 nilai pertama dari variabel X. Terakhir, distribusi dari sampel acak X divisualisasikan menggunakan fungsi hist(). Histogram dibuat dengan 50 interval (breaks) diberi judul "Histogram" label sumbu-x "Nilai" dan batang histogram berwarna hijau. Grafik ini memperlihatkan sebaran data acak yang relatif rata (flat) sesuai karakteristik distribusi uniform, di mana setiap nilai dalam interval [5,10] memiliki peluang kemunculan yang hampir sama.

Kasus Diskret

Jika X bilangan acak diskret dan $\ldots < x_{i-1} < x_i < x_{i+1} < \cdots$ adalah titik yang diskontinu dari $F_X(x)$, maka inverse transform adalah $F_X^{-1}(u) = x_i$, Dimana $F_X(x_{i-1}) < u < F_X(x_i)$ Misalkan suatu distribusi diskret dengan probabilitas:

$P(X = 1) = 0.2 \Rightarrow P(X \leq 1) = 0.2$
$P(X = 2) = 0.45 \Rightarrow P(X \leq 2) = 0.2 + 0.45 = 0.65$
$P(X = 3) = 0.35 \Rightarrow P(X \leq 3) = 0.65 + 0.35 = 1.0$.
Tentukan 1000 sampelnya!

```
n <- 1000 #jumlah sampel
u <- runif(n)
#probabilitas kumulatif
p <- c(0.2, 0.65, 1.0)
#nilai diskret yang sesuai
x_vals <- c(1, 2, 3)
#fungsi invers CDF
x <- findInterval(u, p, rightmost.closed = TRUE) + 1
#menampilkan histogram
hist (x, breaks = seq(0.5, 3.5, by = 1), main = "Histogram", xlab = "Nilai",
col = "lightgreen", right = TRUE)
```



Kode R di atas digunakan untuk membangkitkan bilangan acak diskret menggunakan metode *Inverse Transform Sampling*. Pertama, ditentukan jumlah sampel sebanyak $n = 1000$ lalu dibangkitkan 1000 bilangan acak dari distribusi uniform $U(0,1)$ dengan fungsi runif(n) dan disimpan dalam variabel u. Selanjutnya, didefinisikan probabilitas kumulatif $p = (0.2,\ 0.65,\ 1.0)$ yang merepresentasikan distribusi kumulatif dari suatu peubah acak diskrit X. Probabilitas ini berhubungan dengan nilai diskret $x_{\text{vals}} = (1,2,3)$. Artinya, peubah acak X mengambil nilai 1 dengan probabilitas 0,2; nilai 2 dengan probabilitas 0,45; dan nilai 3 dengan probabilitas 0,35. Transformasi dilakukan dengan fungsi findInterval(u, p,

rightmost.closed = TRUE) + 1, yaitu menentukan di interval mana setiap bilangan acak uuu jatuh, kemudian mengonversinya menjadi nilai diskrit 1, 2, atau 3 sesuai distribusi yang didefinisikan. Akhirnya, data acak yang terbentuk divisualisasikan menggunakan fungsi hist(). Histogram menampilkan frekuensi kemunculan nilai 1, 2, dan 3 dengan batang berwarna hijau muda. Dari grafik terlihat bahwa distribusi sampel mendekati distribusi teoritis, yaitu sekitar 20% untuk nilai 1, 45% untuk nilai 2, dan 35% untuk nilai 3.

Kasus Model Regresi Linear

Misalkan akan membangkitkan model untuk regresi linear sederhana :

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

Dengan $\left(\varepsilon \sim N(0,1)\right),\ (\beta_0 = 0.81)\ dan\ (\beta_1 = 1.15)$

```
b0 <- 0.81
b1 <- 1.15
b0topi <- NULL
b1topi <- NULL

for(i in 1:300){
  epsilon <- rnorm(10)
  X <- rnorm(10)
  Y <- b0 + b1*X + epsilon
  obj <- lm(Y ~ X)
  b0topi <- c(b0topi, obj$coefficients[1])
  b1topi <- c(b1topi, obj$coefficients[2])
}

hasil <- matrix(
  c(mean(b0topi), sd(b0topi),
    mean(b1topi), sd(b1topi)),
  nrow = 2, ncol = 2
)

rownames(hasil) <- c("mean", "sd")
colnames(hasil) <- c("b0", "b1")
hasil
```

```
##             b0        b1
## mean 0.8141478 1.1707914
## sd   0.3349501 0.3757799
```

Kode R di atas melakukan simulasi untuk mengamati sifat estimasi pada model regresi linear sederhana dengan parameter sebenarnya $\beta_0 = 0.81$ dan $\beta_1 = 1.15$. Proses simulasi dijalankan sebanyak 300 kali di mana pada setiap percobaan dibangkitkan sepuluh data acak untuk variabel X dan sepuluh error acak ϵ keduanya dari distribusi normal standar.

Variabel respon Y kemudian dibentuk sesuai model $Y = \beta_0 + \beta_1 X + \epsilon$ dan estimasi parameter diperoleh menggunakan fungsi $lm(Y \sim X)$. Hasil simulasi yang ditampilkan dalam tabel ringkasan menunjukkan bahwa rata-rata estimasi intersep $\widehat{\beta_0}$ adalah 0.814, sangat dekat dengan nilai sebenarnya 0.81. Demikian pula rata-rata estimasi slope $\widehat{\beta_1}$ sebesar 1.171 juga mendekati nilai sebenarnya 1.15. Nilai simpangan baku masing-masing estimasi yaitu 0.335 untuk intersep dan 0.376 untuk slope menggambarkan tingkat variasi antar-sampel. Dengan demikian hasil ini menunjukkan bahwa metode Ordinary Least Squares (OLS) menghasilkan estimasi yang tidak bias (*unbiased estimator*) karena rata-rata hasil estimasi mendekati parameter sebenarnya. Namun, adanya simpangan baku menegaskan bahwa tetap terdapat variasi dari satu sampel ke sampel lain.

2. Acceptance-rejection Method

Jika CDF suatu peubah acak tidak bisa ditentukan, maka Acceptance-rejection method dapat digunakan untuk membangkitkan bilangan acak. Ide dari ARM adalah ada sebuah distribusi (X) yang akan menjadi target untuk pembangkitan bilangan acak namun sulit untuk dilakukan pembangkitan secara langsung. Sehingga, dibandingkan membangkitkan bilangan acak secara langsung dari distribusi tersebut kita akan membangkitkan bilangan acak dari suatu distribusi $(Y)(proposaldistribution)$yang lebih mudah diimplementasikan. (Y) adalah suatu peubah acak yang nilai anggotanya sama dengan (X). Andaikan (X) dan (Y) adalah dua peubah acak dengan fungsi masing-masing (f) dan (g), terdapat konstanta (c) sehingga $(f(t)g(t) \leq c)$, untuk semua $(t: f(t) > 0)$.
Tahapan acceptance-rejection method

1. Tetapkan peubah acak (Y) sebagai target dari sebaran beserta PDF-nya $\big(g(y)\big)$
2. Bangkitkan peubah acak (Y) tersebut
3. Bangkitkan (u) berdasarkan distribusi $\big(\text{Uniform}(0,1)\big)$
4. Hitung nilai (c) dengan mencari nilai maksimum dari $\left(\frac{f(t)}{g(t)}\right)$
5. Jika $\left(u \leq \frac{f(y)}{cg(y)}\right)$, jadikan (Y) sebagai (X)
6. Jika poin 4 terpenuhi maka kembali ke poin 1

Metode Acceptance-Rejection Menggunakan Distribusi Gamma

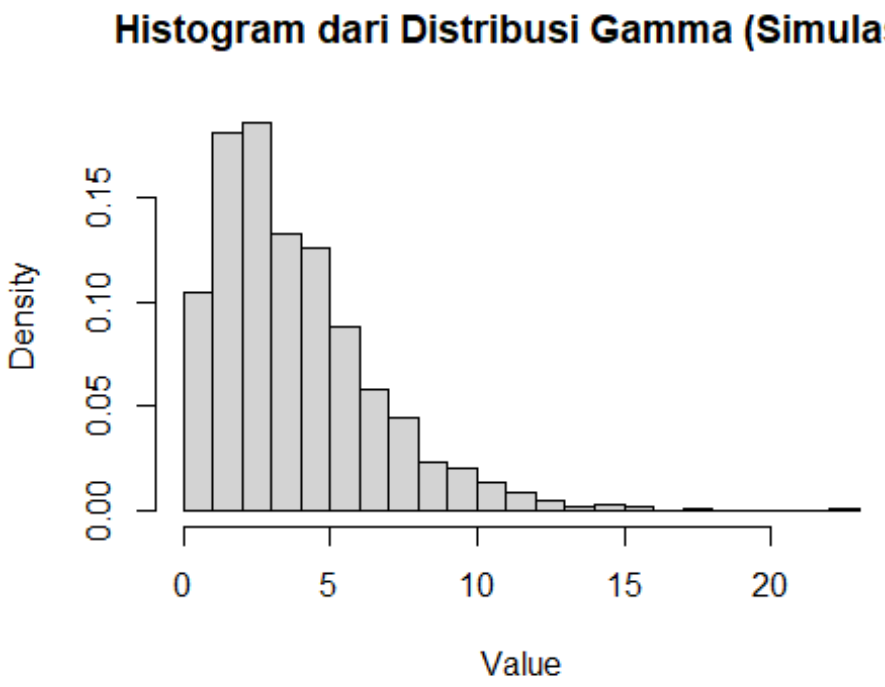
Misalkan ingin menghasilkan sampel dari distribusi Gamma dengan parameter $(\alpha = 2)$ dan $(\beta = 2)$, menggunakan distribusi eksponensial $\big((\lambda = 1)\big)$ sebagai distribusi proposal.

PDF dari distribusi Gamma: $f(x) = \frac{x^{\alpha-1}e^{-x/\beta}}{\beta^{\alpha}\Gamma(\alpha)} = \frac{xe^{-x/2}}{4}$

PDF distribusi proposal (eksponensial): $g(x) = \lambda e^{-\lambda x} = e^{-x}$

```
# parameter untuk distribusi Gamma
alpha <- 2
beta <- 2
```

```r
# parameter untuk distribusi eksponensial
lambda <- 1 / beta

# mencari nilai C
x_values <- seq(0.01, 10, length.out = 1000)
density_gamma_values <- (1 / (gamma(alpha) * beta^alpha)) *
                        (x_values^(alpha - 1)) *
                        exp(-x_values / beta)

density_exponential_values <- lambda * exp(-lambda * x_values)
C <- max(density_gamma_values / density_exponential_values)
n_samples <- 1000

# mempersiapkan vektor untuk menyimpan sampel
samples <- numeric(n_samples)

# metode acceptance-rejection
for (i in 1:n_samples) {
  repeat {
    # menghasilkan sampel dari distribusi proposal (eksponensial)
    x_proposal <- rexp(1, rate = lambda)

    # menghitung densitas dari distribusi Gamma secara manual
    density_gamma <- (1 / (gamma(alpha) * beta^alpha)) *
                     (x_proposal^(alpha - 1)) *
                     exp(-x_proposal / beta)

    # menghasilkan sampel dari distribusi uniform untuk membandingkan
    u <- runif(1, 0, C)  # C sebagai batas atas

    # acceptance-rejection
    if (u < density_gamma / (lambda * exp(-lambda * x_proposal))) {
      samples[i] <- x_proposal
      break
    }
  }
}

# tampilan histogram dari hasil sampel yang dihasilkan
hist(samples, breaks = 30,
     main = "Histogram dari Distribusi Gamma (Simulasi)",
     xlab = "Value", probability = TRUE)
```



Kode R di atas digunakan untuk membangkitkan data acak dari distribusi Gamma dengan parameter $\alpha = 2$ dan $\beta = 2$ menggunakan metode *Acceptance-Rejection*. Distribusi proposal yang dipakai adalah distribusi Eksponensial dengan parameter $\lambda = 1/\beta = 0.5$. Pertama dihitung nilai konstanta C yaitu batas atas untuk membandingkan distribusi Gamma dan Eksponensial. Prosesnya adalah menghasilkan kandidat dari distribusi Eksponensial, menghitung peluang dari distribusi Gamma, lalu mengambil bilangan acak dari $U(0, C)$ untuk menentukan apakah kandidat diterima atau ditolak. Jika memenuhi syarat sampel disimpan jika tidak proses diulang. Dengan cara ini diperoleh 1000 sampel yang mengikuti distribusi Gamma. Hasil akhirnya ditampilkan dalam bentuk histogram, yang menunjukkan sebaran data dengan sumbu-x menunjukkan nilai yang dihasilkan. Histogram ini memperlihatkan bahwa data yang dibangkitkan mengikuti pola distribusi Gamma dengan parameter yang telah ditentukan..

3. Direction Transformation

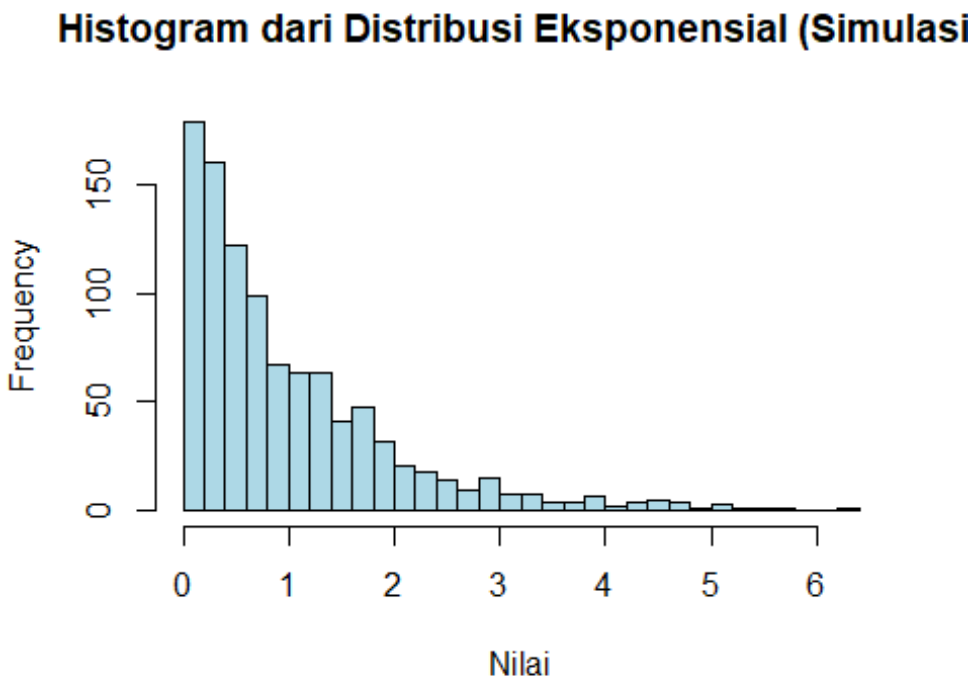
Metode Direct Transformation adalah teknik untuk menghasilkan sampel dari distribusi probabilitas tertentu dengan menggunakan transformasi langsung dari distribusi yang lebih sederhana atau lebih mudah dikelola.

Direction Transformation dari Distribusi Uniform ke Distribusi Eksponensial

Misalkan ingin mentransformasi distribusi uniform ke distribusi eksponensial. Distribusi eksponensial dengan parameter (λ) memiliki fungsi invers CDF : $F^{-1}(u) = -\frac{\ln(1-u)}{\lambda}$ Karena

$(1 - u)$ dan (u) memiliki distribusi yang sama, maka fungsi invers CDF dapat ditulis : $F^{-1}(u) = -\frac{\ln(u)}{\lambda}$

```r
# parameter distribusi eksponensial
lambda <- 1  # parameter laju (rate)

# jumlah sampel yang diinginkan
n <- 1000

# generate bilangan acak dari distribusi uniform [0,1]
u <- runif(n)

# transformasi langsung ke distribusi eksponensial
x <- -log(u) / lambda

# tampilkan histogram dari sampel yang dihasilkan
hist(x, breaks = 30,
     main = "Histogram dari Distribusi Eksponensial (Simulasi)",
     xlab = "Nilai",
     col = "lightblue")
```



Kode R di atas digunakan untuk membangkitkan bilangan acak dari distribusi eksponensial dengan parameter laju $\lambda = 1$ menggunakan metode *Inverse Transform*. Pertama ditentukan jumlah sampel yang ingin dibangkitkan yaitu n = 1000. Kemudian dibangkitkan 1000 bilangan acak dari distribusi Uniform $U(0,1)$ dengan perintah runif(n) dan disimpan ke

dalam variabel u. Selanjutnya dilakukan transformasi ke distribusi eksponensial dengan menggunakan rumus inverse CDF : $X = -\frac{\ln(U)}{\lambda}$ yang dalam kode ditulis sebagai x <- -log(u) / lambda. Dengan transformasi ini diperoleh data acak x yang mengikuti distribusi eksponensial. Akhirnya, hasil sampel divisualisasikan dengan fungsi hist() Histogram dibuat dengan 30 interval diberi judul *"Histogram dari Distribusi Eksponensial (Simulasi)"*, label sumbu-x "Nilai", dan batang berwarna biru muda. Grafik tersebut memperlihatkan pola khas distribusi eksponensial yaitu frekuensi tertinggi di dekat nol dan menurun seiring bertambahnya nilai.