

**PROGRAM STUDI MATEMATIKA**

FAKULTAS SAINS DAN MATEMATIKA

UNIVERSITAS KRISTEN SATYA WACANA

Jl. Diponegoro 52-60 Salatiga 50715 Jawa Tengah, Email : [mathfsm@uksw.edu](mailto:mathfsm@uksw.edu)

**LAPORAN PRAKTIKUM**

Mata Kuliah : Risk Analysis

Materi : Diary Univariate

Nama : Muhammad Anas Rifai

Nim : 662021003

**Penjelasan Kode:**

1. **x = seq(-7.5, 7, by=0.01)**:

Kode ini membuat sebuah urutan nilai x dari -7.5 hingga 7 dengan kenaikan sebesar 0.01. Urutan ini digunakan sebagai sumbu x untuk memplot distribusi normal.

1. **u = dnorm(x)**:

Fungsi dnorm(x) digunakan untuk menghitung nilai distribusi normal standar dengan rata-rata 0 dan standar deviasi 1 di setiap titik x. Hasil ini disimpan dalam variabel u.

1. **v = dnorm(x,1,2)**:

Fungsi dnorm(x,1,2) menghitung nilai distribusi normal dengan rata-rata 1 dan standar deviasi 2 di setiap titik x. Hasilnya disimpan dalam variabel v.

1. **plot(x,u,type="l")**:

Plot distribusi normal standar (N(0,1)) diplot dengan menggunakan nilai x dan u. Opsi type="l" digunakan untuk membuat grafik garis.

1. **lines(x,v,lty=3)**:

Menambahkan distribusi normal dengan rata-rata 1 dan standar deviasi 2 (N(1,4)) ke dalam plot yang sama dengan garis putus-putus (lty=3).

1. **dnorm(0)**:

Menghitung nilai fungsi densitas probabilitas pada titik 0 untuk distribusi normal standar (N(0,1)), yaitu 0.3989423, yang merupakan titik puncak dari distribusi ini.

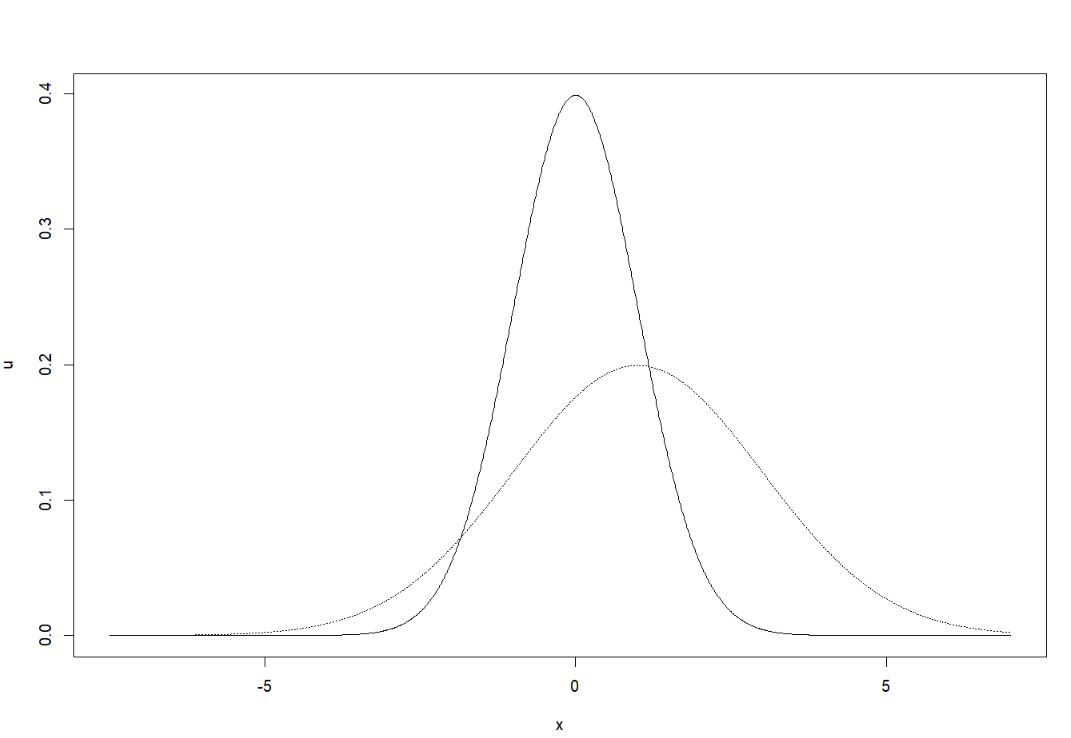
1. **dnorm(1,1,2)**:

Menghitung nilai fungsi densitas probabilitas pada titik 1 untuk distribusi normal N(1,4), yaitu 0.1994711, yang juga merupakan titik puncak dari distribusi ini.

1. **1/sqrt(2\*pi)** dan **1/sqrt(8\*pi)**:

Kedua perhitungan ini merupakan cara manual untuk menghitung tinggi puncak distribusi normal pada titik pusat distribusi masing-masing. Nilai 1/sqrt(2\*pi) (0.3989423) sesuai dengan distribusi N(0,1), sedangkan 1/sqrt(8\*pi) (0.1994711) sesuai dengan distribusi N(1,4).

**Hasil:**

****

* Grafik yang dihasilkan menunjukkan dua distribusi normal. Distribusi N(0,1) ditampilkan sebagai garis penuh, sementara distribusi N(1,4) ditampilkan dengan garis putus-putus.
* Distribusi normal standar (N(0,1)) memiliki puncak di sekitar 0.3989423 pada titik x = 0, sedangkan distribusi N(1,4) memiliki puncak di sekitar 0.1994711 pada titik x = 1.

**Penjelasan Kode:**

1. **simulasi = function(n,B)**:

Fungsi ini mendefinisikan simulasi yang akan dijalankan, di mana n adalah jumlah data yang dihasilkan dalam satu iterasi, dan B adalah jumlah iterasi atau simulasi yang dilakukan. Fungsi ini menghasilkan sebuah vektor berisi probabilitas dari nilai data yang kurang dari 0 dalam setiap iterasi.

1. **has = numeric(B)**:

Membuat sebuah vektor kosong has dengan panjang B, yang akan menyimpan hasil dari setiap iterasi simulasi.

1. **for (i in 1:B)**:

Loop for digunakan untuk menjalankan simulasi sebanyak B kali.

1. **x = rnorm(n,0.1,0.2)**:

Pada setiap iterasi, data acak x dihasilkan dari distribusi normal dengan rata-rata 0.1 dan standar deviasi 0.2, sebanyak n titik data.

1. **has[i] = sum(x<0)/n**:

Di sini, untuk setiap iterasi, dihitung berapa proporsi nilai dari data x yang kurang dari 0, lalu hasilnya disimpan dalam elemen ke-i dari vektor has.

1. **has**:

Setelah seluruh iterasi selesai, fungsi mengembalikan vektor has yang berisi proporsi nilai x yang kurang dari 0 di setiap iterasi.

1. **hasil = simulasi(100, 10000)**:

Menjalankan simulasi dengan 100 titik data per iterasi (n = 100), dan total 10.000 iterasi (B = 10000).

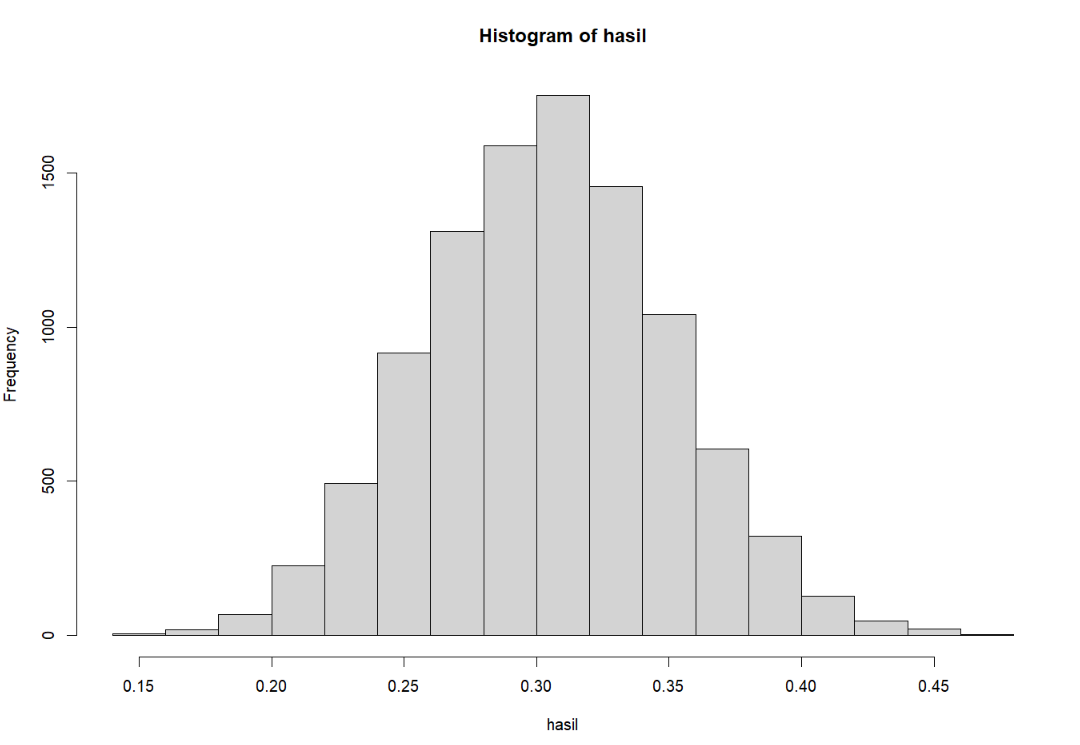
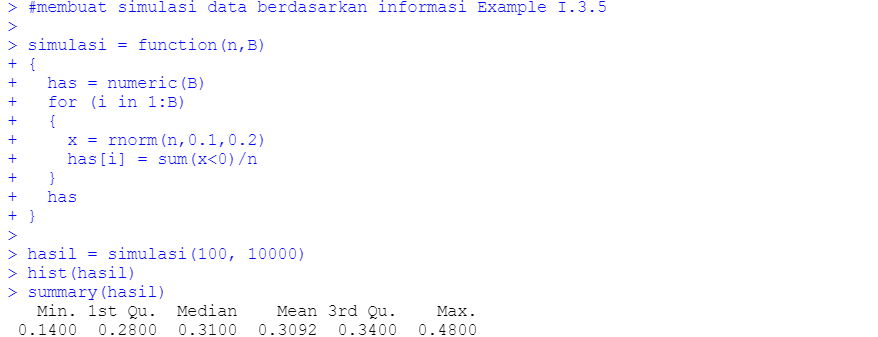
1. **hist(hasil)**:

Membuat histogram dari hasil simulasi untuk melihat distribusi proporsi nilai x yang kurang dari 0 pada tiap simulasi.

1. **summary(hasil)**:

Menampilkan ringkasan statistik (termasuk nilai minimum, kuartil, median, rata-rata, dan maksimum) dari hasil simulasi.

**Hasil:**

****

* Histogram yang dihasilkan menggambarkan distribusi proporsi nilai data yang kurang dari 0 di setiap iterasi. Ini memberikan gambaran visual tentang seberapa sering nilai x kurang dari 0 dalam distribusi yang memiliki rata-rata 0.1 dan standar deviasi 0.2.
* Ringkasan statistik (summary) memberikan informasi lebih detail tentang distribusi proporsi ini, seperti median, mean, serta variasi dari hasil simulasi.

Contoh tambahan :



**Penjelasan Kode:**

1. **simulasi = function(n)**:

Fungsi ini mendefinisikan simulasi yang menghasilkan data berdasarkan distribusi normal untuk dua variabel x dan y, kemudian menghitung kombinasi linier dari keduanya.

1. **x = rnorm(n,0.24, 0.2)**:

Menghasilkan n titik data dari distribusi normal dengan rata-rata 0.24 dan standar deviasi 0.2, disimpan dalam variabel x.

1. **y = rnorm(n,0.16, 0.1)**:

Menghasilkan n titik data dari distribusi normal dengan rata-rata 0.16 dan standar deviasi 0.1, disimpan dalam variabel y.

1. **has = 0.25\*x + 0.75\*y**:

Menghitung kombinasi linier dari x dan y dengan koefisien 0.25 dan 0.75 masing-masing. Hasil dari kombinasi ini disimpan dalam variabel has.

1. **has**:

Mengembalikan vektor has yang berisi hasil kombinasi linier dari x dan y.

1. **hasil = simulasi(10000)**:

Menjalankan simulasi dengan 10.000 titik data.

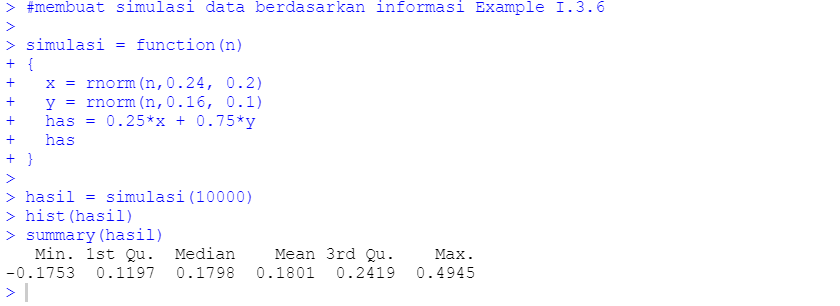
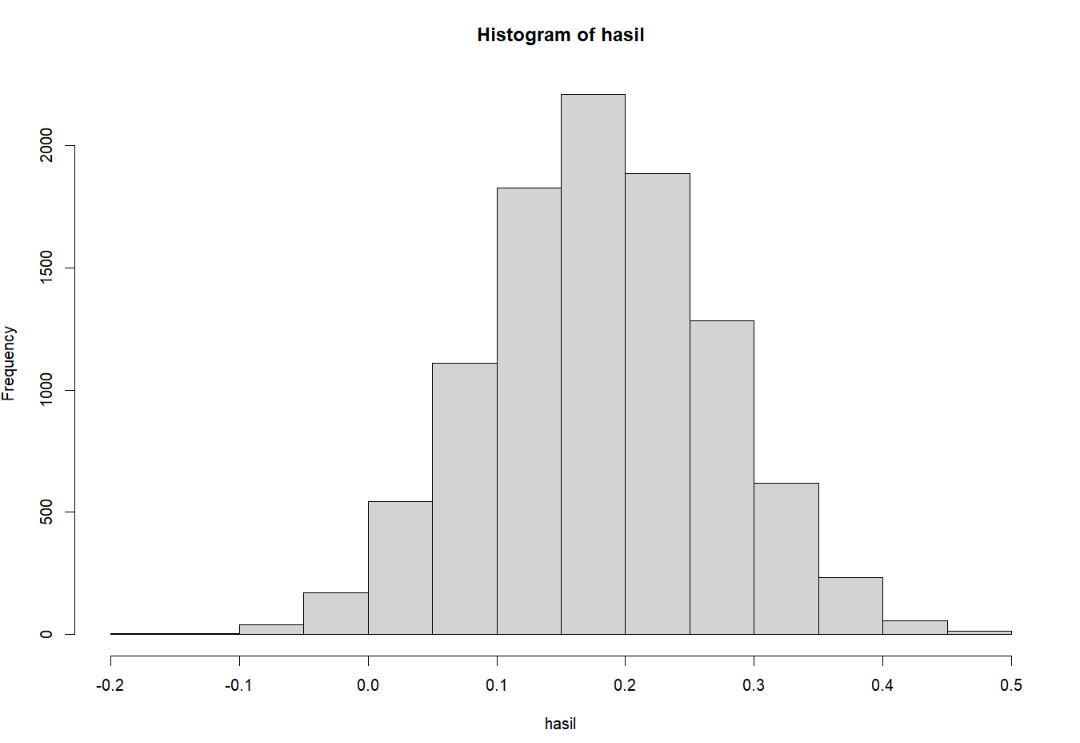
1. **hist(hasil)**:

Membuat histogram dari hasil simulasi has untuk melihat distribusi dari kombinasi linier data x dan y.

1. **summary(hasil)**:

Menampilkan ringkasan statistik dari hasil simulasi has, termasuk nilai minimum, kuartil, median, rata-rata, dan maksimum.

**Hasil:**

**** 

* Ringkasan statistik (summary) memberikan informasi lebih lanjut tentang distribusi nilai-nilai has, seperti median, rata-rata, dan sebaran nilai.
* Histogram yang dihasilkan menunjukkan distribusi dari kombinasi linier 0.25\*x + 0.75\*y, memberikan gambaran visual tentang bagaimana hasil dari kombinasi ini tersebar.

**Penjelasan Kode:**

1. **simulasi = function(n,B)**:

Fungsi ini mendefinisikan simulasi untuk menghitung probabilitas return negatif dalam B iterasi, di mana n adalah jumlah data yang dihasilkan dalam satu iterasi.

1. **has = numeric(B)**:

Membuat vektor kosong has dengan panjang B, yang akan menyimpan hasil probabilitas return negatif pada setiap iterasi simulasi.

1. **for (i in 1:B)**:

Loop for digunakan untuk menjalankan simulasi sebanyak B kali.

1. **R = rnorm(n, 0.18, 0.109)**:

Pada setiap iterasi, data acak R dihasilkan dari distribusi normal dengan rata-rata 0.18 dan standar deviasi 0.109. Nilai ini mensimulasikan return dari sebuah investasi.

1. **has[i] = sum(R<0)/n**:

Menghitung proporsi return negatif, yaitu jumlah elemen R yang kurang dari 0, kemudian membagi dengan n untuk mendapatkan probabilitas return negatif pada iterasi ke-i.

1. **has**:

Setelah seluruh iterasi selesai, fungsi mengembalikan vektor has yang berisi probabilitas return negatif di setiap iterasi.

1. **hasil = simulasi(100, 10000)**:

Menjalankan simulasi dengan 100 titik data per iterasi (n = 100), dan total 10.000 iterasi (B = 10000).

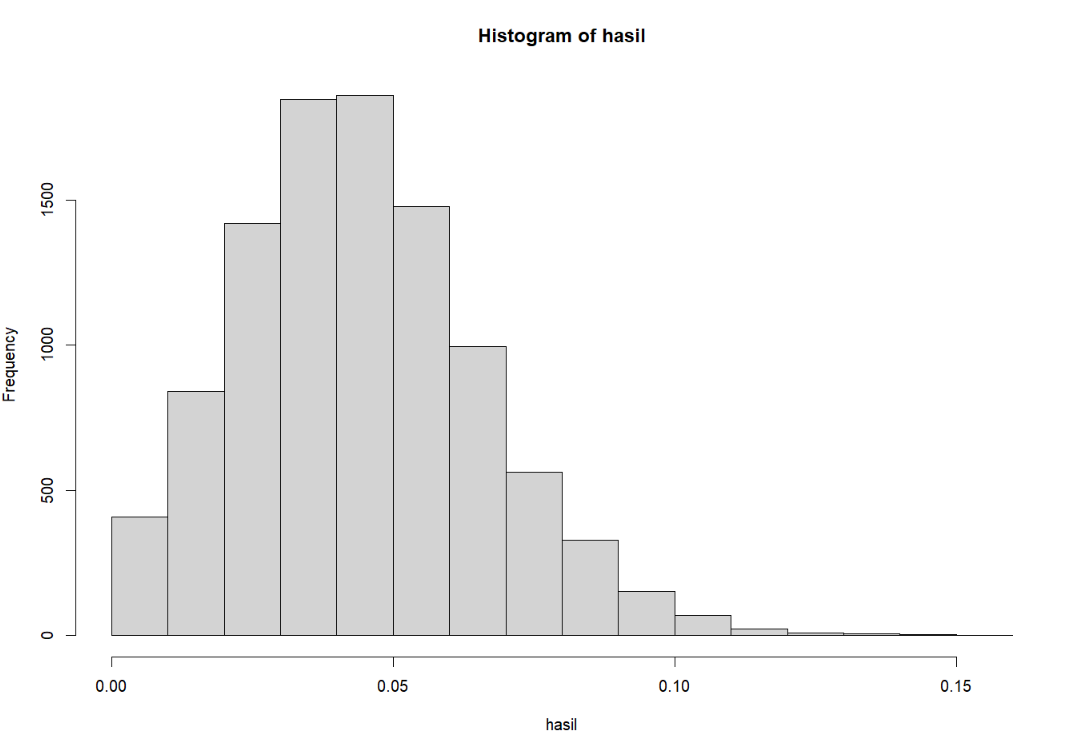
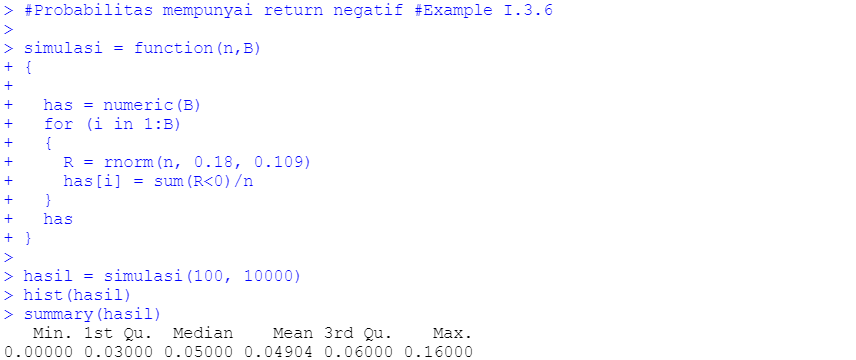
1. **hist(hasil)**:

Membuat histogram dari hasil simulasi untuk melihat distribusi probabilitas return negatif pada tiap simulasi.

1. **summary(hasil)**:

Menampilkan ringkasan statistik dari hasil simulasi, termasuk nilai minimum, kuartil, median, rata-rata, dan maksimum dari probabilitas return negatif.

**Hasil:**

****

* **Histogram** menunjukkan distribusi dari probabilitas return negatif pada tiap iterasi simulasi. Ini memberi gambaran tentang frekuensi dan sebaran probabilitas return negatif.
* **Ringkasan statistik (summary)** memberikan informasi seperti rata-rata, median, serta nilai minimum dan maksimum dari probabilitas return negatif yang dihasilkan dari simulasi.

LANJUTAN :

