Modul 4 Proses Poisson

Indah Suciati

2025-10-15

Counting Process

Counting Process (proses penghitungan) adalah proses stokastik yang digunakan untuk menggambarkan jumlah kumulatif kejadian yang telah terjadi sampai waktu tertentu. Misalnya, banyaknya kendaraan yang melintasi jalan tol setiap jam, jumlah panggilan telepon yang diterima oleh pusat layanan pelanggan dalam satu menit, atau jumlah gempa kecil yang terjadi dalam satu hari. Secara matematis, *counting process* dilambangkan dengan $N(t)$, yaitu banyaknya kejadian yang telah terjadi sampai waktu t. Pada waktu awal, belum ada kejadian sehingga $N(0) = 0$. Nilai $N(t)$ selalu meningkat atau tetap (tidak pernah menurun), setiap kali terjadi satu kejadian, nilai $N(t)$ bertambah satu. Jika digambarkan dalam grafik, proses ini seperti garis tangga (*step function*) yang naik satu langkah setiap kali ada kejadian baru.

Counting process memiliki dua sifat penting yaitu inkremen stasioner dan inkremen independen. Inkremen stasioner merupakan jumlah kejadian dalam suatu selang waktu yang hanya bergantung pada panjang selang tersebut, bukan pada posisi waktunya. Misalnya, jumlah pelanggan antara menit ke-(0–10) dan menit ke-(50–60) memiliki peluang yang sama jika prosesnya stasioner. Inkremen independen merupakan jumlah kejadian dalam dua interval waktu yang tidak tumpang tindih saling bebas, misalnya kejadian pada jam pertama tidak memengaruhi jam berikutnya.

Proses Poisson

Proses Poisson merupakan salah satu bentuk khusus dari *counting process* yang digunakan untuk memodelkan banyaknya kejadian acak yang terjadi dalam suatu interval waktu tertentu. Proses ini sering digunakan ketika kejadian-kejadian terjadi secara acak, saling independen, dan dengan rata-rata laju kejadian yang konstan. Dalam proses Poisson, jumlah kejadian hingga waktu t dinyatakan sebagai $N(t)$, yang merepresentasikan total kejadian yang telah terjadi sejak waktu awal $t = 0$ hingga waktu tersebut. Pada saat awal belum ada kejadian sehingga $N(0) = 0$ dan karena setiap kejadian hanya menambah jumlahnya, maka $N(t)$ bersifat *non-decreasing* (tidak pernah berkurang). Secara matematis, suatu proses stokastik $\{N(t),\ t \geq 0\}$ dikatakan mengikuti Proses Poisson dengan parameter laju $\lambda > 0$ apabila memenuhi tiga sifat utama, yaitu: (1) $N(0) = 0$; (2) memiliki inkremen independen; dan (3) memiliki inkremen stasioner. Dengan sifat tersebut, jumlah kejadian dalam interval waktu t mengikuti distribusi Poisson dengan parameter λt, yang dirumuskan sebagai:

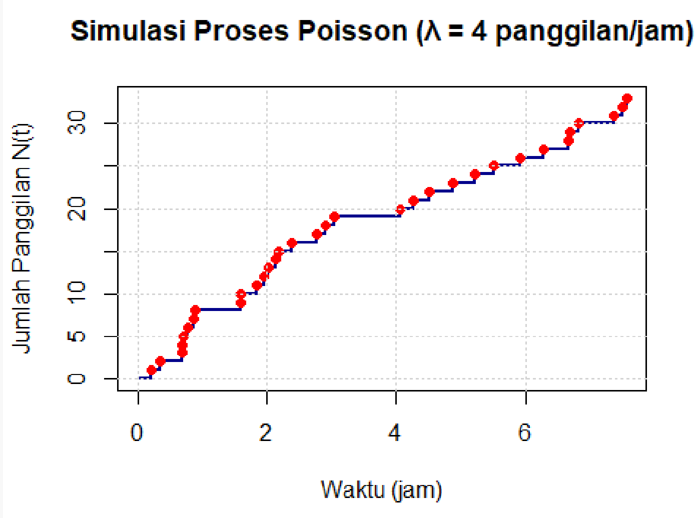
$$P(N(t) = n) = \frac{e^{-\lambda t}.(\lambda t)^n}{n!},\ n \geq 0.$$

Nilai harapan (rata-rata) dan varians dari proses ini sama, yaitu $E(N(t)\,) = Var(N(t)\,) = \lambda t$. Selain itu, selang waktu antar dua kejadian berturut-turut dalam proses Poisson mengikuti distribusi eksponensial dengan parameter yang sama yaitu λ, yang menggambarkan sifat *memoryless* bahwa waktu tunggu sampai kejadian berikutnya tidak bergantung pada kapan kejadian terakhir terjadi. Karena karakteristiknya yang sederhana dan realistis, proses Poisson banyak digunakan untuk memodelkan fenomena acak seperti jumlah pelanggan datang ke toko, jumlah panggilan masuk ke pusat layanan, atau jumlah notifikasi yang diterima ponsel dalam satu hari.

Contoh Simulasi Proses Poisson

Dalam modul ini akan dilakukan simulasi proses Poisson untuk memodelkan kedatangan panggilan telepon pada sebuah pusat layanan pelanggan (*call center*). Diketahui bahwa rata-rata terdapat 4 panggilan masuk setiap jam dan pengamatan dilakukan selama 8 jam kerja. Proses kedatangan panggilan ini diasumsikan terjadi secara acak, independen, dan dengan laju kejadian konstan sepanjang waktu, sehingga dapat dimodelkan menggunakan proses Poisson dengan parameter laju $\lambda = 4$ panggilan per jam.

```
# SIMULASI PROSES POISSON
set.seed(123)   # agar hasil replikasi konsisten
lambda <- 4     # rata-rata panggilan per jam
Tmax <- 8       # waktu pengamatan (jam)

# 1. Simulasi waktu antar kejadian (distribusi eksponensial)
interarrival <- rexp(200, rate = lambda)

# 2. Hitung waktu kumulatif
arrival_time <- cumsum(interarrival)

# 3. Ambil kejadian yang terjadi sebelum Tmax
arrival_time <- arrival_time[arrival_time <= Tmax]

# 4. Buat data frame untuk proses penghitungan
count_data <- data.frame(Time = c(0, arrival_time),
  N_t = 0:length(arrival_time)
)

# 5. Visualisasi Counting Process
plot(count_data$Time, count_data$N_t, type = "s",
     main = "Simulasi Proses Poisson (λ = 4 panggilan/jam)",
     xlab = "Waktu (jam)", ylab = "Jumlah Panggilan N(t)",
     col = "darkblue", lwd = 2)
points(arrival_time, 1:length(arrival_time), pch = 19, col = "red")
grid()

# 6. Total panggilan selama 8 jam
cat("Total panggilan selama", Tmax, "jam =", length(arrival_time), "\n")

## Total panggilan selama 8 jam = 33
```



Kode ini mensimulasikan proses Poisson dengan rata-rata kedatangan $\lambda = 4$ panggilan per jam selama 8 jam pengamatan. Pertama, waktu antar kejadian (*interarrival*) dihasilkan menggunakan distribusi eksponensial dengan `rexp()`, karena dalam proses Poisson, jarak antar kejadian mengikuti distribusi eksponensial. Waktu kedatangan kumulatif dihitung dengan `cumsum()` dan hanya kejadian sebelum 8 jam yang diambil. Data kemudian disusun dalam bentuk *counting process* $(N(t))$, dimana setiap waktu kedatangan ditandai dengan jumlah panggilan yang telah terjadi. Hasil divisualisasikan menggunakan plot step `(type="s")` dengan titik merah menunjukkan setiap panggilan yang terjadi, sehingga terlihat bagaimana jumlah panggilan meningkat seiring waktu. Dari simulasi ini, diperoleh hasil bahwa total panggilan selama 8 jam sebanyak 33 panggilan, hasil ini mendekati nilai rata-rata teoretis $\lambda \times Tmax = 4 \times 8 = 32$. Hal ini menunjukkan bahwa simulasi konsisten dengan karakteristik proses Poisson. Histogram atau plot ini membantu memahami fluktuasi acak dari jumlah kejadian dalam proses Poisson serta konsep *counting process* $N(t)$.

Contoh Perhitungan Kasus Proses Poisson

Sebuah stasiun pengisian bahan bakar umum (SPBU) yang berlokasi di pinggiran kota melakukan pencatatan terhadap kedatangan kendaraan selama jam operasional. Berdasarkan hasil pengamatan sebelumnya, diketahui bahwa rata-rata terdapat 6 kendaraan datang setiap 10 menit untuk melakukan pengisian bahan bakar. Aktivitas kendaraan yang datang ini diasumsikan terjadi secara acak dan saling independen, sehingga dapat dimodelkan menggunakan proses Poisson. Manajer SPBU ingin memahami pola kedatangan kendaraan agar dapat mengatur jumlah petugas dan antrean dengan lebih efisien. Untuk itu, dilakukan beberapa analisis probabilistik sebagai berikut: (1) Berapakah peluang tepat 8 kendaraan datang dalam waktu 10 menit? (2) Berapakah peluang kurang dari 3 kendaraan datang dalam waktu 5 menit? (3) Berapakah peluang antara 5 hingga 10 kendaraan datang dalam waktu 15 menit? (4) Berapakah peluang lebih dari 12 kendaraan datang dalam waktu 20 menit? (5) Selain itu, berapakah nilai harapan, varians, dan simpangan baku dari jumlah kendaraan yang datang selama 20 menit pengamatan?

```r
# CONTOH PERHITUNGAN KASUS PROSES POISSON
lambda_10 <- 6   # rata-rata kendaraan tiap 10 menit

# 1. Peluang tepat 8 kendaraan dalam 10 menit
p_tepat8_10 <- dpois(8, lambda_10)

# 2. Peluang kurang dari 3 kendaraan dalam 5 menit
lambda_5 <- lambda_10 * (5/10)
p_kurang3_5 <- ppois(2, lambda_5, lower.tail = TRUE)

# 3. Peluang antara 5–10 kendaraan dalam 15 menit
lambda_15 <- lambda_10 * (15/10)
p_antara5_10_15 <- ppois(10, lambda_15) - ppois(4, lambda_15)

# 4. Peluang lebih dari 12 kendaraan dalam 20 menit
lambda_20 <- lambda_10 * (20/10)
p_lebih12_20 <- ppois(12, lambda_20, lower.tail = FALSE)

# 5. Nilai harapan, varians, dan simpangan baku untuk 20 menit
mean_20 <- lambda_20
var_20 <- lambda_20
sd_20 <- sqrt(lambda_20)

# FORMAT ANGKA UNTUK OUTPUT
format_angka <- function(x) {
  if (abs(x - round(x)) < 1e-8) {
    return(as.character(round(x)))   # bilangan bulat
  } else {
    return(format(round(x, 4), nsmall = 4))  # 4 desimal
  }
}

# TABEL RINGKASAN HASIL
hasil_poisson <- data.frame(
  Keterangan = c(
    "P(X = 8) kendaraan dalam 10 menit",
    "P(X < 3) kendaraan dalam 5 menit",
    "P(5 ≤ X ≤ 10) kendaraan dalam 15 menit",
    "P(X > 12) kendaraan dalam 20 menit",
    "Nilai harapan E[X] untuk 20 menit",
    "Varians Var[X] untuk 20 menit",
    "Simpangan baku SD[X] untuk 20 menit"
  ),
  Hasil = sapply(
    c(p_tepat8_10, p_kurang3_5, p_antara5_10_15, p_lebih12_20, mean_20,
var_20, sd_20),
    format_angka
  )
```

```
)
print(hasil_poisson, row.names = FALSE)
```

```
##                              Keterangan  Hasil
##       P(X = 8) kendaraan dalam 10 menit 0.1033
##        P(X < 3) kendaraan dalam 5 menit 0.4232
##  P(5 ≤ X ≤ 10) kendaraan dalam 15 menit 0.6510
##      P(X > 12) kendaraan dalam 20 menit 0.4240
##       Nilai harapan E[X] untuk 20 menit     12
##           Varians Var[X] untuk 20 menit     12
##     Simpangan baku SD[X] untuk 20 menit 3.4641
```

Kode R di atas merupakan contoh perhitungan peluang dan statistik dasar pada proses Poisson terkait jumlah kendaraan yang lewat dalam interval waktu tertentu. Rata-rata kedatangan kendaraan ditetapkan sebesar 6 kendaraan per 10 menit. Untuk menghitung beberapa peluang, digunakan fungsi `dpois()` dan `ppois()`. Selain probabilitas, kode di atas juga menghitung nilai harapan, varians, dan simpangan baku untuk interval waktu 20 menit, kemudian hasil ini disusun dalam tabel ringkasan agar lebih jelass.

Pemisahan atau Penyaringan Proses Poisson (Thinning Process)

Secara konsep, *thinning process* dilakukan dengan memberikan probabilitas tertentu kepada setiap kejadian untuk masuk ke kategori tertentu. Misalnya, setiap pelanggan yang datang memiliki peluang p untuk melakukan pembelian dan peluang $(1-p)$ untuk tidak membeli. Jika kedatangan pelanggan mengikuti proses Poisson dengan laju λ orang per jam, maka pelanggan pembeli akan membentuk proses Poisson baru dengan laju $\lambda_1 = p\lambda$ dan pelanggan yang tidak membeli juga membentuk proses Poisson baru dengan laju $\lambda_2 = (1-p)\lambda$. Kedua proses hasil pemisahan ini tetap merupakan proses Poisson, dan keduanya independen satu sama lain. Artinya, terjadinya pembeli tidak memengaruhi kapan non-pembeli datang, dan sebaliknya.

Secara matematis, jika $N(t)$ adalah proses Poisson dengan parameter laju λ, dan setiap kejadian diklasifikasikan menjadi dua tipe berdasarkan probabilitas p dan $(1-p)$, maka:

$$N_1(t) \sim \text{Poisson}(p\lambda t)$$

$$N_2(t) \sim \text{Poisson}((1-p)\lambda t)$$

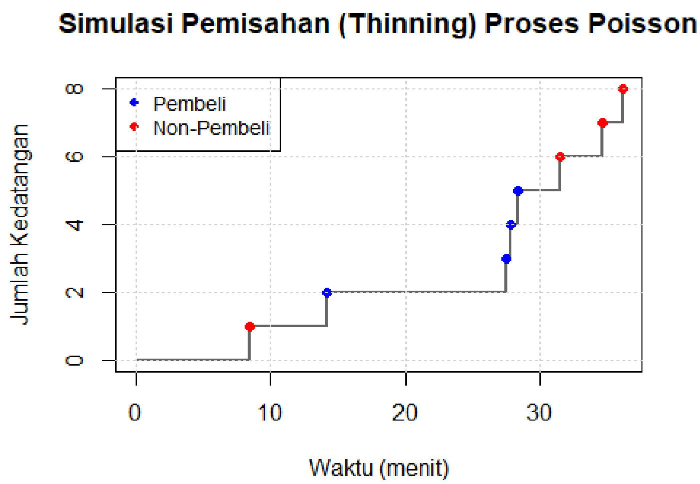
Kedua proses ini independen, dan jumlah totalnya tetap memenuhi $N(t) = N_1(t) + N_2(t)$.

Dalam praktiknya, *thinning process* banyak digunakan untuk memodelkan sistem di mana kejadian memiliki jenis atau kategori tertentu. Misalnya dalam *call center* panggilan masuk bisa dibedakan menjadi pertanyaan umum dan keluhan pelanggan, dalam pabrik kejadian kerusakan mesin bisa diklasifikasikan sebagai kerusakan ringan atau berat, dan dalam lalu lintas, kendaraan yang lewat bisa dikategorikan berdasarkan jenisnya, seperti mobil pribadi atau truk. Dalam RStudio, simulasi proses ini bisa dilakukan dengan dua langkah utama yaitu mensimulasikan total kejadian dari proses Poisson menggunakan `rpois()` dan mengklasifikasikan setiap kejadian ke dalam kategori menggunakan `rbinom()` berdasarkan probabilitas p.

Contoh Simulasi Thinning Process

Dalam modul ini akan dilakukan simulasi proses Poisson untuk memodelkan kedatangan pelanggan pada suatu toko dengan rata-rata $\lambda = 1/10$ pelanggan per menit selama 60 menit. Setiap pelanggan yang datang akan diklasifikasikan secara acak ke dalam dua kategori, yaitu pembeli dengan probabilitas $p = 0.4$ dan non-pembeli dengan probabilitas $1 - p = 0.6$. Simulasi ini bertujuan untuk memperlihatkan bagaimana satu proses Poisson dapat dipisahkan (*thinning*) menjadi dua sub-proses Poisson independen, serta divisualisasikan dalam bentuk grafik langkah (*counting process*) untuk menunjukkan waktu kedatangan masing-masing kategori pelanggan.

```r
# SIMULASI THINNING PROCESS
set.seed(123)
lambda <- 1/10
Tmax <- 60
p <- 0.4

# Simulasi waktu antar kejadian
waktu_antar <- rexp(200, rate = lambda)
waktu_total <- cumsum(waktu_antar)
waktu_total <- waktu_total[waktu_total <= Tmax]

# Tentukan tipe (1 = pembeli, 0 = non-pembeli)
tipe <- rbinom(length(waktu_total), 1, p)

# Pisahkan waktu berdasarkan tipe
waktu_pembeli <- waktu_total[tipe == 1]
waktu_non <- waktu_total[tipe == 0]

# Hitung indeks (urutan) dalam proses total
idx_pembeli <- which(tipe == 1)
idx_non <- which(tipe == 0)

# Plot hasil
plot(c(0, waktu_total), 0:length(waktu_total), type = "s",
     main = "Simulasi Pemisahan (Thinning) Proses Poisson",
     xlab = "Waktu (menit)", ylab = "Jumlah Kedatangan",
     col = "gray40", lwd = 2)
points(waktu_pembeli, idx_pembeli, col = "blue", pch = 19)
points(waktu_non, idx_non, col = "red", pch = 19)
legend("topleft", legend = c("Pembeli", "Non-Pembeli"),
       col = c("blue", "red"), pch = 19, cex = 0.9)
grid()
```



Kode di atas mensimulasikan proses kedatangan pelanggan yang mengikuti proses Poisson kontinu dengan rata-rata satu pelanggan setiap 10 menit $\lambda = 1/10$. Waktu antar kedatangan pelanggan dihasilkan menggunakan fungsi `rexp()` yang berdistribusi eksponensial, karena proses Poisson memiliki sifat bahwa selang waktu antar kejadian (*interarrival time*) bersifat eksponensial. Hasil simulasi berupa deretan waktu kumulatif (`waktu_total`) yang menunjukkan kapan setiap pelanggan tiba selama periode pengamatan 60 menit. Selanjutnya, setiap pelanggan yang datang diklasifikasikan secara acak menggunakan fungsi `rbinom()`, dengan probabilitas $p = 0.4$ untuk menjadi pembeli dan $1 - p = 0.6$ untuk menjadi non-pembeli. Visualisasi hasil simulasi dilakukan menggunakan grafik langkah (*counting process*). Garis abu-abu menggambarkan jumlah kumulatif kedatangan seluruh pelanggan (proses total), sedangkan titik biru menunjukkan kejadian pembeli, dan titik merah mewakili non-pembeli. Setiap titik berada tepat di tangga kenaikan garis abu-abu karena semua sub-kejadian berasal dari proses Poisson yang sama. Distribusi posisi titik biru dan merah tampak acak, sesuai sifat proses Poisson yang memiliki inkremen independent. Dari hasil grafik terlihat bahwa semakin besar nilai probabilitas p, semakin banyak titik biru (pembeli) yang muncul. Sebaliknya, jika p kecil maka titik merah (non-pembeli) lebih dominan. Kedua sub-proses tersebut tetap bersifat Poisson dan saling independen, karena setiap kejadian diklasifikasikan secara acak tanpa memengaruhi kejadian lainnya.

Contoh Perhitungan Kasus *Thinning Process*

Dalam satu hari, sebuah ponsel rata-rata menerima 50 notifikasi dari berbagai aplikasi. Namun, jenis notifikasi yang masuk berbeda-beda, 20% berasal dari grup *chat* (misalnya WhatsApp, Telegram), 50% dari media sosial (Instagram, TikTok, X, dll.), 30% dari email dan aplikasi kerja. Asumsikan notifikasi datang secara acak dan independen satu sama lain. Tentukan rata-rata, varians, dan standar deviasi untuk masing-masing sumber notifikasi, hitung peluang bahwa jumlah notifikasi media sosial lebih dari 30 kali, hitung peluang bahwa jumlah notifikasi grup *chat* berada antara 5–15 kali, hitung peluang bahwa jumlah notifikasi email & kerja kurang dari 10 kali, dan visualisasikan kurva PDF dan CDF jumlah notifikasi media sosial.

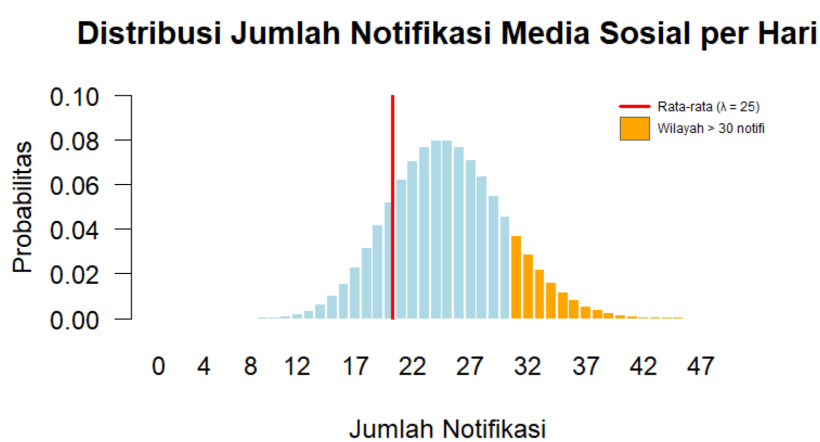
```r
# CONTOH PERHITUNGAN KASUS THINNING PROCESS
# Kasus: Notifikasi Ponsel Selama Sehari
lambda_total <- 50   # rata-rata total notifikasi per hari

# Probabilitas masing-masing jenis
p_chat <- 0.2
p_social <- 0.5
p_work <- 0.3

# Laju sub-proses
lambda_chat <- lambda_total * p_chat
lambda_social <- lambda_total * p_social
lambda_work <- lambda_total * p_work

# PERHITUNGAN DASAR
mean_chat <- lambda_chat
mean_social <- lambda_social
mean_work <- lambda_work

var_chat <- lambda_chat
var_social <- lambda_social
var_work <- lambda_work

sd_chat <- sqrt(var_chat)
sd_social <- sqrt(var_social)
sd_work <- sqrt(var_work)

# MENCARI NILAI PELUANG
# 1. Peluang media sosial > 30
p_social_lebih30 <- ppois(30, lambda_social, lower.tail = FALSE)

# 2. Peluang grup chat antara 5–15
p_chat_antara5_15 <- ppois(15, lambda_chat) - ppois(4, lambda_chat)

# 3. Peluang kerja < 10
p_work_kurang10 <- ppois(9, lambda_work, lower.tail = TRUE)

# FUNGSI UNTUK FORMAT ANGKA
format_angka <- function(x) {
  if (abs(x - round(x)) < 1e-8) {
    return(as.character(round(x)))   # bilangan bulat tanpa koma
  } else {
    return(format(round(x, 2), nsmall = 2))  # desimal 2 angka
  }
}

# TABEL RINGKASAN HASIL
hasil <- data.frame(
```

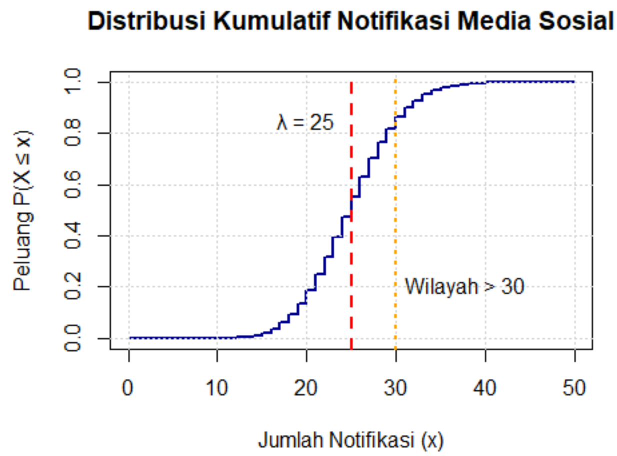
```r
  Keterangan = c(
    "P(X_social > 30)","P(X_chat antara 5-15)","P(X_work < 10)",
    "Mean notifikasi chat","Mean notifikasi media sosial",
    "Mean notifikasi kerja","Varians notifikasi chat",
    "Varians notifikasi media sosial",
    "Varians notifikasi kerja",
    "Standar deviasi notifikasi chat",
    "Standar deviasi notifikasi media sosial",
    "Standar deviasi notifikasi kerja"
  ),
  Hasil = sapply(
    c(p_social_lebih30,p_chat_antara5_15,p_work_kurang10,
      mean_chat,mean_social,mean_work,
      var_chat,var_social,var_work,
      sd_chat,sd_social,sd_work),
    format_angka
  )
)
print(hasil, row.names = FALSE)
```

```
##                                Keterangan Hasil
##                          P(X_social > 30)  0.14
##                     P(X_chat antara 5-15)  0.92
##                            P(X_work < 10)  0.07
##                      Mean notifikasi chat    10
##              Mean notifikasi media sosial    25
##                     Mean notifikasi kerja    15
##                   Varians notifikasi chat    10
##           Varians notifikasi media sosial    25
##                  Varians notifikasi kerja    15
##           Standar deviasi notifikasi chat  3.16
##   Standar deviasi notifikasi media sosial     5
##          Standar deviasi notifikasi kerja  3.87
```

```r
# VISUALISASI DISTRIBUSI MEDIA SOSIAL
x_vals <- 0:50
prob_social <- dpois(x_vals, lambda_social)

barplot(prob_social,
        names.arg = x_vals,
        col = ifelse(x_vals > 30, "orange", "lightblue"),
        border = NA,
        main = "Distribusi Jumlah Notifikasi Media Sosial per Hari",
        xlab = "Jumlah Notifikasi",
        ylab = "Probabilitas",
        las = 1,
        ylim = c(0, 0.1))  # batas sumbu y sampai 1

# Tambahkan garis rata-rata
abline(v = lambda_social, col = "red", lwd = 2)
```

```r
text(lambda_social + 1, 0.95, paste("Rata-rata =", lambda_social),
     col = "red", adj = 0, cex = 0.5)

# Menambahkan legenda manual
usr <- par("usr")
x_leg <- usr[2] - 15
y_leg <- usr[4] * 0.95
segments(x_leg, y_leg, x_leg + 3, y_leg, col = "red", lwd = 2)
text(x_leg + 4, y_leg, "Rata-rata (λ = 25)", adj = 0, cex = 0.5)
rect(x_leg, y_leg - 0.005, x_leg + 3, y_leg - 0.015, col = "orange", border =
"gray40")
text(x_leg + 4, y_leg - 0.01, "Wilayah > 30 notifikasi", adj = 0, cex = 0.5)

# PLOT CDF (Kumulatif) DISTRIBUSI POISSON
x_vals <- 0:50
cum_prob <- ppois(x_vals, lambda_social)

plot(x_vals, cum_prob, type = "s", lwd = 2, col = "darkblue",
     main = "Distribusi Kumulatif Notifikasi Media Sosial",
     xlab = "Jumlah Notifikasi (x)",
     ylab = "Peluang P(X ≤ x)",
     ylim = c(0, 1))
grid()

# Garis bantu rata-rata
abline(v = lambda_social, col = "red", lwd = 2, lty = 2)
text(lambda_social -5 , 0.85, paste("λ =", lambda_social), col = "black")

# Tambahkan area untuk >30 (opsional visual)
abline(v = 30, col = "orange", lwd = 2, lty = 3)
text(31, 0.2, "Wilayah > 30", col = "black", adj = 0)
```





Kode ini mensimulasikan pemisahan proses Poisson untuk notifikasi ponsel selama satu hari dengan total rata-rata 50 notifikasi. Notifikasi dibagi menjadi tiga kategori yaitu chat (20%), media sosial (50%), dan pekerjaan (30%), sehingga masing-masing sub-proses tetap mengikuti Poisson independen dengan rata-rata proporsional yaitu 10, 25, dan 15 notifikasi per hari. Kode tersebut digunakan untuk menghitung nilai harapan, varians, dan simpangan baku masing-masing sub-proses, serta probabilitas kejadian tertentu, kemudian hasil ditampilkan dalam tabel ringkasan. Selain perhitungan, kode juga memvisualisasikan PDF dan CDF notifikasi media social. Barplot probabilitas menampilkan distribusi Poisson harian, dengan area >30 diberi warna berbeda, dan garis rata-rata $\lambda = 25$ ditandai dengan warna merah. CDF (kumulatif) ditampilkan menggunakan *plot step*, yang menunjukkan peluang $P(X \leq x)$ untuk setiap jumlah notifikasi, dengan garis rata-rata dan wilayah >30 sebagai panduan visual.

Gabungan Dua Proses Poisson Independen (Superposition)

Gabungan proses Poisson independen adalah proses yang terbentuk dari penggabungan dua atau lebih proses Poisson yang independen. Hasil penggabungan tetap merupakan proses Poisson, dengan laju total sama dengan penjumlahan laju masing-masing proses. Konsep ini merupakan kebalikan dari pemisahan (*thinning*). Jika pada *thinning process* kita memisahkan satu proses menjadi beberapa sub-proses, maka pada *superposition process* kita melakukan hal sebaliknya yaitu menggabungkan beberapa proses independen menjadi satu proses tunggal. Secara matematis, misalkan terdapat dua proses Poisson yang independen, yaitu $N_1(t) \sim \text{Poisson}(\lambda_1 t)$ dan $N_2(t) \sim \text{Poisson}(\lambda_2 t)$. Maka jumlahnya $N(t) = N_1(t) + N_2(t)$ akan menikuti distribusi $N(t) \sim \text{Poisson}\big((\lambda_1 + \lambda_2)t\big)$.

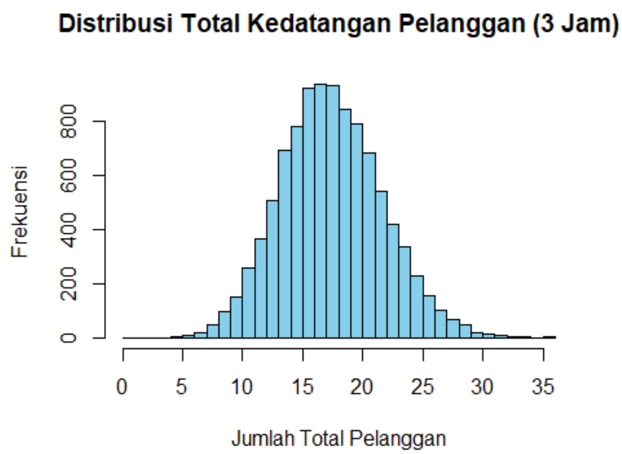
Bayangkan kamu mengamati kedatangan pelanggan ke sebuah pusat perbelanjaan yang memiliki dua pintu masuk yaitu pintu utama dan pintu samping. Pelanggan yang datang lewat pintu utama membentuk proses Poisson dengan rata-rata 8 orang per 10 menit, sedangkan pelanggan dari pintu samping membentuk proses Poisson dengan rata-rata 5 orang per 10 menit. Jika kedua pintu beroperasi secara independen, maka total pelanggan yang datang melalui kedua pintu juga mengikuti proses Poisson, dengan laju total $\lambda = \lambda_1 + \lambda_2 = 8 + 5 = 13$. Jadi rata-rata ada 13 pelanggan per 10 menit yang tiba di toko. Fenomena ini sering terjadi dalam sistem nyata, misalnya total panggilan telepon yang masuk ke *call center* dari dua wilayah berbeda, jumlah email yang diterima dari dua server independen, dan jumlah kendaraan yang melewati dua jalur tol menuju satu pintu keluar.

Contoh Simulasi Gabungan Dua Proses Poisson Independen

Sebuah kafe memiliki dua jenis pelanggan yang datang secara acak. Pelanggan yang membeli kopi (Proses A) datang dengan rata-rata 4 pelanggan per jam, sedangkan pelanggan yang hanya menggunakan Wi-Fi tanpa membeli (Proses B) datang dengan rata-rata 2 pelanggan per jam. Kedua jenis pelanggan ini datang secara independen, artinya kedatangan satu jenis pelanggan tidak memengaruhi jenis lainnya. Dalam praktikum ini, kita ingin mensimulasikan jumlah pelanggan yang datang selama 3 jam, menghitung total jumlah pelanggan dari kedua proses, dan memvisualisasikan distribusi kedatangan untuk memahami bagaimana kedua proses Poisson ini berinteraksi dalam jangka waktu tertentu.

```r
# SIMULASI GABUNGAN DUA PROSES POISSON
set.seed(123)

# Parameter
lambda_A <- 4   # rata-rata kedatangan pelanggan kopi per jam
lambda_B <- 2   # rata-rata kedatangan pelanggan Wi-Fi per jam
t <- 3          # satuan waktu (durasi simulasi dalam jam)

# Simulasi kedatangan masing-masing proses Poisson
n_A <- rpois(1, lambda_A * t)  # jumlah pelanggan kopi
n_B <- rpois(1, lambda_B * t)  # jumlah pelanggan Wi-Fi

# Total kedatangan
n_total <- n_A + n_B

# Menampilkan hasil simulasi menggunakan print
print("Hasil Simulasi:")

## [1] "Hasil Simulasi:"

print(paste("Jumlah pelanggan kopi:", n_A))

## [1] "Jumlah pelanggan kopi: 10"

print(paste("Jumlah pelanggan Wi-Fi:", n_B))

## [1] "Jumlah pelanggan Wi-Fi: 5"

print(paste("Total pelanggan:", n_total))

## [1] "Total pelanggan: 15"

# Simulasi berulang untuk melihat distribusi total kedatangan
sim <- 10000
total_kedatangan <- replicate(sim, {
```

```
  rpois(1, lambda_A * t) + rpois(1, lambda_B * t)
})

# Visualisasi distribusi total kedatangan
hist(total_kedatangan, breaks = 0:max(total_kedatangan), col="skyblue",
     main="Distribusi Total Kedatangan Pelanggan (3 Jam)",
     xlab="Jumlah Total Pelanggan", ylab="Frekuensi")
```



Kode R di atas digunakan untuk simulasi gabungan dua proses Poisson yaitu kedatangan pelanggan di sebuah kafe selama 3 jam. Langkah pertama yaitu mensimulasikan jumlah pelanggan dari masing-masing proses satu kali. Hasil simulasi menunjukkan bahwa misalnya jumlah pelanggan kopi adalah 14 dan jumlah pelanggan Wi-Fi adalah 5, sehingga total pelanggan yang datang selama 3 jam adalah 19 orang. Untuk mendapatkan gambaran distribusi total kedatangan, dilakukan simulasi berulang sebanyak 10.000 kali. Dari hasil ini, dapat dibuat histogram distribusi total kedatangan yang memperlihatkan frekuensi masing-masing jumlah pelanggan. Histogram ini membantu dalam melihat pola distribusi total kedatangan, di mana jumlah total pelanggan yang sering muncul berkisar di sekitar nilai rata-rata gabungan.

Contoh Perhitungan Kasus Gabungan Dua Proses Poisson Independen

Sebuah rumah sakit memiliki dua jenis pasien rawat jalan yang datang setiap harinya. Pasien penyakit ringan rata-rata datang 5 orang per hari, sedangkan pasien penyakit serius rata-rata datang 2 orang per hari. Diasumsikan bahwa kedatangan pasien ringan dan pasien serius saling independen dan mengikuti proses Poisson. Tentukan: (1) Distribusi total jumlah pasien yang datang setiap hari (dari 0 sampai 15 pasien), (2) Peluang bahwa dalam satu hari jumlah total pasien > 7, (3) Peluang bahwa jumlah total pasien antara 6–9 orang, (4) Peluang bahwa jumlah total pasien tepat 8 orang, (5) Nilai harapan dan varians dari total pasien per hari.

```
# CONTOH PERHITUNGAN KASUS GABUNGAN DUA PROSES POISSON
set.seed(123)

# Parameter
lambda_ringan <- 5   # pasien penyakit ringan per hari
```

```r
lambda_serius <- 2   # pasien penyakit serius per hari
t <- 1            # satuan waktu (durasi per hari)

# Rata-rata total (gabungan dua proses Poisson)
lambda_total <- (lambda_ringan + lambda_serius) * t

# Distribusi total pasien (teoretis) menggunakan Poisson
# Misal kita ingin melihat distribusi dari 0 sampai 15 pasien
k_values <- 0:15
P_total <- dpois(k_values, lambda_total)

# Menampilkan distribusi teoretis
print("Distribusi total pasien per hari (P(N=k)):")

## [1] "Distribusi total pasien per hari (P(N=k)):"

print(data.frame(Jumlah_Pasien=k_values, Probabilitas=round(P_total,4)))

##    Jumlah_Pasien Probabilitas
## 1              0       0.0009
## 2              1       0.0064
## 3              2       0.0223
## 4              3       0.0521
## 5              4       0.0912
## 6              5       0.1277
## 7              6       0.1490
## 8              7       0.1490
## 9              8       0.1304
## 10             9       0.1014
## 11            10       0.0710
## 12            11       0.0452
## 13            12       0.0263
## 14            13       0.0142
## 15            14       0.0071
## 16            15       0.0033

# Mencari Nilai Peluang
# 1) Peluang total > 7
P_lebih7 <- 1 - ppois(7, lambda_total)

# 2) Peluang total antara 6-9 pasien
P_antara6_9 <- ppois(9, lambda_total) - ppois(5, lambda_total)

# 3) Peluang total tepat 8 pasien
P_tepat8 <- dpois(8, lambda_total)

# Menampilkan hasil peluang
print(paste("Peluang total pasien > 7:", round(P_lebih7,4)))

## [1] "Peluang total pasien > 7: 0.4013"
```

```r
print(paste("Peluang total pasien antara 6-9:", round(P_antara6_9,4)))

## [1] "Peluang total pasien antara 6-9: 0.5298"

print(paste("Peluang total pasien tepat 8:", round(P_tepat8,4)))

## [1] "Peluang total pasien tepat 8: 0.1304"

# Nilai harapan dan varians
# Untuk gabungan dua proses Poisson independen:
E_total <- lambda_total
Var_total <- lambda_total
print(paste("Nilai harapan total pasien:", E_total))

## [1] "Nilai harapan total pasien: 7"

print(paste("Varians total pasien:", Var_total))

## [1] "Varians total pasien: 7"
```

Kode ini menghitung distribusi peluang dan statistik dasar untuk total pasien yang datang ke rumah sakit dalam satu hari, di mana pasien dibagi menjadi dua jenis yaitu penyakit ringan ($\lambda = 5$ per hari) dan penyakit serius ($\lambda = 2$ per hari). Karena kedatangan pasien ringan dan serius independen, maka jumlah total pasien mengikuti distribusi Poisson dengan rata-rata gabungan λ total = 5 + 2 = 7. Distribusi peluang untuk jumlah pasien dari 0 sampai 15, memperlihatkan bahwa probabilitas tertinggi terjadi sekitar 6–7 pasien. Selain itu, kode juga menampilkan hasil perhitungan peluang, nilai harapan, dan varians total pasien, yang keduanya sama dengan λ total = 7, sesuai sifat distribusi Poisson.