

Como simular filas reais?

Geração de um Processo de Poisson Não Homogêneo

ESTAT0090 – Estatística Computacional

Prof. Dr. Sadraque E. F. Lucena

sadraquelucena@academico.ufs.br

Cenário

Você precisa simular a chegada de clientes em uma cafeteria popular, onde o fluxo de clientes não é constante ao longo do dia. De manhã cedo, quando as pessoas pegam café para o trabalho, há um ritmo; no horário de almoço, há um pico; no meio da tarde, é mais tranquilo; e, no final da tarde, antes do fechamento, há outro aumento. Você sabe que as chegadas dos clientes seguem um Processo de Poisson, mas que a taxa de chegada varia conforme o horário do dia.

Objetivos da aula

- Compreender o conceito e a necessidade de um Processo de Poisson Não Homogêneo (PPNH) para modelar taxas de chegada que variam no tempo.
- Entender a lógica por trás do Método de Thinning (ou Afinamento) para simular um PPNH.
- Aplicar o algoritmo do Método de Thinning para gerar eventos de um PPNH em um período de tempo definido.
- Identificar a taxa máxima (λ) e definir a função de intensidade $\lambda(t)$ para a simulação de um PPNH a partir de dados ou de um cenário.
- Simular o fluxo de clientes em um cenário prático.

Processo de Poisson Não Homogêneo (PPNH)

Método Thinning (ou Afinamento)

- **Encontre uma taxa “máxima”:** Sabendo que a taxa de chegada de clientes $\lambda(t)$ varia ao longo de todo o período T , determine a maior taxa λ .
 - Matematicamente: $\lambda(t) \leq \lambda$ para todo $t \leq T$.
- **Gere um Processo de Poisson Homogêneo “exagerado”:** Gere um Processo de Poisson “normal” (homogêneo) com essa taxa “exagerada” λ . Esse processo vai gerar muitos eventos, alguns que você quer e outros que não.
- **Afinamento (ou seleção por amostragem):** Para cada evento que ocorre nesse Processo de Poisson homogêneo na taxa λ , você joga uma “moeda viciada”. A chance de esse evento ser “aceito” e fazer parte do seu PPNH original é dada pela razão $\frac{\lambda(t)}{\lambda}$ no momento t em que ele ocorreu.
 - Se $\lambda(t)$ for alta naquele t , a chance de aceitar é grande.
 - Se $\lambda(t)$ for baixa, a chance é menor.
- É como se você estivesse selecionando os eventos que correspondem à sua taxa de intensidade real $\lambda(t)$.

Método Thinning (ou Afinamento)

Algoritmo

Passo 1: Inicialize.

- $t=0$ (tempo atual começa em zero)
- $I=0$ (contador de eventos começa em zero)

Passo 2: Gere o tempo entre eventos usando a distribuição Exponencial, considerando um processo homogêneo “exagerado”.

- Se $t > T$, significa que o próximo evento cairia depois do nosso tempo de simulação desejado. Então, paramos.

Passo 3: Gere $u \sim U(0, 1)$.

Método Thinning (ou Afinamento)

Algoritmo (continuação)

Passo 4: Se $u \leq \frac{\lambda(t)}{\lambda}$, então o evento é aceito.

- Incrementamos o contador de eventos: $I = I + 1$.
- Armazenamos o tempo do evento: $S[I] = t$. (onde $S[I]$ é o tempo do I -ésimo evento)

Passo 5: Volte para o Passo 2 para gerar o próximo evento do processo homogêneo.

Exemplo 14.1

Suponha que uma cafeteria abre das 08h às 20h, com as seguintes taxas por hora:

- 08h-09h (primeira hora): 1 cliente por hora
- 09h-10h (segunda hora): 2 clientes por hora
- 10h-11h (terceira hora): 3 clientes por hora
- 11h-12h (quarta hora): 4 clientes por hora
- 12h-13h (quinta hora): 5 clientes por hora
- 13h-20h (demais horas): 10 clientes por hora

Simule o fluxo de clientes em um dia.

Exemplo 14.1

```
# Função que define o valor de lambda de acordo com a hora
lambdat <- function(t){
  t <- trunc(t)
  lambda <- ifelse(t < 1, 1,
                  ifelse(t < 2, 2,
                        ifelse(t < 3, 3,
                              ifelse(t < 4, 4,
                                    ifelse(t < 5, 5, 10))))))
  return(lambda)
}
```


Exemplo 14.1

```
# Função que simula o processo de Poisson
ppnh <- function(lambda_t, lambda, T = 12){ # T = 12: loja 12h aberta
  t <- 0; I <- 0; S <- vector()
  repeat{
    t <- t + rexp(1, lambda) # incrementando o tempo até nova chegada
    if(t > T)                # se o tempo t é maior que o tempo total T
      break                  # sai do laço
    u <- runif(1)
    if(u <= lambda_t(t)/lambda){
      I <- I + 1 # incrementa o contador de eventos
      S[I] <- t  # registra o tempo do evento
    }
  }
  cat("Número de eventos:", I, "\n")
  return(S)
}
```

Exemplo 14.1

- Como a maior taxa é 10, vamos usar `lambda = 10`.

```
set.seed(1234)
tempos1 <- ppnh(lambda_t = lambdat, lambda = 10)
tempos2 <- ppnh(lambda_t = lambdat, lambda = 10)
tempos3 <- ppnh(lambda_t = lambdat, lambda = 10)

# convertendo o tempo para as horas que a loja está aberta
tempos1 <- tempos1 + 8
tempos2 <- tempos2 + 8
tempos3 <- tempos3 + 8

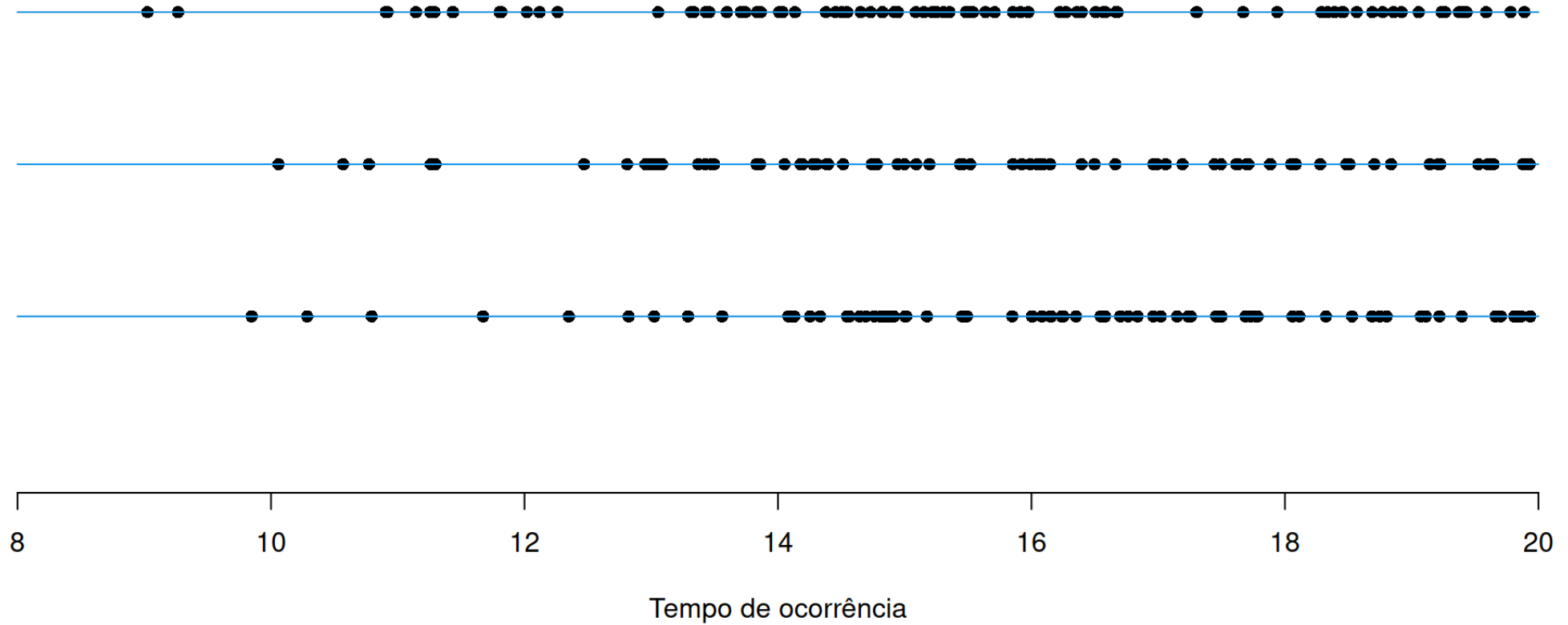
par(mar=c(5,0,0,0))
plot(c(tempos1, tempos2, tempos3),
     c(rep(2, length(tempos1)), rep(1, length(tempos2)),
       rep(0, length(tempos3))), axes = F, ylim = c(-1, 3),
     xlim = c(8, 20), ylab = "",
     xlab = "Tempo de ocorrência", pch = 16)
segments(x0 = 8, y0 = 2, x1 = 20, y1 = 2, col = 4)
segments(x0 = 8, y0 = 1, x1 = 20, y1 = 1, col = 4)
segments(x0 = 8, y0 = 0, x1 = 20, y1 = 0, col = 4)
axis(1, at = seq(8,20,2))
```

Exemplo 14.1

Número de eventos: 92

Número de eventos: 73

Número de eventos: 76



Eficiência do Thinning (ou Afinamento)

- Se os valores de $\lambda(t)$ estiverem próximos de λ durante todo o intervalo, o algoritmo é mais eficiente.
- Isso porque se λ for muito maior que a maioria dos valores de $\lambda(t)$, você vai gerar muitos eventos do processo homogêneo que serão “rejeitados” no Passo 4.
- Existem alternativas a esse algoritmo que visam diminuir o número de eventos rejeitados e diminuir o custo computacional.

Exercício 14.1

O arquivo `registros.csv` contém os horários exatos em que os clientes de uma agência bancária pegaram uma senha para atendimento no caixa em uma segunda-feira típica. Use os registros para simular o fluxo de entrada dos clientes em uma segunda-feira típica.

Ganho da aula

- Capacidade de simular situações do dia a dia onde a demanda varia.

Fim

Aula baseada no material “Métodos Computacionais Aplicados à Estatística Implementação no Software R” de Cristiano de Carvalho Santos.