Statistics for Engineers
TI0111

Homework II

Nome completo: Leonardo Dayvison Silva de Almeida Teixeira, André Venâncio Sousa Grangeiro Filho
Numero de matricula: 566868, 564717

Questão 1

Em um restaurante muito frequentado, aproximadamente 70% dos clientes pedem uma sobremesa após o prato principal. Seja X a variável aleatória que representa o número de clientes que pedem sobremesa em uma amostra aleatória de $n = 50$ clientes.

1. Determine a função de distribuição de X.
2. Construa os gráficos da função massa de probabilidade (PMF) e da função distribuição acumulada (CDF) de X.
3. Calcule o valor esperado, a variância e o desvio padrão dev X.
4. Calcule a probabilidade de:
   (a) $P(X \geq 20)$.
   (b) $P(30 < X < 43)$.
   (c) $P(X = 31)$.
5. Suponha que o restaurante estoque sobremesas com base na demanda esperada. Como o uso da distribuição de X poderia ajudar a reduzir desperdício e evitar falta de produtos.
6. Como mudanças em p (por exemplo, sobremesa se torna mais popular, $p = 0.8$) ou em n (número de clientes) afetariam a forma e as probabilidades de X?

Solução da questão 1

1. Sabendo que a variável aleatória X representa a quantidade de resultados "positivos"em dada quantidade de experimentos de Bernoulli (o cliente quer ou não quer a

sobremesa), sabemos que sua distribuição é binomial. A distribuição de probabilidade de variáveis binomiais é dada por:

$$P(X = k) = \binom{n}{k}(p)^k(1-p)^{n-k} \tag{1}$$

Em que "n"é a quantidade de experimentos realizados, "k"é a quantidade de sucessos cuja probabilidade estamos avaliando e "p"é a probabilidade de obter um sucesso. Substituindo os valores para os dados enunciados na questão, obtemos a distribuição:

$$P(X = k) = \binom{50}{k}(0.7)^k(0.3)^{50-k}$$

Substituindo os valores, podemos verificar o resultado obtido em R:

```
k <- 0:n
n <- 50
p <- 0.7

probability <- dbinom(x = k,
                      size = n,
                      prob = p)
probability
```

Por exemplo, se substituirmos k = 40:

$$P(X = k) = \binom{50}{40}(0.7)^{40}(0.3)^{10} = 0.038$$

Que está conforme o resultado da distribuição em R, indicado na figura 1:

```
> probability
 [1] 7.178980e-27 8.375477e-25 4.787981e-23 1.787513e-21 4.900764e-20 1.052031e-18
 [7] 1.841054e-17 2.700212e-16 3.386516e-15 3.687540e-14 3.527746e-13 2.993239e-12
[13] 2.269873e-11 1.548170e-10 9.547048e-10 5.346347e-09 2.728865e-08 1.273470e-07
[19] 5.447622e-07 2.140820e-06 7.742632e-06 2.580877e-05 7.938153e-05 2.254896e-04
[25] 5.919101e-04 1.436369e-03 3.222622e-03 6.683956e-03 1.281092e-02 2.267679e-02
[31] 3.703876e-02 5.575728e-02 7.724706e-02 9.831444e-02 1.147002e-01 1.223469e-01
[37] 1.189483e-01 1.050175e-01 8.382972e-02 6.018544e-02 3.861899e-02 2.197829e-02
[43] 1.098914e-02 4.770481e-03 1.770860e-03 5.509343e-04 1.397297e-04 2.774775e-05
[49] 4.046546e-06 3.853854e-07 1.798465e-08
```

Figura 1: Resultado q1i1

2. Para construirmos os gráficos, utilizaremos a função plot da linguagem R. Para a PMF, utilizaremos os valores no intervalo (0,50) para substituir k na função de distribuição e, para a CDF, faremos o mesmo, porém, somando todos os resultados do intervalo (0,i) para todo valor k=i, resultando em $P(X \leq k)$. Esses cálculos são feitos automaticamente pelas funções dbinom() e pbinom(), que fazem o cálculo para um único valor e o acumulativo, respectivamente.

```
plot(x = k,
     y = probability,
     ylab = "Probability",
     xlab = "k",
     main = "Probability Mass Function")

cumfunc <- pbinom(q = k,
                  size = n,
                  prob = p)
plot(x = k,
     y = cumfunc,
     xlab = "k",
     ylab = "Probability",
     main = "Cumulative Distribution Function")
```

Resultando nos gráficos das figuras 2 e 3:

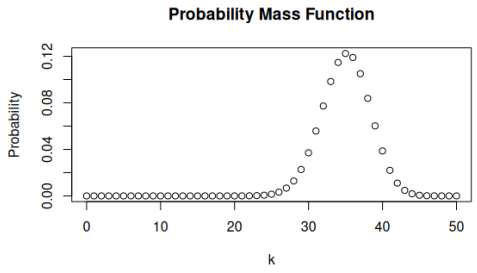

Figura 2: PMF da distribuição binomial

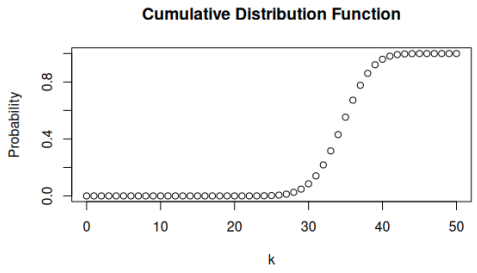

Figura 3: CDF da distribuição binomial

3. O valor esperado da variável aleatória trata-se da média de todos os seus possíveis valores, ponderada por suas respectivas probabilidades, podemos enunciar:

$$E[X] = \sum_{x_i \in R_x} x_i \cdot P(X = x_i) \tag{2}$$

Em que R_x é o suporte da PMF de X. Logo, podemos calculá-lo:

$$E[X] = 1 \cdot P(X = 1) + 2 \cdot P(X = 2)... = 35$$

A variância pode ser definida como a medida do quão dispersos estão os valores de nossas amostras, calculada por:

$$Var(X) = E[(X - \mu_x)^2] \tag{3}$$

Sendo μ_x o valor esperado / média dos valores de X.
O desvio padrão é apenas a raiz quadrada da variância, logo, é intuitivo calculá-la.

```
expected_value <- sum(k*probability)

diff <- (k - expected_value)^2
x_variance <- sum(diff*probability)

std_dev <- sqrt(x_variance)

expected_value
x_variance
std_dev
```

Resultando nos valores indicados pela figura 4:

```
> expected_value
[1] 35
> x_variance
[1] 10.5
> std_dev
[1] 3.24037
```

Figura 4: Valor esperado, Variância e Desvio Padrão

4. (a) Para calcular a probabilidade enunciada na questão, podemos subtrair a CDF de k=1 e subtrair de 1, uma vez que o valor total da PMF soma em 1, e subtrair esse valor "eliminará"a probabilidade de qualquer valor abaixo de 20. Assim:

$$P(X \geq 20) = 1 - P(X \leq 19)$$

(b) Nesse caso, precisaremos obter a CDF de X com k=42 e, dela, subtrair a CDF de X com k=30, para obtermos o intervalo exclusivo de probabilidades 30<X<43. Desse modo:

$$P(30 < X < 43) = P(X \leq 42) - P(X \leq 30)$$

(c) Para esta probabilidade, apenas substituiremos k=31 na PMF de X, de modo a obter $P(X = 31)$.

```
prob_leq_20 <- 1 - pbinom(q=19, size=n, prob=p)
prob_btw_3043 <- pbinom(q=42, size=n, prob=p) - pbinom(q=30, size=n, prob
    =p)
prob_eq_31 <- dbinom(x=31, size=n, prob=p)

prob_leq_20
prob_btw_3043
prob_eq_31
```

Obtendo os resultados apresentados na figura 5:

```
> prob_leq_20
[1] 0.9999972
> prob_btw_3043
[1] 0.9079332
> prob_eq_31
[1] 0.05575728
```

Figura 5: Cálculos de probabilidade do item 4, questão 1

5. Com os valores que nos foram dados, pudemos calcular a quantidade mais provável de clientes pedindo sobremesa em uma noite, sendo este o valor esperado de X. Logo, esse valor deveria ser a quantidade de sobremesas no estoque, visto que há uma chance maior de suprir a demanda de clientes mantendo o menor desperdício possível. Para obter um resultado mais realista, o restaurante deveria obter a quantidade média de clientes que frequentam o restaurante e utilizar o valor em n, para saber quantos clientes, no total, esperar que peçam sobremesa.

6. Podemos observar, por cálculos ou empiricamente, que a mudança nos valores de "n"e "p"afetam diretamente as probabilidades de valores em X. Aumentamos um pouco cada um desses valores e pudemos observar mudanças nos gráficos das figuras 6 e 7:

```
popular_dessert_prob <- dbinom(x = k,
                               size = n,
                               prob = p + 0.1)
more_clients <- c(k, 51:60)
more_clients_prob <- dbinom(x = more_clients,
                                size = n + 10,
                                prob = p)
plot(y=popular_dessert_prob,
     x=k,
     ylab="Probability␣w/␣more␣popular␣dessert")
plot(y = more_clients_prob,
     x = more_clients,
     ylab="Probability␣w/␣more␣clients")
```

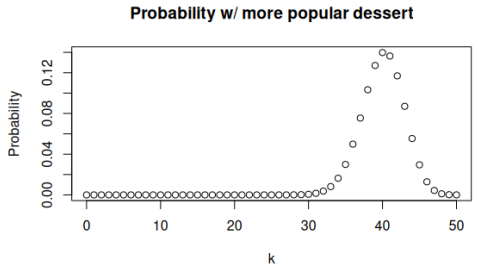

Figura 6: PMF with p=0.8

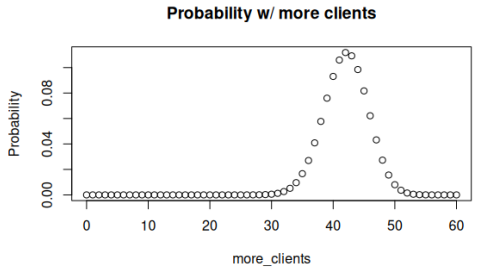

Figura 7: PMF with n=60

Podemos observar que, ao alterar os valores, o gráfico da PMF é "deslocado", aumentando o valor esperado, provocando mudança direta na função.

Questão 2

Um site realiza uma pesquisa online e oferece uma recompensa a um usuário selecionado aleatoriamente que responde a uma série de perguntas. Cada um dos 10 milhões de visitantes diários tem, independentemente, probabilidade $p = 10^{-7}$ de ganhar a recompensa.

1. Encontre uma aproximação simples e adequada para a função de massa de probabilidade (PMF) do número de vencedores em um dia, X. Justifique claramente se essa aproximação é apropriada para os valores dados de n and p.

2. Calcule o valor esperado, $E[X]$, e a variância $\mathrm{Var}(X)$, usando tanto a distribuição exata quanto a aproximada. Comente sobre a semelhança entre os resultados.

3. Suponha que você ganhe a recompensa, mas que possa haver outros vencedores. Seja $W \sim \mathrm{Pois}(1)$ o número de vencedores além de você. Se houver vários vencedores, o prêmio é sorteado aleatoriamente entre todos eles. Encontre a probabilidade de que você realmente receba o prêmio.

4. Gere um grande número de simulações diárias para o número de vencedores. Crie uma comparação visual entre os resultados empíricos e a aproximação considerada no item 1. Descreva brevemente o que a visualização indica sobre a qualidade da aproximação.

Solução da questão 2

1. Ao analisar os dados apresentados na questão, verificamos que a distribuição exata para essa situação é uma binomial com $n = 10^7$ e $p = 10^{-7}$. Entretanto, podemos perceber que a probabilidade de uma pessoa ganhar é baixa (evento raro, com p pequeno), enquanto a quantidade de usuários que acessam o site por dia é alta (alta possibilidade de ocorrência,

com n alto), caracterizando uma situação em que a distribuição de Poisson é aplicável[1]. Assim, uma aproximação adequada para a função de massa de probabilidade (PMF) da variável aleatória X (número de vencedores) é dada por uma Poisson com parâmetro $\lambda = 10^7 \times 10^{-7} = 1$. Então, temos:

$$P(X = k) = \frac{e^{-\lambda}\lambda^k}{k!} \tag{4}$$

$$P(X = k) = \frac{e^{-1} \cdot 1^k}{k!} = \frac{1}{e \cdot k!} \tag{5}$$

2. Para a distribuição exata, que é uma Binomial $X \sim B(10^7, 10^{-7})$, temos:

$$E[X] = np = 10^7 \cdot 10^{-7} = 1 \tag{6}$$

$$Var(X) = np(1-p) = 1 \cdot (1 - 10^{-7}) = 0,9999999 \tag{7}$$

Para a distribuição aproximada, com $X \sim Pois(1)$:

$$E[X] = \lambda = 1 \tag{8}$$

$$Var(X) = \lambda = 1 \tag{9}$$

Comparando esses resultados, podemos perceber que os valores esperados são idênticos, enquanto a variância apresenta uma diferença de apenas 10^{-7} entre o modelo exato e o aproximado. Essa discrepância mínima valida o uso da distribuição de Poisson como aproximação, mostrando que as propriedades estatísticas da distribuição original são preservadas.

3. É necessário calcular a probabilidade de que um participante ganhe o prêmio dado que ele foi selecionado. Dessa forma, tomando G como o evento de ganhar e S como o evento de ser selecionado (qualificado), podemos aplicar a regra de Bayes:

$$P(G \mid S) = P(S \mid G) \times \frac{P(G)}{P(S)} \tag{10}$$

Sabemos que uma pessoa só pode ganhar se tiver sido selecionada, então $P(S \mid G) = 1$. Além disso, temos, do enunciado, que a probabilidade de ser selecionado é $P(S) = p = 10^{-7}$. Assim, substituindo esses valores na equação (10):

$$P(G \mid S) = 1 \times \frac{P(G)}{10^{-7}} \tag{11}$$

[1] https://mathcenter.oxford.emory.edu/site/math117/connectingPoissonAndBinomial/

Resta calcular a probabilidade a priori ($P(G)$) de se ganhar a recompensa. O enunciado informa que o número de outros vencedores segue a distribuição $W \sim Pois(1)$. Como o número total de usuários n é muito grande, a distribuição do número total de vencedores (X) segue o mesmo parâmetro $\lambda = 1$ da variável W.

O prêmio é entregue se houver pelo menos um vencedor no total ($X \geq 1$). Calculamos essa probabilidade:

$$P(X \geq 1) = 1 - P(X = 0) = 1 - e^{-1} \tag{12}$$

Considerando a simetria entre os usuários, a probabilidade incondicional de um usuário específico receber o prêmio é a probabilidade do prêmio sair dividida pelo número total de usuários (n):

$$P(G) = \frac{P(X \geq 1)}{n} = \frac{1 - e^{-1}}{10^7} \tag{13}$$

Retornando à equação do Teorema de Bayes e substituindo $P(G)$:

$$P(G \mid S) = \frac{\frac{1-e^{-1}}{10^7}}{10^{-7}} = \frac{1 - e^{-1}}{1} \tag{14}$$

$$P(G \mid S) = 1 - e^{-1} \approx 0,6321 \tag{15}$$

4. Utilizando o código 1, são geradas 100.000 simulações da distribuição exata desse cenário, uma Bin($n = 10^7$, $p = 10^{-7}$), as quais são representadas pelo histograma em azul claro na figura. Além disso, em vermelho, sobrepostos ao histograma, estão os pontos que indicam a função aproximada $X \sim Pois(1)$.

Listado 1: Codigo desenvolvido na solução da questão 2 item 4.

```
simulacoes <- rbinom(100000, size = 10^7, prob = 10^-7)
df_sim <- data.frame(vencedores = simulacoes)

x_vals <- 0:8
df_teo <- data.frame(x = x_vals, prob = dpois(x_vals, lambda = 1))

ggplot(df_sim, aes(x = vencedores)) +
  geom_histogram(aes(y = after_stat(density)), binwidth = 1, fill = "
      lightblue", color = "black") +

  geom_point(data = df_teo, aes(x = x, y = prob), color = "red") +
  geom_line(data = df_teo, aes(x = x, y = prob), color = "red") +
  labs(title = "Simulacao␣vs␣Teoria", x = "Numero␣de␣Vencedores", y = "
      Probabilidade")
```

Analisando a figura 8, percebe-se uma ótima sobreposição entre as frequências empíricas e a curva teórica. Isso confirma que a aproximação de Poisson é extremamente eficaz para este cenário, oferecendo uma alternativa mais simples e com cálculos menos complicados do que a Binomial com n elevado.

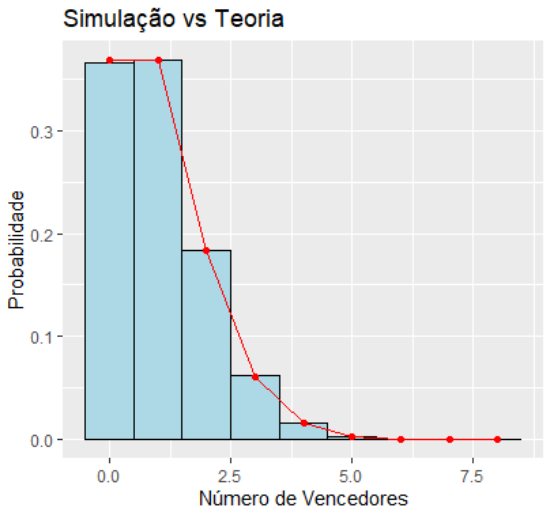

Figura 8: Representação visual: dados empíricos (azul) vs aproximação de Poisson (vermelho).

Questão 3

Você é responsável por monitorar a temperatura de uma CPU multicore em uma unidade de processamento embarcada. Sob carga normal, a temperatura da CPU apresenta flutuações devido a mudanças na carga de trabalho, nas condições ambientais e na eficiência do sistema de resfriamento. Testes mostram que a temperatura em regime estacionário da CPU segue uma distribuição normal com temperatura média $\mu = 62\,^{\circ}\text{C}$ e desvio-padrão $\sigma = 3,5\,^{\circ}\text{C}$. Sua tarefa é simular medições de temperatura da CPU e analisar suas propriedades estatísticas.

1. Crie uma função que gere valores com distribuição normal usando a transformação de Box-Muller[2], a partir de entradas aleatórias uniformes. Especificamente:

   (a) Gere duas variáveis aleatórias uniformes independentes: U_1, $U_2 \sim \text{Unif}(0, 1)$.

   (b) Calcule dois valores normais padrão usando as fórmulas de Box-Muller:

   $$Z_1 = \sqrt{-2\ln(U_1)}\cos(2\pi U_2), \quad Z_2 = \sqrt{-2\ln(U_1)}\sin(2\pi U_2)$$

   Z_1 e Z_2 são variáveis aleatórias independentes com distribuição normal padrão. Concatene-as para formar um vetor Z de valores normais padrão.

   (c) Converta cada valor normal padrão para a distribuição de temperatura da CPU:

   $$T = 62 + 3.5\,Z$$

[2] https://en.wikipedia.org/wiki/Box-Muller_transform

2. Use seu gerador de números aleatórios para gerar 1.000 medições de temperatura da CPU.

   Gere mais 1.000 valores de temperatura utilizando o gerador de números aleatórios normal embutido do R, com a mesma média e desvio-padrão.

3. Para ambos os conjuntos de dados simulados, calcule:

   (a) Média amostral.

   (b) Desvio-padrão amostral.

   (c) Temperatura mínima e máxima observada.

   (d) Probabilidade empírica e teórica $P(T > 68)$.

   (e) Probabilidade empírica e teórica $P(60 < T < 65)$.

   (f) Probabilidade teórica $P(T > 75)$.

   Algum dos conjuntos de dados simulados (1.000 amostras) contém valores acima de 75°C? Caso não, explique por que eventos raros requerem tamanhos de amostra grandes para serem observados.

4. Visualize os resultados criando:

   (a) Um histograma das temperaturas simuladas da CPU (pode plotar os dois conjuntos de dados separadamente ou sobrepostos).

   (b) A função densidade de probabilidade (PDF) normal teórica (média 62 °C, desvio padrão $3,5$ °C) sobreposta ao histograma.

5. Discuta seus resultados respondendo às seguintes perguntas: As distribuições empíricas da temperatura da CPU se assemelham à curva normal teórica? Quão próximas estão a média amostral e o desvio-padrão amostral dos valores esperados 62 °C e $3,5$ °C? Há diferenças perceptíveis entre o conjunto de dados gerado com seu RNG manual e o produzido pelo RNG embutido do R? Como essa simulação pode ajudar na avaliação de estratégias de resfriamento ou de escalonamento dinâmico de clock? Por que geradores de números aleatórios uniformes são a base dos sistemas de RNG?

Solução da questão 3

1. A questão enuncia um passo-a-passo para construir a função da transformação de box-muller, a qual será apresentada, também, por passos.

(a) Utilizaremos a função runif() do R para gerarmos duas variáveis aleatórias independentes e uniformemente distribuidas entre 0 e 1, como mostrado no Listado 3

```
boxmuller <- function(){
  U1 = runif(1)
  U2 = runif(1)
```

(b) Em seguida, utilizamos as fórmulas de Box-Muller para calcular valores normais padrão, como indica o Listado 3

```
  R = sqrt(-2*log(U1))
  theta = 2*pi*U2
  Z0 = R*cos(theta)
  Z1 = R*sin(theta)
```

(c) E, finalmente, convertemos para a distribuição da CPU com nossa média e nosso desvio padrão e retornamos os valores concatenados, como indica o Listado 3

```
  temp1 = Z0*3.5 + 62
  temp2 = Z1*3.5 + 62
  res <- c(temp1, temp2)
  return(res)
}
```

Agora, a função boxmuller() criada já pode ser utilizada pra gerar valores aleatórios da nossa distribuição.

2. Agora, vamos utilizar a função construída para gerar 1000 amostras aleatórias da CPU enunciada pela questão, como mostra o Listado 3

```
# Empty vectors
v1 = rep(NA, 1000)
v2 = rep(NA, 1000)

idx <- 1
for (i in 1:500) {
  vals <- boxmuller()
  v1[idx]     <- vals[1]
  v1[idx + 1] <- vals[2]
  idx <- idx + 2
}
```

E, em seguida, geramos mais 1000 valores utilizando a função embutida rnorm(), como no listado 3

```
v2 <- rnorm(1000, mean=62, sd=3.5)
```

Agora, temos dois conjuntos de amostras, o gerado pela nossa função, na variável v1, e o gerado pela função embutida do R, na variável v2.

3. Para calcular os parâmetros requisitados pela questão, utilizamos as funções nativas do R em conjunto com a aplicação direta da fórmula matemática, a fim de validar os resultados obtidos.

(a) Como é possível verificar no Listado 3, calculamos tanto a média ($m1$) do conjunto gerado usando o método de Box-Muller, quanto a média ($m2$) do conjunto gerado pela função `rnorm`. A variável $m1\_calc$ foi utilizada para verificar o uso da função `mean` nativa do R.

```
m1 = mean(v1)
m1_calc = sum(v1)/1000
```

```
m2 = mean(v2)
```

(b) Calculamos os desvios-padrão para o conjunto de Box-Muller ($dp1$) e para o `rnorm` ($dp2$). Os valores foram muito próximos, indicando consistência na dispersão dos dados. Também calculamos manualmente a variância e o desvio ($dp1\_calc$) para validar o uso da função `sd`.

```
dp1 = sd(v1)
var1_calc = sum((v1-m1)^2)/(1000-1)
dp1_calc = sqrt(var1_calc)

dp2 = sd(v2)
```

(c) Verificamos os valores mínimos e máximos de cada amostra. Na execução atual, a diferença entre os máximos ($max1$ e $max2$) foi de aproximadamente $0,7°C$, enquanto a diferença entre os mínimos foi de cerca de $2,9°C$. Tais variações são esperadas devido à aleatoriedade do processo de amostragem.

```
#max
max1 = max(v1)
max1_calc = v1[1]
for (i in 1:1000) {
  if (v1[i] > max1_calc) {
    max1_calc <- v1[i]
  }
}
max2 = max(v2)

#min
min1 = min(v1)
min1_calc = v1[1]
for (i in 1:1000) {
  if (v1[i] < min1_calc) {
    min1_calc <- v1[i]
  }
}
min2 = min(v2)
```

(d) Calculamos a probabilidade empírica do evento $T > 68$ para o Box-Muller ($p68\_emp1$) e para o `rnorm` ($p68\_emp2$). A probabilidade teórica ($p68\_teo$) é idêntica para ambos os modelos. Obtivemos valores empíricos, respectivamente, de $0,036$ e $0,042$ para $p68\_emp1$ e $p68\_emp2$ contra um teórico de $\approx 0,043$.

```
aux <- v1[v1 > 68]
p68_emp1 = length(aux)/length(v1)

aux <- v2[v2 >68]
p68_emp2 <- length(aux)/length(v2)


p68_teo = pnorm(68, mean=62, sd=3.5, lower.tail = FALSE, log.p = FALSE)
```

(e) De maneira análoga, analisamos o intervalo $60 < T < 65$. As probabilidades empíricas obtidas foram $p60\_emp1 = 0,494$ e $p60\_emp2 = 0,532$, comparadas a um

valor teórico $p60\_teo \approx 0,520$. Novamente, os resultados oscilam em torno do valor esperado, validando a simulação.

```
aux <- v1[60 < v1 & v1 < 65]
p60_emp1 = length(aux)/length(v1)

aux <- v2[60 < v2 & v2 < 65]
p60_emp2 <- length(aux)/length(v2)

p60_teo = pnorm(65, mean=62, sd=3.5, lower.tail = TRUE, log.p = FALSE) -
    pnorm(60, mean=62, sd=3.5, lower.tail = TRUE, log.p = FALSE)
```

(f) A probabilidade teórica de $T > 75$ é de aproximadamente $0,0001$. Nas diversas simulações que realizamos, nenhum dos vetores apresentou valores superiores a 75°C. Analisando os dados, vemos que isso ocorre porque o número esperado de eventos é $E = n \times p = 1000 \times 0,0001 = 0,1$. Como o valor esperado é muito inferior a 1, seria necessário um tamanho amostral significativamente maior (10^4 ou 10^5) para que a ocorrência de um evento raro como esse se tornasse provável.

```
p75_teo = pnorm(75, mean=62, sd=3.5, lower.tail = FALSE, log.p = FALSE)

v1_75 <- v1[v1 > 75]
v2_75 <- v2[v2 > 75]
```

4. Vamos plotar o histograma dos dois conjuntos de amostras gerados. Para isso, vamos criar dois dataframes, **df1** e **df2**, os quais terão uma coluna com os valores armazenados em v1 e v2 e uma coluna que servirá para "etiquetar"a qual conjunto de amostras o respectivo dado pertence. Os dois serão concatenados em um novo dataframe **df3**, cujos valores serão convertidos para numéricos e plotados. Isso foi realizado como no Listado 3

```
label <- rep("empvals", 1000)
samples <- v1
df1 <- as.data.frame(cbind(samples, label))

label <- rep("rvals", 1000)
samples <- v2
df2 <- as.data.frame(cbind(samples, label))

df3 <- rbind(df1, df2)
df3$samples <- as.numeric(df3$samples)

ggplot(df3, aes(x=samples, color=label, fill=label)) +
  geom_histogram(position="identity", alpha=0.5)+
  theme(legend.position="top") + theme_classic()
```

Gerando, como resultado, o gráfico indicado na figura 9

Podemos notar que ambas as distribuições são muito parecidas, concentrando sua maior quantidade de amostras em valores próximos à média estabelecida em 62, podemos supor, empiricamente, que elas estão igualmente distribuidas.

Para que consigamos sobrepor a PDF da distribuição normal, iremos converter o histograma para densidade, uma vez que seu eixo Y indica a contagem de amostras,

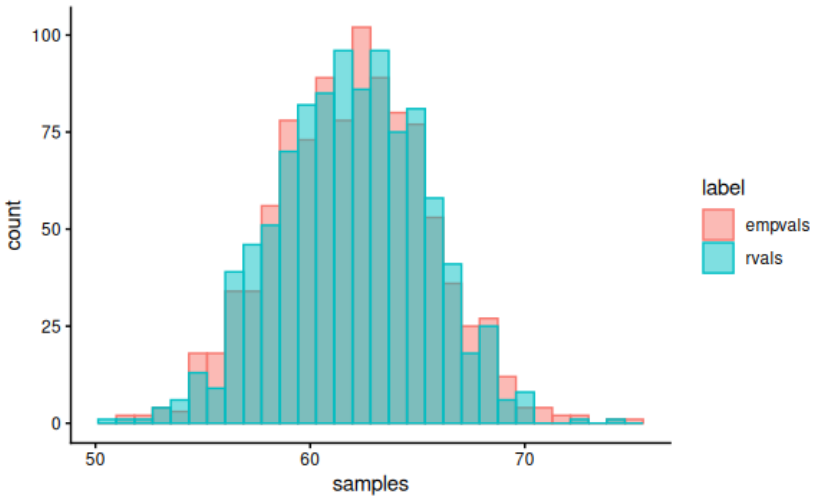

Figura 9: Gráfico comparativo dos conjuntos de amostras

enquanto o eixo Y da PDF indica a densidade de probabilidades. Isso foi feito como no Listado 3

```
ggplot(df3, aes(x = samples, color = label, fill = label)) +
  geom_histogram(
    aes(y = after_stat(density)),
    position = "identity",
    alpha = 0.5,
    bins = 30
  ) +
  stat_function(
    fun = dnorm,
    args = list(mean = cpumean, sd = cpusd),
    color = "black",
    linewidth = 1
  ) +
  theme_classic() +
  theme(legend.position = "top")
```

Gerando agora o gráfico representado na figura 10

Podemos observar que ambas as distribuições estão muito próximas da curva sobreposta. Esse fato torna nítido que ambas são igualmente distribuidas, sob distribuição normal de média 62 e desvio padrão 3.5, mostrando a eficácia da transformação de Box-Muller na geração de amostras normalmente distribuidas.

5. Podemos observar, por exemplo, por meio dos parâmetros calculados durante o item 3, que as distribuições empíricas obtidas, tanto via método de Box-Muller quanto pela função embutida no R, assemelham-se à curva normal teórica. O histograma gerado no Item 4 exibe o característico formato de sino centrado por volta do valor teórico da média, 62, evidenciando a convergência dos dados para a distribuição esperada. Observou-se

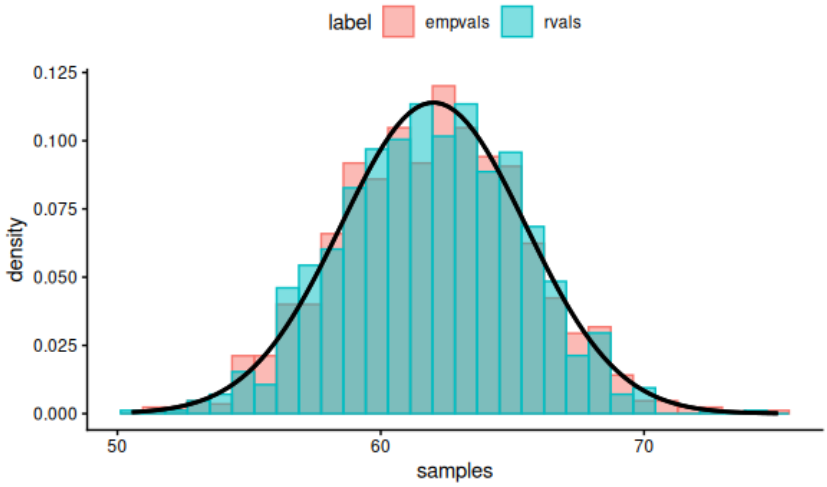

Figura 10: Curva da PDF da distribuição normal sobreposta ao histograma

que a média amostral e o desvio-padrão amostral aproximaram-se consistentemente dos valores esperados ($\mu = 62$ e $\sigma = 3,5$). As pequenas divergências verificadas são atribuídas à variação natural da amostragem e tendem a diminuir com o aumento do tamanho da amostra (n). Além disso, não houve diferenças estatísticas perceptíveis entre o conjunto gerado pelo algoritmo manual e o gerado pela função nativa do R, o que valida a implementação do método de Box-Muller realizada no Item 1.

No contexto de sistemas embarcados, essa simulação é uma importante ferramenta para a avaliação de estratégias de resfriamento e escalonamento dinâmico de clock. Quantificando probabilidades, como a chance de aproximadamente 4% da temperatura exceder 68°C ou a raridade extrema de eventos acima de 75°C, é possível definir limiares precisos para o acionamento de sistemas de resfriamento, otimizando o consumo energético sem expor o hardware a riscos reais durante a fase de testes.

Por fim, a utilização de geradores de números aleatórios uniformes ($Unif(0, 1)$) como base para sistemas de RNG é útil devido à natureza limitada dos processadores atuais. É mais eficiente, para um computador, gerar números pseudoaleatórios que estejam nesse intervalo entre 0 e 1. Depois, partindo da distribuição uniforme, aplicam-se transformações matemáticas como o método de Box-Muller utilizado neste trabalho, a fim de transformar essa probabilidade "plana"em distribuições estatísticas mais complexas, como a Normal, necessárias para modelar fenômenos reais.