Relatório de Processos Estocásticos - Simulações de Problemas em R

Gustavo Sutter Pessurno de Carvalho (9763193)
Victor Henrique de Souza Rodrigues (9791027)

07 de Junho de 2018

1 Questão 1:

Nessa questão foram utilizadas técnicas de potenciação de matriz e o método dos autovalores e autovetores para calcular a probabilidade $P = (X_3 = 0|X_0 = 0)$ cuja matriz de transição encontra-se na lista 1 exercício 05.

Figura 1: Funções para calcular pelo método das potências e dos autovalores e autovetores, além da matriz de transição.

```r

potM <- function(M, n){ #Funcao para calcular a n-esima potencia de uma matriz.
    if (n == 1) return (M)
    if (n > 1) return (M %*% potM (M, n-1))
}

autoM <- function(M, n){ #receba a matriz e a potencia a ser calculada pelo metodos dos autovalores.

    auto <- eigen(t(M)) #decomposicao espectral de uma matriz.
    Q <- as.matrix(auto$vectors) #atribuindo os autovetores a uma matriz de consulta.
    DK <- diag(auto$values^n) #matriz diagonal elevada a n eh igual ao seus elementos elevados a n

    return (t(Q%*%DK%*%solve(Q))) # onde solve irá inverter a matriz e %*% faz a multiplicacao de matriz.
}

M = matrix(                     #preenchendo os valores da matriz.
        c(0.3, 0.3, 0.4, 0.2, 0.6, 0.2, 0, 0, 1),
        nrow = 3,
        ncol = 3,
        byrow = TRUE
    )

cat("\nResultado obtido pelo metodo dos autovalores:", autoM(M,3)[1,3])
cat("\nResultado obtido pelo metodo das potencias:", potM(M,3)[1,3],"\n\n")
```

A partir da execução do código exemplificado na figura 1, obteve-se o seguinte resultado em ambos os métodos: $P = (X_3 = 2|X_0 = 0) = 0.694$. Porém dada a complexidade da operação de multiplicação de matriz é mais eficiente utilizar a técnica dos autovalores e autovetores.

2 Questão 2:

A resolução dessa questão envolve o modelo de Ehrenfest. Foi considerado durante a resolução que o número total de bolinhas na urna seriam 4. A partir disso foi solicitado o calculo das potências n=37,38,39,40 da matriz de transição.

Figura 2: Funções que geram a matriz do modelo para um valor qualquer, e a função iterativa que calcula a potenciação de matriz.

```r
ehrenfest <- function(max){ # a urna pode ter de zero a max estados sendo que max eh o numero de bolinhas

    M <- matrix (0, nrow = max+1, ncol = max+1)
    M[1,2] <- 1 #caso trivial, se nao tem bolinhas em um instante, no outro instante concerteza ira ter 1 com prob 1.
    M[max+1, max] <- 1 #caso trivial, se tem todas as bolinhas no proximo instante terá n-1 com prob 1

    for(i in 2: max){ #no prox passo, so pode ter uma bolinha a menos ou uma a mais, por isso que a matriz eh montada dessa maneira.
        M[i, i-1] <- (i-1)/max
        M[i, i+1] <- (max-i+1)/max
    }

    return (M)
}

potM <- function(M, n){ #funcao que faz a potencia das matrizes

    Maux <- M
    n = n-1

    while(n>0){
        Maux <- Maux%*%M
        n = n-1
    }
    return (Maux)
}
```

Figura 3: Saída da potenciação de matrizes (n=37,38,39,40)

```
M elevada a 37:
      [,1] [,2] [,3] [,4]  [,5]
[1,] 0.000  0.5 0.00  0.5 0.000
[2,] 0.125  0.0 0.75  0.0 0.125
[3,] 0.000  0.5 0.00  0.5 0.000
[4,] 0.125  0.0 0.75  0.0 0.125
[5,] 0.000  0.5 0.00  0.5 0.000

M elevada a 38:
      [,1] [,2] [,3] [,4]  [,5]
[1,] 0.125  0.0 0.75  0.0 0.125
[2,] 0.000  0.5 0.00  0.5 0.000
[3,] 0.125  0.0 0.75  0.0 0.125
[4,] 0.000  0.5 0.00  0.5 0.000
[5,] 0.125  0.0 0.75  0.0 0.125

M elevada a 39:
      [,1] [,2] [,3] [,4]  [,5]
[1,] 0.000  0.5 0.00  0.5 0.000
[2,] 0.125  0.0 0.75  0.0 0.125
[3,] 0.000  0.5 0.00  0.5 0.000
[4,] 0.125  0.0 0.75  0.0 0.125
[5,] 0.000  0.5 0.00  0.5 0.000

M elevada a 40:
      [,1] [,2] [,3] [,4]  [,5]
[1,] 0.125  0.0 0.75  0.0 0.125
[2,] 0.000  0.5 0.00  0.5 0.000
[3,] 0.125  0.0 0.75  0.0 0.125
[4,] 0.000  0.5 0.00  0.5 0.000
[5,] 0.125  0.0 0.75  0.0 0.125
```

Como pode-se notar, ao elevar a matriz M em potências impares grandes, como n=37 e n=39 nota-se que ela converge sempre para os mesmos valores. O mesmo fenômeno ocorre para potências pares grandes, como n=38 e n=40.

De acordo com a teoria vista, esse processo é de classe fechada, finita, recorrente e irredutível.

Os autovalores da transposta da matriz de transição são: M = [1, -1 , -0.5 ,0.5 ,1.343484e-16] obtidos através do comando: **eigen(t(M))$values**, e eles não tem relação com as potências calculadas.

A matriz desse problema não é regular, então não existe uma potência dela onde todos os seus elementos serão positivos e maiores que zero (isso pode ser facilmente verificado pois não tem como ela se manter em um mesmo estado durante uma transição). Além disso, podemos perceber que os seus autovalores não são todos positivos e maiores que zero, o que reforça que ela não é uma matriz regular. Portanto não apresenta distribuição estacionária. Calculando as potências como 2000 e 2001 observa-se que elas diferem entre si ($M^{2000} = M^{38} \neq M^{2001} = M^{39}$), caso o problema tivesse uma distribuição estacionária elas dariam resultados muito próximos, o que não ocorre.

Obs: Ao executar a função recursiva apresentada em sala **potM** para valores de n maiores que 1966 levava a um erro de estouro de pilha. Por isso utilizei uma versão iterativa dessa função durante esse exercício.

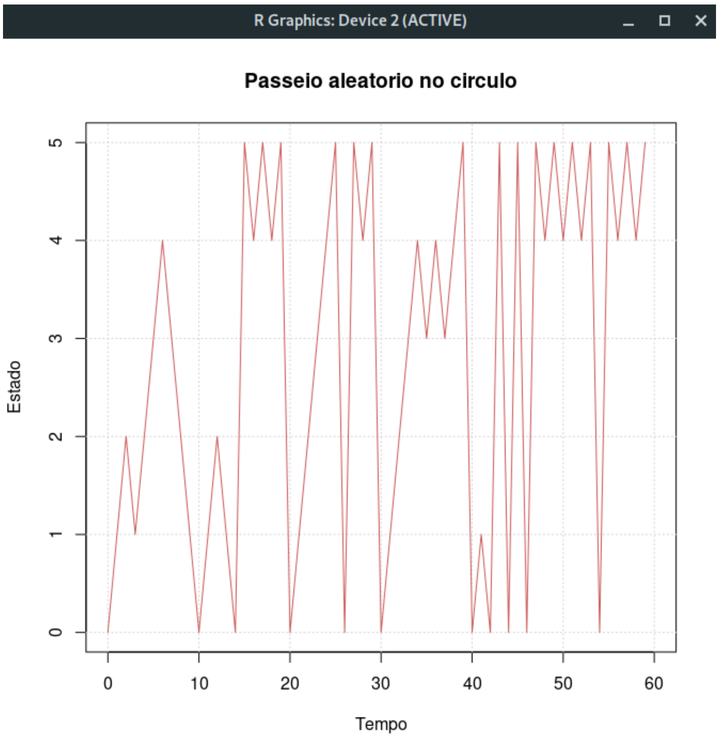
3 Questão 3:

Para essa questão foi construída a simulação de um processo de passeio em um círculo com prob de transição p=0.5 e na simulação consideramos 60 execuções.

Figura 4: Código que gera uma simulação de um passeio aleatório em um circulo com 6 estados, de zero a cinco

```r
passeio_circulo <- function(p, time){

    estados <- numeric() # vetor de estados
    estados[1] <- 0 # estado inicial

    for(i in 2:time){

        u <- runif(n=1, min = 0, max = 1) #gera em uma iteracao um valor entre (0,1)

        if(u<=p){
            if(estados[i-1] == 5){ # se tiver que avançar para frente e estiver em 5 volta para o zero
                estados[i] <- 0
            } else{
                estados[i] <- estados[i-1] + 1
            }
        }

        else{
            if(estados[i-1] == 0){ # se tiver que voltar um estado e estiver em zero vai para o cinco
                estados[i] <- 5
            } else{
                estados[i] <- estados[i-1] - 1
            }
        }
    }

    return (estados)
}

time <- 60
p <- 0.5
estados <- passeio_circulo(p, time)
```

Figura 5: Representação gráfica do passeio aleatório no circulo de 6 estados (0 a 5)



4 Questão 4:

Nessa questão foi solicitado executar 1000 iterações do paradoxo de partida, sendo que Catarina ia para o ponto inicialmente ás 0,45h em uma escala de 0 até 7 (5h-12h) segundo o que foi apresentado em sala.

Figura 6: Código que resolve o problema do paradoxo de partida

```r
catarinafunc <- function(t_catarina, t_end, lambda){

    times <- numeric(0)
    i <- 1
    current_time <- 0 #tempo comeca em zero

    while(current_time < t_end){
        times[i] <- current_time
        current_time <- current_time + rexp(1 , rate = 5.5) #tempo de saida dos onibus
        i <- i+1
    }
    index_catarina_bus <- min(which(times > t_catarina)) #descobrindo o onibus que a catarina pegou
    wait_time <- times[index_catarina_bus] - t_catarina
    interval_time <- times[index_catarina_bus] - times[index_catarina_bus-1] # tempo entre o onibus dela e o anterior

    return(list(interval_time,wait_time))
}

    wait_time <- 0
    for(i in 1:1000){
        data <- catarinafunc(0.75, 7, 5.5)
        wait_time <- wait_time + data[[2]]
    }
    cat("\nTempo de espera da Catarina:", wait_time/1000, "\n")

    interval <- 0
    wait_time <- 0
    for(i in 1:1000){
        data <- catarinafunc(3.5,7, 5.5)
        interval <- interval + data[[1]]
        wait_time <- wait_time + data[[2]]
    }
    cat("\nMedia dos intervalos:", interval/1000, "Tempo de espera da Catarina(x2):", wait_time/500, "\n\n")
```

No código acima, é calculado o tempo de espera até que a Catarina pegue o ônibus e o tempo entre o ônibus que ela pegou e o anterior.

Calculando a média dessas 1000 execuções, concluiu-se que Catarina espera 0.181093 antes de pegar o ônibus.

Já na segunda parte do problema, mesmo ela indo para o ponto no tempo t=3,5 ela esperou em média 0.1797265 antes de pegar o ônibus, valor próximo ao calculado anteriormente. Além disso, o dobro do tempo que ela esperou 0.359453 que é muito próximo a média de intervalos entre o ônibus que ela pega e o anterior (0.3578936).

5 Questão 5:

É passado dois conjuntos de dados bi-dimensionais e pede-se para fornecermos indícios se os dados se distribuem segundo processos de Poisson.

O primeiro indício que obtemos é através do gráfico, uma vez que em um processo de Poisson bi-dimensional os pontos se distribuem uniformemente, dessa forma, em ambos os gráficos pode-se notar uma distribuição uniforme o que deixa indícios de ser uma distribuição de Poisson.

Figura 7: Plot do primeiro conjunto de dados

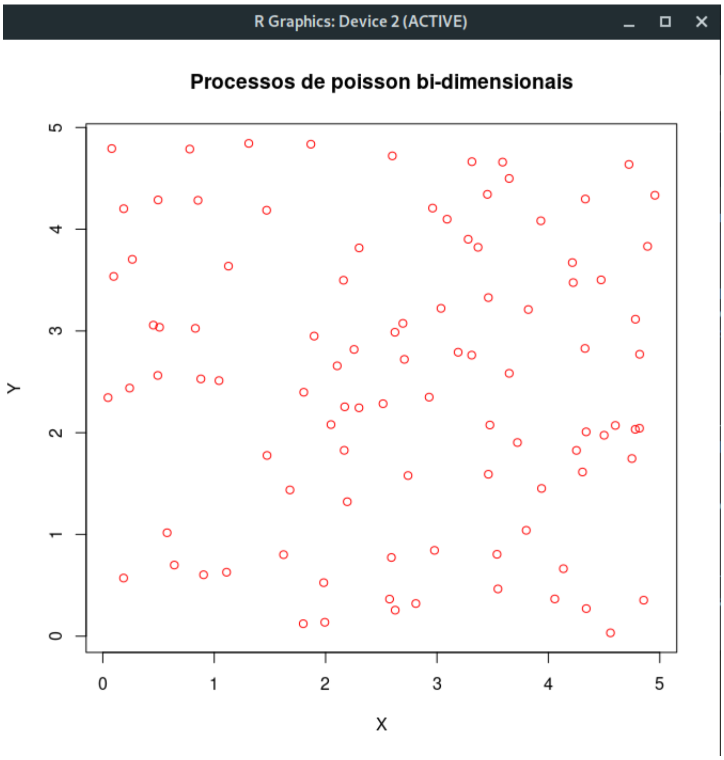
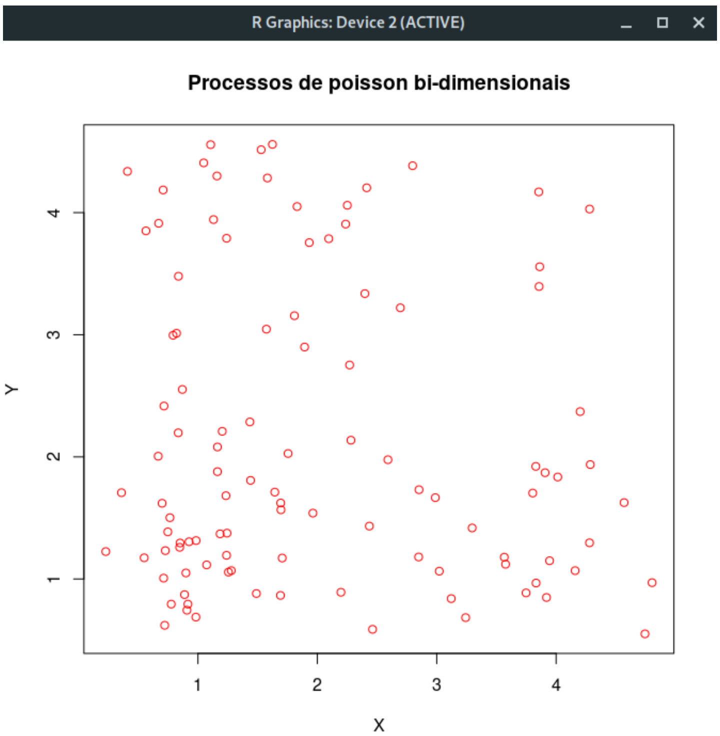


Figura 8: Plot do segundo conjunto de dados



Para as segundas justificativas foi fixado o lambda em 4, uma vez que dado a função que gera um processo de Poisson, para área 25 o numero de pontos era próximo ao numero de entradas no arquivo de dados. A partir disso, foi reduzida a área e observando o comportamento dos pontos.

Figura 9: Função que conta quantos elementos do conjunto de dados estão em um determinado subconjunto

```r
for(i in (0:4)){

    counter <- 0
    lim <- 5-i

    for(j in (1:length(xcsv))){
        if(xcsv[j] < lim && ycsv[j] < lim){
            counter = counter + 1
        }
    }

    cat("\n Area:",lim*lim,"Pontos:",counter,"Pontos estimados",lim*lim*4,"\n")

}
```

Figura 10: Resultado do código para o primeiro csv

```
Area: 25 Pontos: 98 Pontos estimados 100

Area: 16 Pontos: 59 Pontos estimados 64

Area: 9 Pontos: 35 Pontos estimados 36

Area: 4 Pontos: 11 Pontos estimados 16

Area: 1 Pontos: 3 Pontos estimados 4
```

Figura 11: Resultado do código para o segundo csv

```
Area: 25 Pontos: 104 Pontos estimados 100

Area: 16 Pontos: 82 Pontos estimados 64

Area: 9 Pontos: 54 Pontos estimados 36

Area: 4 Pontos: 35 Pontos estimados 16

Area: 1 Pontos: 6 Pontos estimados 4
```

Como pode-se observar, no primeiro conjunto de dados o número de pontos presentes em um determinado subconjunto é próximo ao real, o que dá outro indicio de se tratar de uma distribuição de Poission. Entretanto no segundo conjunto de dados o número de pontos fica bem longe em alguns casos, o que mostra que essa distribuição não é aleatória, portanto não é um processo de Poisson bi-dimensional. Sendo assim, a conclusão final desse exercício é que o primeiro conjunto de dados é oriundo de um processo de Poisson espacial, enquanto que o segundo não.