Estatística Numérica Computacional
Trabalho nº1

Marta Paz nº49861
Rafael Almeida nº49788
Rafael Gameiro nº50677
Ricardo Pinto nº49811

October, 2018

Exercício 1

Considere a variável aleatória X com função densidade de probabilidade

$$f(x) = \begin{cases} 3x^2, & 0 < x < 1 \\ \\ 0, & \text{para outros valores de x} \end{cases}$$

Considere agora a variável aleatória $Y = g(X)$, sendo $g(x) = log(2x^2 + 3)$. Estime $P(1 < Y < 1.5)$ usando o método de Monte Carlo e apresente também o desvio padrão da estimativa.

Resolução

1. Sabendo que $0 < x < 1$, pelo ramo da função onde x varia, e que g é crescente (isto porque a função logarítmica cresce ao tender para mais infinito), então podemos concluir que $g(0) < g(X) < g(1)$. De outro modo, substituindo as expressões pelos seus valores respetivos, temos: $1,099 < Y < 1,61$.
Assim:

$$\begin{aligned} P(1 < Y < 1.5) &= P(1 < Y < 1.099) + P(1.099 < Y < 1.61) \\ &= 0 + P(1.099 < Y < 1.61) = P(g^{-1}(1.099) < X < g^{-1}(1.61)) \end{aligned}$$

2. De seguida, calculámos o valor de $g^{-1}(x)$:

$$\begin{aligned} y = log(2x^2 + 3) \Leftrightarrow e^y = 2x^2 + 3 \Leftrightarrow x^2 = \frac{e^y - 3}{2} \\ \Leftrightarrow x = \sqrt{\frac{e^y - 3}{2}} \Leftrightarrow g^{-1}(x) = \sqrt{\frac{e^y - 3}{2}} \end{aligned}$$

3. Ficamos, assim, com o integral,

$$\int_{g^{-1}(1.099)}^{g^{-1}(1.61)} f(x)dx$$

onde aplicámos o método de Monte Carlo em R.

```r
#Exercicio 1
#numero de amostras
n=10000

invertg <- function(x){
  sqrt((exp(1)^x)-3)/sqrt(2)
}

#limites
a <- invertg(1.099)
b <- invertg(1.5)

func <- function(x){
if (0 < x && x < 1){
    3*x^2
  } else 0
}

u = runif(n, min =a, max = b)

amostra <- 1:n
for (i in 1:n){
  amostra[i] <- func(u[i])
}
prob <-(b-a)*mean(amostra)
sd <- sd(amostra)
hist(amostra)
```

R: $P(1 < Y < 1.5) \approx 0.64$ sd≈ 0.66

Exercício 2

Sejam X e Y duas variáveis aleatórias independentes ambas com função densidade de probabilidade, $f(x) = \frac{2}{\sqrt{2\pi}}.e^{-\frac{x^2}{2}}$, $0 < x < +\infty$, (ou seja, estas variáveis são o valor absoluto de uma normal standart) e seja $\theta = E(e^{X+Y})$.

a) Use o método de Monte Carlo para estimar θ, usando variáveis aleatórias com distribuição $Unif(0, 1)$.

Resolução

Para este exercício, efetuámos uma mudança de variável para que o integral que queremos calcular que é:

$$\int_0^{+\infty}\int_0^{+\infty} e^{x+y}dxdy$$

Para que o domínio de integração esteja entre 0 e 1, fizemos a seguinte conta:

$$u = \frac{1}{x+1} \Leftrightarrow (x+1)u = 1 \Leftrightarrow (x+1) = \frac{1}{u} \Leftrightarrow x = \frac{1}{u} - 1 \Leftrightarrow x = \frac{1-u}{u}$$

$$u(0) = \frac{1}{0+1} = 1 = b$$

$$u(+\infty) = \frac{1}{+\infty + 1} = 0 = a$$

Nota: a mudança de variável para o y é igual à de x, mas como x e y são variáveis independentes tem-se de usar uma letra diferente

$$\int_0^1 \int_0^1 e^{\left(\frac{2}{\sqrt{2\pi}}.e^{-\frac{\left(\frac{(1-u)}{u}\right)^2}{2}}\right)}.e^{\left(\frac{2}{\sqrt{2\pi}}.e^{-\frac{\left(\frac{(1-t)}{t}\right)^2}{2}}\right)}.-\frac{1}{u^2}.-\frac{1}{t^2}dudt$$

Como não conseguimos calcular este integral analiticamente vamos recorrer ao método de Monte Carlo para calcular uma estimativa para o seu valor.
Para tal, fizemos os seguintes passos em R:

1. Criámos uma função auxiliar que nos vai dar a fórmula da expressão que queremos calcular o estimador, e consequentemente, o integral
2. Criámos uma amostra aleatória para o valor de x e outra para o valor de y
3. Em seguida, calculámos o valor da expressão com os valores aleatórios e obtivemos o resultado do estimado, calculando a média de todos os valores

```r
#Exercicio 2 - alinea a)
a <- 0                                                   #limite inferior
b <- 1                                                   #limite superior

n <- 1000                                                #numero de amostras

func <- function(x,y){                                   #funcao a analisar
  u = 1/(x+1)                                            #mudancas de variaveis
  v = 1/(y+1)

  auxX= (2/sqrt(2*pi))*exp(-(((1-u)/u)^2)/2)
  auxY= (2/sqrt(2*pi))*exp(-(((1-v)/v)^2)/2)

  exp(auxX+auxY) * -1/(u^2) * -1/(v^2)
}


resultArea <- 1:n
amostra <- 1:n
xRand = runif(min=a,max=b,n=n)
yRand = runif(min=a,max=b,n=n)

for(j in 1:n){
  amostra[j] = func(xRand[j],yRand[j])
}

mean(amostra)                                             #valor esperado
sd(amostra)                                               #desvio padrão
```

R: ≈ 20.16

b) Reestime o parâmetro θ através do método de Monte Carlo usando agora variáveis aleatórias com outra distribuição, uma distribuição que considere adequada.

Resolução

1. Para este exercício, considerámos que a distribuição mais adequada para calcular as variáveis aleatórias seria utilizar a distribuição normal, uma vez que duas vezes a função densidade de probabilidade da normal é igual à função densidade de probabilidade das variáveis X e Y.

2. Assim sendo, no R, geramos 1000 valores aleatórios com distribuição uniforme para duas variáveis diferentes e de seguida aplicámos a tranformação de Box-Muller para obtermos distribuições normais. Depois, para cada valor gerado nas duas variáveis anteriores, calculámos $exp(Xi + Yi)$.

```r
a <- 0  #limite inferior no caso da distribuicao normal e a media
b <- 1  #limite superior no caso da distribuicao normal e a variancia

n <- 1000 #numero de amostras

#nao e preciso meter aqui a funcao pois variaveis geradas sobre essa funcao ja sofrem a funcao utilizada

amostra <- 1:n
#for (i in 1:n){
xRand = runif(min=0, max=1, n)    #amostra de 1000 elementos entre a e b (esta dentro do for pois vamos mudar os valores por cad
yRand = runif(min=0, max=1, n)


func <- function(x){
  2*x
}

#gerar uma amostra que segue a distribuicao normal apartir de 2 amostras q seguem uma distribuicao uniforme
z1 = sqrt(-2*log(xRand))*cos(2*pi*yRand)
z2 = sqrt(-2*log(xRand))*sin(2*pi*yRand)


for(j in 1:n){
  amostra[j] = exp(func(z1[j]) + func(z2[j]))
}

mean(amostra)    #valor esperado
sd(amostra)     #desvio padrão
hist(amostra)   #histograma
```

R: $\theta \approx 35.6$

c) Considere o estimador de Monte Carlo da alínea (a), $\hat{\theta}$ e seja $C = \overline{UV}$, ou seja, $C = \frac{1}{N} \sum_{i=1}^{n} U_i V_i$ uma variável de controlo, em que U e V são variáveis aleatórias com distribuição Unif(0,1). Reestime θ usando esta variável de controle. Estime o desvio-padrão e compare os desvios-padrão dos dois estimadores. Discuta os resultados numéricos e analíticos.

Resolução

1. Para resolver este exercício, utilizámos a seguinte fórmula para reestimar a variável

$$\hat{\theta}c = \hat{\theta} - \beta * (C - E(u))$$

2. Para tal, utilizámos parte do código feito no exercicio (a) e gerámos uma variável de controle apartir da função generateControl implementada no R.

3. De seguida, tivemos de calcular um β. Para calcular este valor, usámos a seguinte fórmula:

$$\beta = \frac{(Cov(\hat{\theta}, C))}{Var(c)}$$

Para calcular o beta, criámos uma função no R, que calcula a covariância entre a variavel de controle dada e a amostra calculada do exercicio a). Para além do cálculo da covariância, também tivemos de calcular a variância da variável de controle, C.

4. De seguida, gerámos uma nova amostra seguindo a seguinte formula:

$$finalC[i] = amostra[i] - (\beta * (c[i] - mean(c)))$$

5. A média dessa amostra é o estimador que queremos calcular que terá o seguinte valor: $\hat{\theta}c = 19.98$

6. O valor do desvio padrão de $\hat{\theta}c = 8.014776$ e o valor do desvio padrão de $\hat{\theta} = 8.017608$, sendo que obtivemos o mesmo valor para o valor do estimador.

7. Comparando os valores apresentados no ponto 6, podemos verificar que, usando a variável de controlo, o valor do estimador fica igual ao valor do estimador quando não utilizávamos a variável de controlo, mas, em contrapartida, o valor do desvio padrão desce ligeiramente quando o estimador é calculado com a variável de controlo. Assim, podemos concluir que o cálculo de um estimador é mais preciso e/ou as variáveis aleatórias geradas estão mais próximas do valor real.

```r

#Exercicio 2 - alinea c)
a <- 0                                                      #limite inferior
b <- 1                                                      #limite superior

n <- 1000                                                   #numero de amostras


func <- function(x,y){                                      #funcao a analisar
  u = 1/(x+1)                                               #mudanca de variavel
  v = 1/(y+1)                                               #mudança de variavel

  auxX= (2/sqrt(2*pi))*exp(-(((1-u)/u)^2)/2)
  auxY= (2/sqrt(2*pi))*exp(-(((1-v)/v)^2)/2)

  exp(auxX+auxY)  * -1/(u^2) * -1/(v^2)
}

generateVarControl <- function(){                           #funcao que gera a amostra de controlo
  nctrl = 1000
  c = runif(min=0, max=1, nctrl)
  v = runif(min=0, max=1, nctrl)

  aux <- 1:nctrl
  for (i in 1:nctrl){
    aux[i] = c[i]*v[i]
  }
  aux
}


resultArea <- 1:n
amostra <- 1:n
xRand = runif(min=a,max=b,n=n)                              #amostra de 1000 elementos entre a e b
```

```r

for(j in 1:n){
  amostra[j] = func(xRand[j],yRand[j])
}

c = generateVarControl()                                    #amostra de controlo
teta_chapeu = amostra                                       #valores tirado do exercio 2a)
beta = cov(x=teta_chapeu, y=c)/var(c)                       #beta minimo

finalC = 1:n
for(i in 1:n){                                              #geracao da amostra final usando a variavel de controlo
  finalC[i]=amostra[i] - (beta*(c[i]-mean(c)))
}

sd(amostra)                                                 #sd inicial
mean(amostra)                                               #estimador inicial

sd(finalC)                                                  #sd final
mean(finalC)                                                #estimador final
```