Estatística Numérica Computacional
Trabalho nº2
Grupo V

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

October, 2018

Exercício 1

Considere as duas amostras de duas v. aleatórias X e Y

Amostras de X	92	90	85	96	92	88	96	88
Amostras de Y	89	90	88	93	90	85	95	90

(a) Use $\hat{\rho} = \frac{\sum_{i=1}^{N}(X_i-\bar{X})(Y_i-\bar{Y})}{\sqrt{\sum_{i=1}^{N}(X_i-\bar{X})^2 \sum_{i=1}^{N}(Y_i-\bar{Y})^2}}$ para estimar o coeficiente de correlação entre as duas variáveis.

Resolução

A resolução desta alínea foi feita somente em R.

```r
#*************************************************************************************************

#EXERCICIO 1

x <- c(92,90,85,96,92,88,96,88)
y <- c(89,90,88,93,90,85,95,90)

alpha <- 0.025
n <- 999


#EXERCICIO 1 - ALINEA A

# Calculo de Pearson Coefficient
func <- function(x,y){
  sum((x-mean(x))*(y-mean(y)))/sqrt((sum((x-mean(x))^2))*(sum((y-mean(y))^2))
}

pCoeficient = func(x,y)

```

R: $\hat{\rho} \approx 0.798$

(b) Calcule a variância de jackknife de $\hat{\rho}$.

Resolução

Para calcular a variância de jackknife temos a seguinte fórmula:

$$Var_{jack} = \frac{1}{n(n-1)}(\sum_{j=1}^{N} l_{jack}^2 - nb_{jack}^2)$$

Para tal, também precisamos de calcular o enviesamento.

Definição

Suponhamos que $\rho = t(F)$ e que F é uma função de distribuição cumulativa e $t(.)$ algo funcional. Então, a função de influência de t em F é dada por:

$$L_t(y, F) = \lim_{\varepsilon \to 0} \frac{t\left[(1-\varepsilon)F + \varepsilon H_y\right] - t(F)}{\varepsilon}$$

e

$$H_y(u) = \begin{cases} 0, & u < y \\ \\ 1, & u \geq y \end{cases}$$

Suponhamos que $x_1, x_2, ..., x_n$ é uma amostra e $\hat{F}$ é uma função empírica da amostra. A função de influência empírica é definida como:

$$l(y) = L_t(y, \hat{F})$$

Os valores da função de influência empírica nos pontos de dados são chamadas de valores de influência empírica.

$$l_j = l(x_j) = L_t(x_j, \hat{F}), \quad j = 1, 2, ..., n$$

Uma extensão do Teorema de Taylor diz que para funções t e medidas G e F,

$$t(G) \approx t(F) + \int L_t(y, F) dG(y)$$

Isto é um resultado exato se t é uma estatística linear. Aplicando essa fórmula com F à função de distribuição cumulativa e à função $G = \hat{F}$ obtemos

$$t(\hat{F}) \approx t(F) + \int L_t(y, F) d\hat{F}(y) \quad \text{,que é}$$

$$t(\hat{F}) \approx t(F) + \frac{1}{n} \sum_{j=1}^{N} L_t(x_j, \hat{F}) = t(F) + \frac{1}{n} \sum_{j=1}^{N} l_j$$

$$\Leftrightarrow \rho - \hat{\rho} = -\frac{1}{n} \sum_{j=1}^{N} l_j$$

O Jackknife fornece uma maneira de aproximar valores de influência empíricos através da reamostragem dos dados.

$$L_t(y,F) = \lim_{\varepsilon \to 0} \frac{t\left[(1-\varepsilon)\hat{F} + \varepsilon H_y\right] - t(\hat{F})}{\varepsilon}$$
$$\approx \frac{t\left[(1-\varepsilon)\hat{F} + \varepsilon H_y\right] - t(\hat{F})}{\varepsilon}$$

Se tomarmos $\varepsilon = \frac{1}{n-1}$, então

$$(1-\varepsilon)\hat{F} + \varepsilon H_{x_j} = \frac{n}{n(n-1)}\hat{F} - \frac{1}{n-1}H_{x_j} = \hat{F}_{-j}$$

é uma distribuição sem peso no ponto x_j e peso $\frac{1}{n-1}$ no resto da amostra. Isto é equivalente a ter apenas a amostra de tamanho $n-1$ encontrada omitindo x_j da amostra original. Então, a aproximação Jackknife ao valor da influência empírico l_j é

$$l_{jack:j} = (n-1)[t(\hat{F} - t(\hat{F}_{-j}))] = (n-1)(\hat{\rho} - \rho_{-j})$$

As estimativas imparciais do enviesamento e da variância, que usam os valores de influência empíricos do Jackknife a que chamamos de viés de Jackknife e variância de Jackknife, são:

$$b_{jack} = -\frac{1}{n}\sum_{j=1}^{N} l_{jack:j}$$
$$Var_{jack} = \frac{1}{n(n-1)}(\sum_{j=1}^{N} l_{jack:j}^2 - n b_{jack}^2)$$

```r
#*****************************************************************************************************************************************************

#EXERCICIO 1 - ALINEA B

# Criacao das matrizes xAux e yAux
# Ciclo for que preenche as matrizes xAux e yAux com os valores "teta_chapeu exceto o elemento na posicao j"
xAux <- matrix(nrow=8, ncol = 7)
yAux <- matrix(nrow=8, ncol = 7)

for(l in 1:8){
  tempX=x[-l]     # Remocao do elemento l do vetor x, com posterior criacao do vetor tempX ja sem esse elemento
  tempY=y[-l]     # Remocao do elemento l do vetor y, com posterior criacao do vetor tempY ja sem esse elemento

  for(r in 1:7){
    xAux[row=l,col=r]=tempX[r]
    yAux[row=l,col=r]=tempY[r]
  }
}


# Calculo do lJack_j: (n-1) * (teta_chapeu - teta_chapeu_excetoJ)
# Seja n a dimensao das amostras, teta_chapeu o valor de pCoeficient calculado na alinea a)
# Consequente calculo do somatorio do lJack de todas as posicoes
lJack = 0
somaBjack=0

for(i in 1:8){
  lJack = 7 * (pCoeficient - func(xAux[i, ], yAux[i, ]))
  somaBjack = somaBjack + lJack
}


# Calculo do bJack: -(1/n) * somatorio(lJack_j)
bJack = -(1/8) * somaBjack


# Calculo do varJack: (1/(n(n-1))) * (somatorio(lJack^2 - n*bJack^2))
somaVarJack=0

for (i in 1:8){
  lJack = 7 * (pCoeficient - func(xAux[i, ], yAux[i, ]))
  somaVarJack = somaVarJack + (lJack^2)
}

varJack = (1/(8*(8-1))) * (somaVarJack-(8*(bJack^2)))

```

R: $Var_{jack} \approx 0.014$

(c) Construa um intervalo de confiança bootstrap básico para ρ.

Resolução

Para calcular um intervalo de confiança de Bootstrap tivémos de gerar várias amostras bootstrap, tanto para X como para Y. Para tal, aplicámos a cada amostra a fórmula do coeficiente de correlação, e de seguida o enviesamento a cada resultado obtido. Ordenámos os valores e utilizando a estatística pivot,

$$\hat{a}_\alpha = \hat{\rho}^*_{(R+1).\alpha} - \hat{\rho}$$
$$\hat{a}_{1-\alpha} = \hat{\rho}^*_{(R+1).(1-\alpha)} - \hat{\rho}$$

obtivémos o intervalo de confiança na forma,

$$]\hat{\rho} - \hat{a}_{1-\alpha}, \hat{\rho} - \hat{a}_\alpha[$$

```r
#*****************************************************************************************************************************************************

#EXERCICIO 1 - ALINEA C

# Geracao de amostras Bootstrap
amBootsX <- matrix(NA, nrow=n,ncol = 8)
amBootsY <- matrix(NA, nrow=n,ncol = 8)

for(r in 1:n) {
  amBootsX[r,] <- sample(x, replace = TRUE)
  amBootsY[r,] <- sample(y, replace = TRUE)
}


# Geracao de amostras auxiliares
amostra_Vies <- 1:n
amostra_Ro <- 1:n

for(r in 1:n) {
  amostra_Ro[r] <- func(amBootsX[r,], amBootsY[r,])
  amostra_Vies[r] <- amostra_Ro[r] - pCoeficient
}


# Calculo do Intervalo de Confiança
# Limite Inferior: teta_chapeu - a_1menosAlpha
# Limite Superior: teta_chapeu - a_alpha
# Seja o alpha = 0.025

# Passo 1: Gerar amostra usada para calcular o Intervalo de Confianca
# Seja amIC a amostra do Intervalo de Confianca
amIC <- 1:n+1
for(r in 1:r) {
  amIC[r] <- amostra_Vies[r]
}

amIC[n+1] <- 0
amIC <- sort(amIC)

# Passo 2: Calculo do a_alpha e do a_1menosAlpha
a_alpha <- amIC[(n+1)*alpha]
a_1menosAlpha <- amIC[(n+1)*(1-alpha)]

# Passo 3: Estabelecer limites inferior e superior
limite_inferior_Boots <- pCoeficient-a_1menosAlpha
limite_superior_Boots <- pCoeficient-a_alpha

```

R: $]0.86, 2.33[$

(d) Construa um intervalo de confiança t-bootstrap para ρ. Use a variância bootstrap para estimar a variância de ${\hat{\rho}_r}^*$.

Resolução

Para o cálculo do intervalo de confiança bootstrap studentized utilizámos uma estatística pivot diferente da usada no cálculo do bootstrap básico:

$$\hat{z} = \frac{\hat{\rho}^* - \hat{\rho}}{\sqrt{Var_{Boots}(\hat{\rho}^*)}}$$

$$Var_{Boots}(\hat{\rho}^*) = \frac{1}{n-1}\sum_{j=1}^{N}({\hat{\rho}_j}^* - \bar{\hat{\rho}}^*)^2$$

Primeiramente, usámos as amostras bootstrap geradas da alínea c) para o cálculo dos $\hat{\rho_r}^*$. Para obtermos a variância bootstrap de cada $\hat{\rho_r}^*$, gerámos um novo conjunto de amostras bootstrap com base em cada amostra previamente usada na alínea anterior.
O resto do procedimento, foi a reordenação dos diferentes valores de z, e a determinação do intervalo de confiança, usando a fórmula

$$]\hat{\rho} - \hat{z}^*_{(R+1)(1-\alpha)}\sqrt{Var_{Boots}(\hat{\rho})}, \hat{\rho} - \hat{z}^*_{(R+1)(\alpha)}\sqrt{Var_{Boots}(\hat{\rho})}[$$

```r
#EXERCICIO 1 - ALINEA D

# Calculo do theta da amostra
amRho <- 1:n


for(r in 1:n){
  amRho[r] <- func(amBootsX[r,], amBootsY[r,])
}

meanAmRho <- mean(amRho, na.rm = TRUE)


# Calculo do varBoots: (1/(r-1)) * (somatorio( (teta_chapeu_estrela_R - media_teta_chapeu_estrela)^2) )
# Seja r a dimensao n da amostra bootstrap
varboots <- (1/(n-1))*sum((amRho - meanAmRho)^2, na.rm = TRUE)


# Criacao de uma matriz com os elementos das amostras das variancias
# Calculo de todos os z's e suas variancias
ryr <- 1:8
rxr <- 1:8
z <- 1:(n+1)
amRhoStar <- 1:n

for(r in 1:n){
  for(i in 1:n){
    rxr <- sample(amBootsX[r, ], replace = TRUE)
    ryr <- sample(amBootsY[r, ], replace = TRUE)

    amRhoStar[i] <- func(rxr, ryr)
  }

  meanAmRhoStar <- mean(amRhoStar, na.rm = TRUE)

  # Calculo da variancia para o z atual
  varbootsStar <- (1/(n-1))*sum((amRhoStar - meanAmRhoStar)^2, na.rm = TRUE)
  z[r] <- (amRho[r] - pCoeficient)/sqrt(varbootsStar)
}


z[n+1] <- 0
amIC <- 1:(n+1)


# Calculo do Intervalo de Confiança
# Limite Inferior: teta_chapeu - z_estrela_((R+1)(1menosAlpha)) * raiz(variancia)
# Limite Superior: teta_chapeu - z_estrela_((R+1)*alpha) * raiz(variancia)
amIC <- sort(z)

limite_inferior_tBoots <- pCoeficient - (amIC[(n+1)*(1-alpha)]*sqrt(varboots))
limite_superior_tBoots <- pCoeficient - (amIC[(n+1)*alpha]*sqrt(varboots))
```

R: $]0.89, 2.27[$

Discussão

Comparativamente com o método Bootstrap, o método Jackknife é mais rápido a executar e permite calcular estatísticas não-paramétricas. Por outro lado, uma vez que o intervalo de confiança baseia-se em aproximações, o melhor método a ser usado é o de Bootstrap.

O intervalo de confiança bootstrap studentized tem menor amplitude que o intervalo de confiança bootstrap básico. Uma vez que quanto menor for a amplitude de um intervalo, maior será a sua precisão, então podemos concluir que o intervalo de confiança bootstrap studentized é mais preciso que o de bootstrap básico. O intervalo de confiança bootstrap básico não se encontra como seria esperado, pois o valor ρ calculado anteriormente não pertence a esse intervalo. O intervalo de confiança bootstrap studentized também não está como era esperado, pois o valor ρ calculado anteriormente não pertence a esse intervalo.

Exercício 2

Considere a amostra seguinte:

1	3	3	1	1	3	2	2	3	0	2	4	2	6	4	2	4	3	2	4

Considere a hipótese nula, H_0 : A v.a. subjacente a esta amostra tem distrbuição binomial de parâmetros $(8, 0.3)$. Use a função de distribução empírica da estatística de teste para estimar o $p-value$ da estatística de teste qui-quadrado, T que permite testar a hipótese nula.

$$T = \sum_{i=1}^{k} \frac{(N_i - np_i)^2}{np_i}$$

$i \in \{1, ..., K\}$ - suporte da v.a.
$\{p_i, i = 1, ..., K\}$ - função de probabilidade da v.a.
n - dimensão da amostra.
N_i - número de observações na amostra que tomam o valor i.

Resolução

Neste exercício, foi-nos dada uma amostra e a seguinte hipótese nula:

H_0: A variável aleatória subjacente a esta amostra tem distribuição binomial de parâmetros $(8, 0.3)$.

De seguida, era pedido que usássemos a função de distribuição empírica da estatística de teste para estimar o $p-value$ da estatística de teste qui-quadrado, T, que permite testar a hipótese nula fornecida.
Desta forma, para o teste de hipótese que realizámos, cosiderámos as seguintes hipóteses:

H_0: A variável aleatória subjacente a esta amostra tem distribuição binomial de parâmetros $(8, 0.3)$.

H_1: A variável aleatória subjacente a esta amostra não tem distribuição binomial de parâmetros $(8, 0.3)$

Para calcular o valor do p-value, utilizámos a seguinte fórmula:

$$p - value = \hat{P}(T > t_{obs}|H_0) = 1 - \hat{F}_{H_0}(t_{obs}) = \frac{\#\{t_j : t_j \geq t_{obs}\} + 1}{m + 1}$$

Para podermos utilizar a fórmula acima, tivemos de calcular os seguintes valores:

1. t_{obs}, que corresponde ao valor da estatística de teste dos valores observados (valores estes que foram dados no enunciado). Recorremos à seguinte fórmula para o cálculo da estatística de teste (que foi dada no enunciado do problema):

   $$T = \sum_{i=1}^{k} \frac{(N_i - np_i)^2}{np_i}$$

   E realizámos os seguintes passos no R:

   (a) Percorremos a amostra dada no enunciado e contámos o número de ocorrências de cada elemento (0 a 6, pois a amostra só tem elementos compreendidos entre esses valores). Para guardar o número de ocorrências usámos um vetor, nObs, em que cada posição correspondia a um valor do intervalo entre 0 e 6.

   (b) Fizémos um ciclo onde é calculado a probabilidade binomial, com os parâmetros fornecidos, de cada ocorrência.

   (c) Por último, aplicámos a fórmula da estatística de teste acima ao conjunto das probabilidades calculadas no ponto anterior e daqui obtivémos que $t_{obs} = 2.205978$.

2. t_j, que corresponde ao valor da estatística de teste assumindo que a hipótese nula é verdade. Para tal, criámos uma matriz, amAux, onde cada linha vai corresponder a uma amostra gerada com o comando *rbinom* do R usando os parâmetros da distribuição binomial que nos deram no enunciado.
   De seguida criámos uma matriz, nObsAux, onde guardámos o número de ocorrências decada, em cada amostra gerada. Depois, calculámos a probabilidade de cada posição de nObsAux acontecer e, com os valores obtidos, pudémos calcular os t_j's. Para cada t_j, comparámos o seu valor com t_obs e contámos quantas vezes t_j era igual ou superior que t_{obs}. Esse valor foi guardado na nossa variável *soma*.

3. Por último, aplicámos a fórmula do $p - value$ aos valores que obtivémos, ou seja:

$$p - value = \frac{soma + 1}{(999 + 1) + 1} \approx 0.86$$

Para verificar se aceitamos a hipótese nula, considerámos três valores diferentes para o nivel de significância: $\alpha = 0.01$, $\alpha = 0.025$ e $\alpha = 0.5$.
Para os três valores de alfa considerados, verificámos que $p - value \geq \alpha$, logo não rejeitamos a hipótese nula nos três níveis de significância. Isto é, existem evidências estatísticas para afirmar que a variável aleatória subjacente à amostra fornecida tem distribuição binomial de parâmetros $(8, 0.3)$.

```r
#EXERCICIO 2


amostra <- c(1,3,3,1,1,3,2,2,3,0,2,4,2,6,4,2,4,3,2,4)
nObs <- c(1,3,6,5,4,1) # Numero de vezes que os valores 0, 1, 2, 3, 4 e 6 se repetem, respetivamente
k <- 6


# Funcao Auxiliar: Funcao Fatorial
fact <- function(x) {
  if(x == 0) {
    1
  }else {
    x * fact(x-1)
  }
}


# Calculo da Probabilidade da Distribuicao Binomial: probBinomial = (n!/(k!(n-k)!)) * (p^k) * ((1-p)^(n-k))
# Seja B(n,p) = B(8,0.3), temos n=8 e p=0.3.
# Seja k cada valor de nObs
probBinomial <- 1:6
for(i in 1:k) {
  probBinomial[i] <- (fact(8)/(fact(nObs[i])*fact(8-nObs[i]))) * (0.3^nObs[i]) * ((0.7)^(8-nObs[i]))
}


# Calculo do T, usando a funcao de probabilidade dada: estaTest = somatorio( ((N_i-n*p_i)^2) / (n*p_i) )
# Seja N_i cada valor de nObs, n a dimensao da amostra e p_i cada valor da funcao de probabilidade
estaTest <- 0
for(i in 1:k) {
  estaTest <- sum( ((nObs[i]-20*probBinomial[i])^2) / (20*probBinomial[i]) )
}
```

```r
# Calculo do p-value: (#{tj : tj>=tobs} + 1) / (m+1)
# Seja tj cada valor da amostra inicial, tobs a estatistica de teste e m a dimensao de nObs
soma=0
probAux <- 1:p
amAux <- matrix(nrow=(n+1), ncol=p)

#geracao de 1000 amostra com probabilidade binomial (8,0.3)
for(i in 1:(n+1)) {
  probAux <- rbinom(8,8,0.3)
  for(r in 1:p){
    amAux[i,r] <- probAux[r]
  }
}


#Calcular o numero de ocorrencias de cada valor na amostra gerada
nObsAux <- matrix(nrow=(n+1), ncol=9)
for(r in 1:(n+1)){
  for(c in 1:9){
    nObsAux[r,c] <- 0
  }
}
```

```
for(j in 1:(n+1)){
  for(i in 1:p){
    if(amAux[j,i] == 0){
      nObsAux[j, 1] = nObsAux[j,1]+1
    }
    else if(amAux[j,i] == 1){
      nObsAux[j, 2] = nObsAux[j,2]+1
    }
    else if(amAux[j,i] == 2){
      nObsAux[j, 3] = nObsAux[j,3]+1
    }
    else if(amAux[j,i] == 3){
      nObsAux[j, 4] = nObsAux[j,4]+1
    }
    else if(amAux[j,i] == 4){
      nObsAux[j, 5] = nObsAux[j,5]+1
    }
    else if(amAux[j,i] == 5){
      nObsAux[j, 6] = nObsAux[j,6]+1
    }
    else if(amAux[j,i] == 6){
      nObsAux[j, 7] = nObsAux[j,7]+1
    }
    else if(amAux[j,i] == 7){
      nObsAux[j, 8] = nObsAux[j,8]+1
    }
    else if(amAux[j,i] == 8){
      nObsAux[j, 9] = nObsAux[j,9]+1
    }
  }
}


```

```
estaTestAux <- 0
probBinomialAux <- 1:9


#faz o calculo de soma = #{tj : tj>=tobs}
for(i in 1:(n+1)){

  #calcula a probabilidade da ocorrencia de cada elemento gerado, assumindo que H0 e verdade
  for(j in 1:9){
    if(nObsAux[i,j] != 0){
      probBinomialAux[j] <- (fact(8)/(fact(nObsAux[i,j])*fact(8-nObsAux[i,j])))*(0.3^nObsAux[i,j])*((0.7)^(8-nObsAux[i,j]))
    }else{
      probBinomialAux[j] <- 0
    }
  }

  for(j in 1:9){
    if(probBinomialAux[j] == 0){
      estaTestAux <- estaTestAux + 0
    }else{
      estaTestAux <- sum(((nObsAux[i,j]-20*probBinomialAux[j])^2)/(20*probBinomialAux[j]))
    }
  }

  if(estaTestAux >= estaTestObs){
    soma = soma+1
  }

  estaTestAux <- 0
}


# calcula o pvalue = soma/m+1, com m = numero de elementos gerados
pvalue <- (soma+1)/(n+2)

```