Cálculo Numérico - 2021.2 - Tarefa 5 - Professor João Paixão

Realizado por: David Rodrigues Albuquerque - 120047390

Questões discutidas com: Beatriz Almeida Ramos, Carlos Bravo, Matheus Barroso

Exercício 1

Para este exercício, devemos determinar uma curva na forma $T = c_0 x^{c_1}$ o qual melhor se ajusta nos dados fornecidos utilizando o método de mínimos quadrados com coeficientes não-lineares. Em seguida, devemos utilizar o modelo estipulado para calcular T(0.3) com 3 casas decimais.

No primeiro passo, podemos importar de nossa biblioteca as funções vandermonde e regressão para calcular o sistema correspendente utilizando mínimos quadrados com coeficientes não-lineares. Para isto, utilizaremos as entradas x e y na forma $\overline{x} = ln(x)$ e $\overline{y} = ln(y)$ para fazer a transformada do caso não-linear para linear.

In [ ]:
```
using Plots
using LinearAlgebra
```

In [ ]:
```
function vandermonde(x,y,grau)
    n,=size(y)
    V=zeros(n,grau+1)
    for i=1:n #linhas
        for j=1:(grau+1)
            V[i,j]=x[i]^(j-1)
        end
    end
    return V
end
```

```
vandermonde (generic function with 1 method)
```

In [ ]:
```
function regressao(x,y,grau)
    V=vandermonde(x,y,grau)
    c=V\y #mínimos quadrados
    return c, grau
end
```

```
regressao (generic function with 1 method)
```

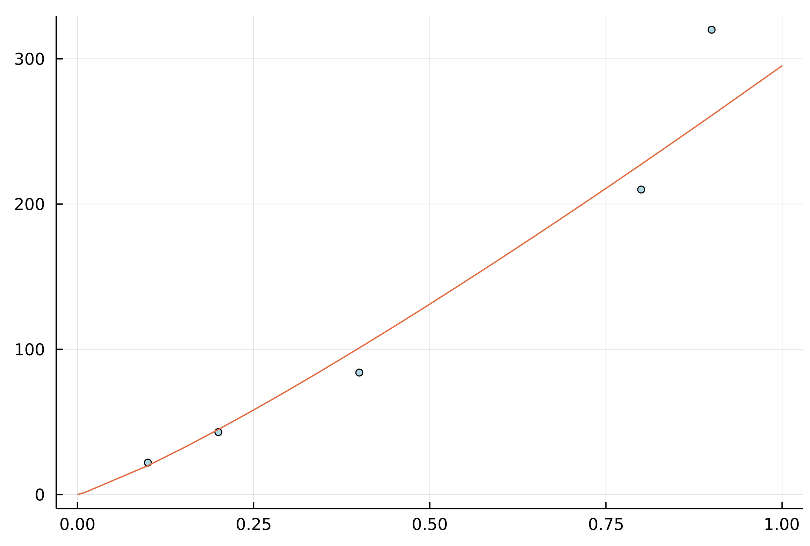
In [ ]:
```
function aproxima_funcao(x,y)
    x_barra = log.(x) # ln(x)
    y_barra = log.(y) # ln(y)
    coefs = regressao(x_barra,y_barra,1)
    return coefs
end
```

```
aproxima_funcao (generic function with 1 method)
```

Com isto, podemos utilizar a função aproxima função para determinar os coeficientes da aproximação da função que desejamos, atentando-se ao fato de retorná-los de volta ao não-linear utilizando a inversa da função logaritmo natural. Como calculamos os valores de x na forma $\overline{x}$, só precisaremos aplicar a inversa em x,

In [ ]:
```
x = [0.1,0.2,0.4,0.8,0.9]
y = [22,43,84,210,320]

coefs, _ = aproxima_funcao(x,y)
c0 = exp(coefs[1])
c1 = coefs[2]
f(x) = (c0)*(x^(c1))
scatter(x, y, c=:lightblue, ms=3, leg=false)
plot!(f,0,1)
```



Com isto, podemos calcular $T(0.3)$.

In [ ]:
```
round(f(0.3), digits=3)
```

```
72.061
```

Exercício 2.a)

Neste exercício, devemos utilizar os pontos descritos para encontrar uma função $f(x)$ que satisfaça os pontos dados e estimar a distância percorrida pela ônibus em $t = 125$. Isto é, encontrada a função, devemos calcular $\int_0^{125} (f(x) + E_\epsilon(x))\,dx$ pois, integrando o gráfico velocidade, teremos a distância percorrida.

Dados os pontos utlizados, podemos importar de nossa biblioteca a função de interpolação polinomial para encontrar a função descrita em cada um dos intervalos.

In [ ]:
```
function interpolacao_onibus(x,y,grau)
    # Cria a matriz V
    V = vandermonde(x,y,grau)
    c=V\y # Resolve o sistema linear Vc=y
    return c #vetor de coeficientes
end
```

```
interpolacao_onibus (generic function with 1 method)
```

In [ ]:
```
function trapezio(f,a,b,n) #calcular a integral f(x) de a até b
    h=(b-a)/n
    S=0.0
    for i=1:(n-1)  #calcula o "meio"
        x=a+i*h
        S+=2*f(x)
    end
    S=h/2*(S+f(a)+f(b)) #calcula "as pontas"
    return S
end
```

```
trapezio (generic function with 1 method)
```

Com isto, podemos utilizar o método do trapézio em cada intervalo (x_n, x_{n+1}) para os valores descritos. Após, basta somar as áreas encontradas.

In [ ]:
```
using Polynomials
```

In [ ]:
```
altura = 0.0
x = [0,10,15,20,32,59,62,125]
y = [0,185,319,447,742,1325,1445,4151]
xn = [0,10]
yn = [0,185]
coefs = interpolacao_onibus(xn,yn,1)
p = Polynomial(coefs)
altura += trapezio(p,xn[1],xn[2],1000)
xn = [10,15]
yn = [185,319]
coefs = interpolacao_onibus(xn,yn,1)
p = Polynomial(coefs)
altura += trapezio(p,xn[1],xn[2],1000)
xn = [15,20]
yn = [319,447]
coefs = interpolacao_onibus(xn,yn,1)
p = Polynomial(coefs)
altura += trapezio(p,xn[1],xn[2],1000)
xn = [20,32]
yn = [447,742]
coefs = interpolacao_onibus(xn,yn,1)
p = Polynomial(coefs)
altura += trapezio(p,xn[1],xn[2],1000)
xn = [32,59]
yn = [742,1325]
coefs = interpolacao_onibus(xn,yn,1)
p = Polynomial(coefs)
altura += trapezio(p,xn[1],xn[2],1000)
xn = [59,62]
yn = [1325,1445]
coefs = interpolacao_onibus(xn,yn,1)
p = Polynomial(coefs)
altura += trapezio(p,xn[1],xn[2],1000)
xn = [62,125]
yn = [1445,4151]
coefs = interpolacao_onibus(xn,yn,1)
p = Polynomial(coefs)
altura += trapezio(p,xn[1],xn[2],1000)
round(altura, digits=3)
```

```
219567.5
```

Com isso, temos que o ônibus espacial endeavour, no momento $t = 125$ estava na altura de aproximadamente 219567.5 pés, equivalente a aproximadamente $66, 924174$ km.

Exercício 2.b)

Para estipularmos o erro máximo cometido, precisaríamos ter como informação um M tal que $f''(x) \leq M$ para todo $a \leq x \leq b$. Como não temos esta informação, é computacionalmente custoso calcular o erro máximo nas condições dadas.

Exercício 2.c)

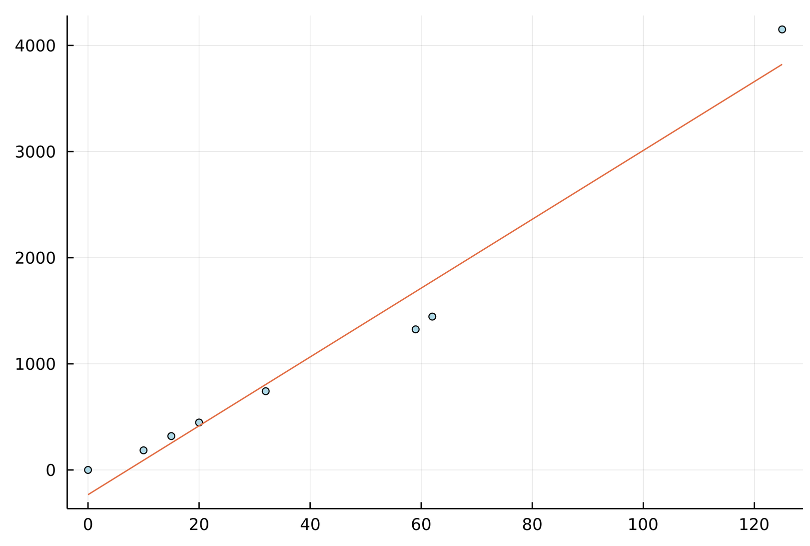
Para encontrarmos uma reta $p_1 = c_0 x + c_1$ que melhor descreve a distribuição dos pontos, podemos utilizar de nossa biblioteca o método da regressão. Depois disso, podemos utilizar o método do trapézio para calcular $\int_0^{125} p_1(x)\,dx$. Vamos utilizar então a função aproxima_onibus em conjunto com a regressão.

In [ ]:
```
function aproxima_onibus(x,y)
    coefs = regressao(x,y,1)
    return coefs
end
```

```
aproxima_onibus (generic function with 1 method)
```

In [ ]:
```
x = [0,10,15,20,32,59,62,125]
y = [0,185,319,447,742,1325,1445,4151]

coefs, _ = aproxima_onibus(x,y)
p = Polynomial(coefs)
scatter(x, y, c=:lightblue, ms=3, leg=false)
plot!(p,0,125)
```



Com isto, como já temos uma reta, podemos utilizar o método do trapézio de 0 até 125 diretamente e estimarmos a altura.

In [ ]:
```
altura = 0.0
x = [0,10,15,20,32,59,62,125]
y = [0,185,319,447,742,1325,1445,4151]
xn = [0,125]
coefs,_ = aproxima_onibus(x,y)
p = Polynomial(coefs)
altura += trapezio(p,xn[1],xn[2],1)
round(altura, digits=3)
```

```
224307.609
```

Com isso, temos que o ônibus espacial endeavour, no momento $t = 125$, de acordo com a aproximação pela reta, estava na altura de aproximadamente $224307, 609$ pés, equivalente a aproximadamente $68, 368$ km

Exercício 3.a)

Para este exercício, precisamos determinar uma aproximação para a área limitada ao primeiro quadrante da equação $x^2 + y^2 = 1$. Como utilizaremos o método do trapézio e estamos interessados apenas no primeiro quadrante, para simplificar as contas, utilizaremos a equação equivalente ao semi-círculo formado pelos 1º e 2º quadrantes, isto é, $y = \sqrt{1 - x^2}$. Após finalizar estes cálculos, com a aproximação obtida, devemos fazer uma aproximação de π utilizando o valor encontrado.

Para isto, adaptaremos o método do trapézio, utilizando a função trapezio_circulo, para fazer a aproximação utilizando a altura $h = 0.1$. Neste caso, como já temos a altura h, não precisaremos utilizar a interpolação para calcular os trapézios.

In [ ]:
```
function trapezio_circulo(f,a,b) #calcular a integral f(x) de a até b
    n = 10 # Como h=0.1, n = 10
    h = 0.1
    S=0.0
    for i=1:(n)  #calcula o "meio"
        x=a+i*h
        S+=2*f(x)
    end
    S=h/2*(S+f(a)+f(b)) #calcula "as pontas"
    return S
end
```

```
trapezio_circulo (generic function with 1 method)
```

Utilizando então os limites de integração $a = 0$ e $b = 1$, podemos estimar a área do primeiro quadrante.

In [ ]:
```
x = [0,1]
y = [1,0]
f(x) = sqrt(1-x^2)
area = trapezio_circulo(f,x[1],x[2])
```

```
0.7761295815620796
```

Como a parte da função do primeiro quadrante é referente a $\frac{\pi}{4}$, podemos multiplicar a área encontrada para dar uma estimativa de π.

In [ ]:
```
area * 4
```

```
3.1045183262483182
```

Exercício 3.b)

Para calcular o erro da aproximação, precisamos encontrar um M tal que $f''(x) <= M$ para todo $a \leq x \leq b$. Como $f(x) = \sqrt{1 - x^2}$, temos que $f''(x) = -\frac{1}{(1 - x^2)(\sqrt{1 - x^2})}$.

Neste caso, precisamos englobar todos os valores de a até b no cálculo. Entretanto, na extremidade $x = 1$, podemos encontrar uma indefinição.

$$f''(1) = -\frac{1}{(1 - 1^2)(\sqrt{1 - 1^2})} = -\frac{1}{0}$$

Com isto, já não podemos determinar o erro estimado para o caso dado, pois $f''(1)$ está indeterminado.

Exercício 4)

Para calcularmos uma integral na forma $\int_a^b \int_{h(y)}^{g(y)} f(x, y) dx dy$, podemos modificar o método Integral_Dupla para receber como parâmetro os limites de integração como função. Para isto, vamos criar uma nova função, Integral_Dupla_Funcao.

In [ ]:
```
function Integral(f,a,b)
    return trapezio(f,a,b,10000)
end
```

```
Integral (generic function with 1 method)
```

In [ ]:
```
# Integral dupla de h(x,y) de a até b no x e de c(y) até d(y) no y
function Integral_Dupla_Funcao(h,a,b,c,d)
    function g(y)
        f(x)=h(x,y)
        return Integral(f,a(y),b(y)) # Neste caso, utilizamos os limites de integração em função  de y
    end
    return Integral(g,c,d)
end
```

```
Integral_Dupla_Funcao (generic function with 1 method)
```

Como prova real, podemos utilizar o cálculo da função $\int_6^5 \int_{3y}^{2y} f(x, y) dx dy$.

In [ ]:
```
h(x,y)=x^2*y
a(y) = 2y
b(y) = 3y
Integral_Dupla_Funcao(h,a,b,5,6)
```

```
5891.266670138121
```

Com isto, temos que Integral_Dupla_Funcao é capaz de calcular a integral dupla de uma função na forma descrita. Podemos comparar então o resultado com alguma calculadora de integrais, como a fornecida pelo Symbolab.

$\int_6^5 \int_{3y}^{2y} x^2 y dx dy = \frac{88369}{15}$ (Decimal: $5891.26666\ldots$)