Relatório 5 de CCI – Cálculo Numérico em Matlab

Instituto Tecnológico de Aeronáutica

Professor Vitor Curtis

Pedro Luchiari de Carvalho – Turma 4

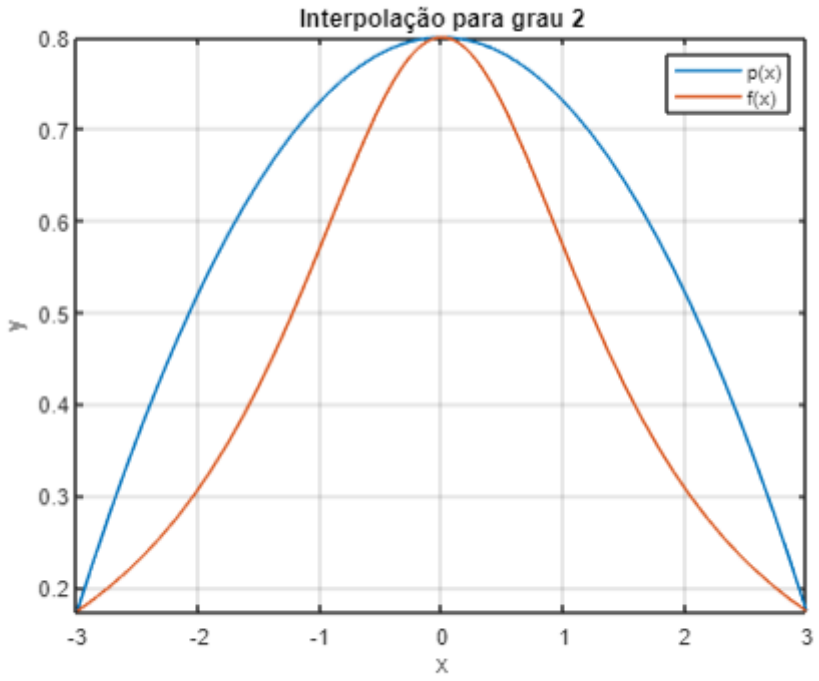
São José dos Campos, 25 de maio de 2022

Questão 1:

```
%cálculo dos nós de Chebyshev:
n=2;
a=-3;
b=3;
for i = 0 : n
    xc(i+1) = cos(pi*i/n);
end

for i = 0:n
    x(i+1) = (a+b)*0.5 - (b-a)*0.5*xc(i+1);
end

fprintf('nos de Chebyshev:')
for i=1:n+1
    fprintf('x%d = %.5f.\n',i-1, x(1,i));
end

%Agora vem a interpolação (usaremos a função polyfit). O output serão os coeficientes
% do nosso polinômio, que serão usados na subsequente plotagem:
y = arrayfun(@(x) f(x), x);
l = linspace(-3,3);
p = polyfit(x, y, n);

fplot(@(l) polyval(p, l), [-3, 3])
hold on
fplot(@(l) f(l), [-3, 3])
grid on
title('Interpolação para grau 2')
xlabel('x');
ylabel('y');
legend('p(x)', 'f(x)');

maxerror = 0;
for i = 1:100
    if polyval(p, l(i))-f(l(i)) > maxerror
       maxerror = polyval(p, l(i))-f(l(i));
    end
end

display (maxerror);
```

```
function d = f(x)
    d = 4/(5+2*x^2);
end
```

O Código acima fornece uma interpolação de grau 2, situação na qual usar os nós de Chebyshev é indiferente. Segue abaixo o output do programa:



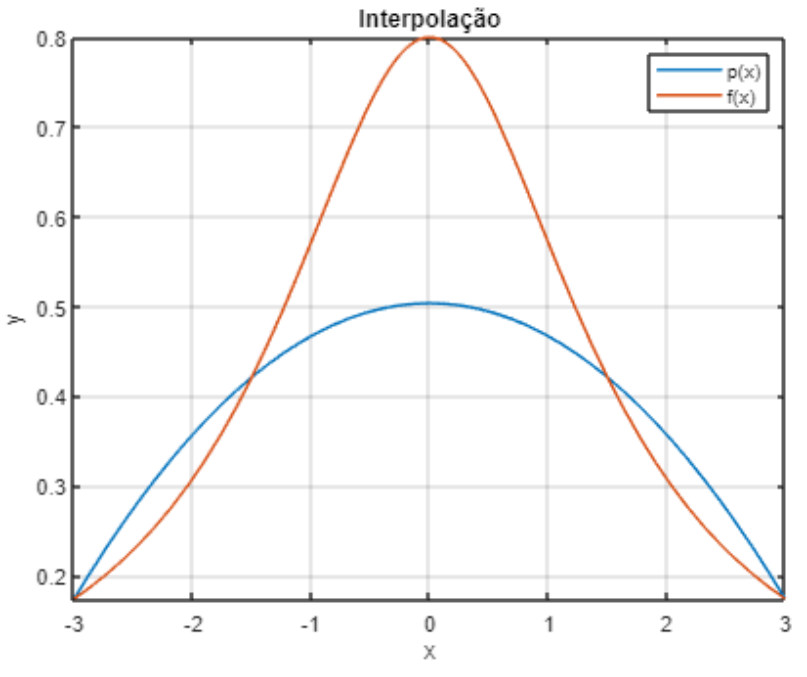

```
nos de Chebyshev:
x0 = -3.00000.
x1 = -0.00000.
x2 = 3.00000.
```

```
maxerror = 0.2278
```

Para interpolações de graus variando de 3 a 10, basta variar n de 3 a 10 no programa. Efetuando isso, chega-se nos seguintes resultados:

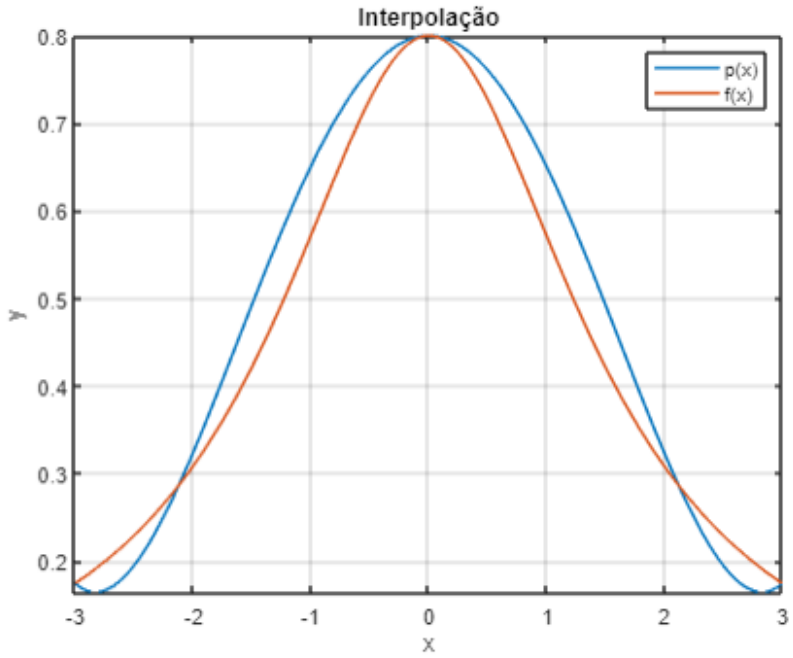
n=3:

```
nos de Chebyshev:
x0 = -3.00000.
x1 = -1.50000.
x2 = 1.50000.
x3 = 3.00000.
```



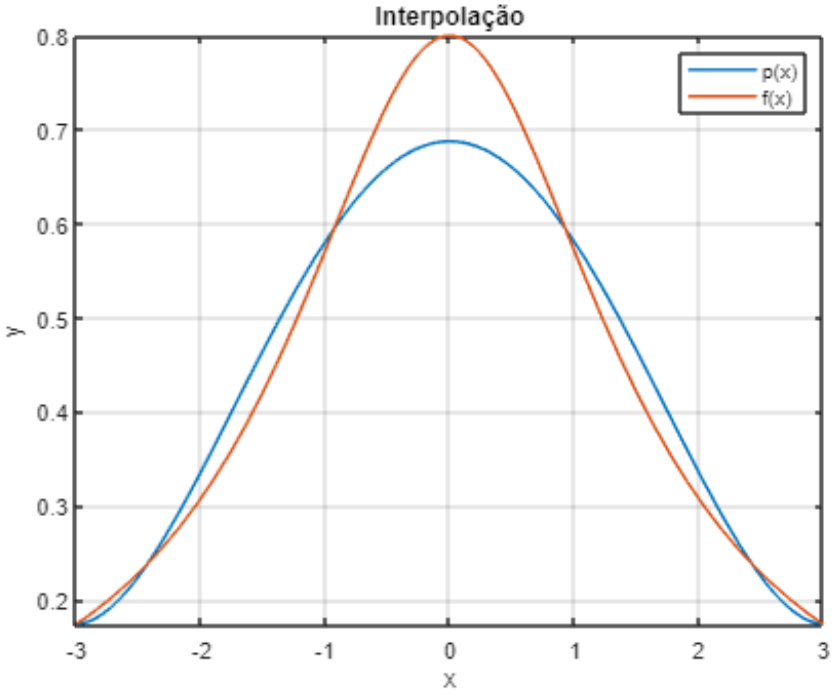

```
maxerror = 0.0538
```

n = 4:



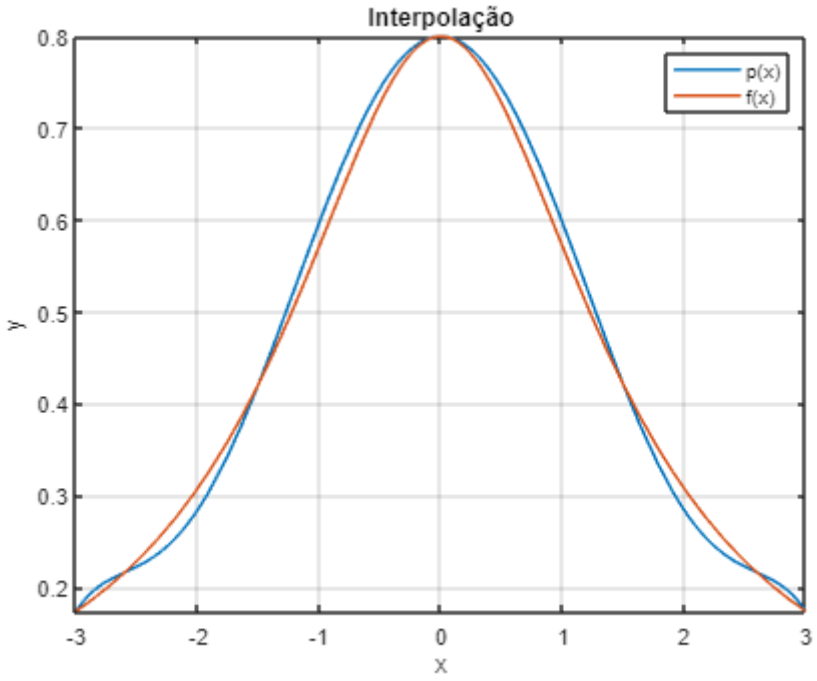

```
nos de Chebyshev:

x0 = -3.00000.
x1 = -2.12132.
x2 = -0.00000.
x3 = 2.12132.
x4 = 3.00000.

maxerror = 0.0840
```

N=5:



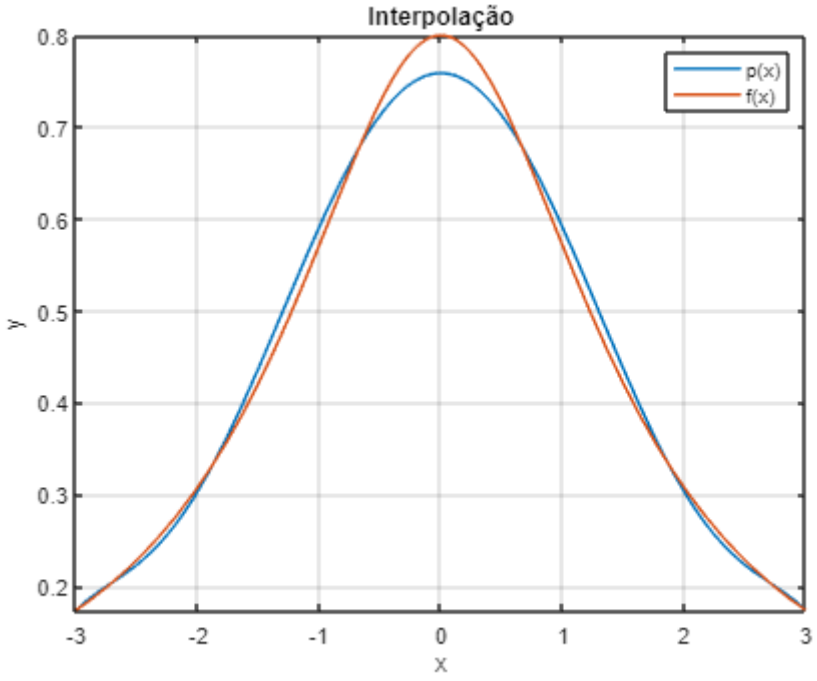

```
nos de Chebyshev:

x0 = -3.00000.
x1 = -2.42705.
x2 = -0.92705.
x3 = 0.92705.
x4 = 2.42705.
x5 = 3.00000.

maxerror = 0.0447
```

N=6:



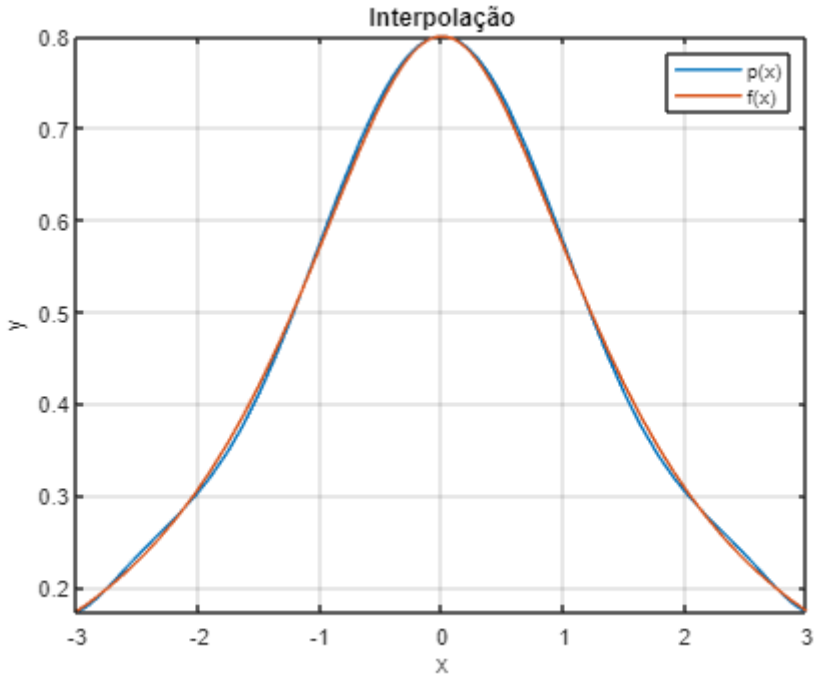

```
nos de Chebyshev:

x0 = -3.00000.
x1 = -2.59808.
x2 = -1.50000.
x3 = -0.00000.
x4 = 1.50000.
x5 = 2.59808.
x6 = 3.00000.

maxerror = 0.0272
```

N=7:



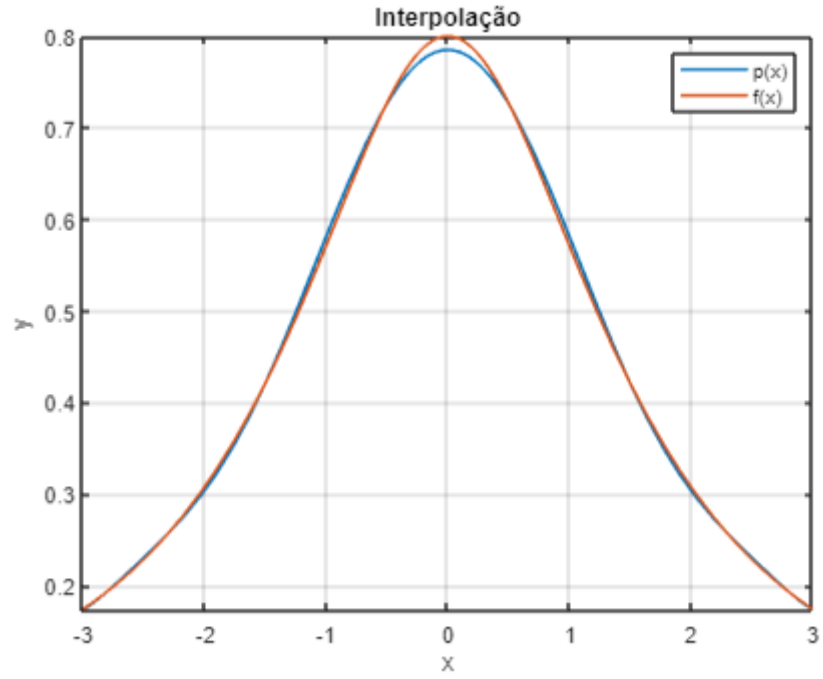

```
nos de Chebyshev:

x0 = -3.00000.
x1 = -2.70291.
x2 = -1.87047.
x3 = -0.66756.
x4 = 0.66756.
x5 = 1.87047.
x6 = 2.70291.
x7 = 3.00000.

maxerror = 0.0231
```

N=8:



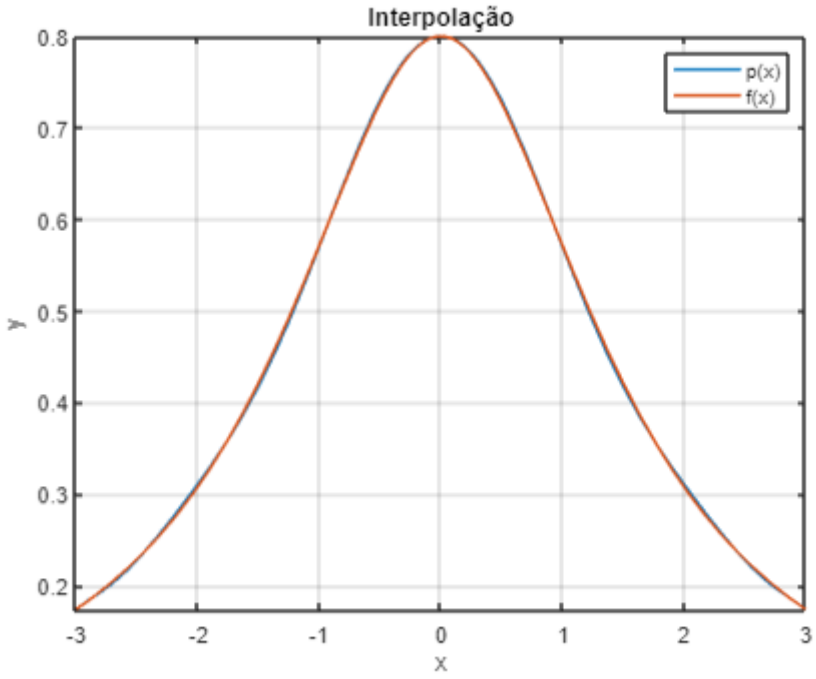

```
nos de Chebyshev:

x0 = -3.00000.
x1 = -2.77164.
x2 = -2.12132.
x3 = -1.14805.
x4 = -0.00000.
x5 = 1.14805.
x6 = 2.12132.
x7 = 2.77164.
x8 = 3.00000.

maxerror = 0.0085
```

N=9:




```
nos de Chebyshev:

x0 = -3.00000.
x1 = -2.81908.
x2 = -2.29813.
x3 = -1.50000.
x4 = -0.52094.
x5 = 0.52094.
x6 = 1.50000.
x7 = 2.29813.
x8 = 2.81908.
x9 = 3.00000.

maxerror = 0.0101
```

N=10:




```
nos de Chebyshev:

x0 = -3.00000.
x1 = -2.85317.
x2 = -2.42705.
x3 = -1.76336.
x4 = -0.92705.
x5 = -0.00000.
x6 = 0.92705.
x7 = 1.76336.
x8 = 2.42705.
x9 = 2.85317.
x10 = 3.00000.

maxerror = 0.0031
```

Questão 2:

```
x=[0 0.4 0.65 1.31 1.48 1.62 2.45];
y=[1 0.471698 0.252685 0.076848 0.0612265 0.051624 0.0232463];

valor1=SplineCubicaN(x,y,0.22)
valor2=SplineCubicaN(x,y,0.47)
valor3=SplineCubicaN(x,y,1.09)
valor4=SplineCubicaN(x,y,1.36)
valor5=SplineCubicaN(x,y,1.92)

y1 = arrayfun(@(x) f(x), x);
display (y1);

error1=abs(valor1-y1(1))/y1(1)
error2=abs(valor2-y1(2))/y1(2)
error3=abs(valor3-y1(3))/y1(3)
error4=abs(valor4-y1(4))/y1(4)
error5=abs(valor5-y1(5))/y1(5)
```

```
function[val]=SplineCubicaN(x,y,xx);
    n=length(x)-1; m = n - 1; h = zeros(1,m+1);
    for i = 0:m
        h(i+1) = x(i+2) - x(i+1);
    end
    A=zeros(n-1,n-1); A(1,1)=1; A(length(x),length(x))=1;
    for i=2:n
        for j=2:n
            if i==j
                A(i,j)=2*(h(i-1)+h(i)); A(i,j-1)=h(i-1); A(i,j+1)=h(i);
            end
        end
    end
    b=zeros(length(x),1);
    for i=2:n
        b(i,1)=6*(((y(i+1)-y(i))/(h(i)))-((y(i)-y(i-1))/(h(i-1))));
    end
    g=A\b;
    %coeficientes ak, bk, ck e dk
    for k=1:length(x)-1
        ak(k)=(g(k+1,1)-g(k))/(6*h(k));
        bk(k)=g(k+1)/2;
        ck(k)=((y(k+1)-y(k))/h(k))+(2*h(k)*g(k+1)+g(k)*h(k))/6;
        dk(k)=y(k+1);
    end
    i=1;
    while xx>x(i+1)
        i=i+1;
    end
    val=ak(i)*(xx-x(i+1)).^3+bk(i)*(xx-x(i+1)).^2+ck(i)*(xx-x(i+1))+dk(i);
end

function d=f(x)
    d=1/(1+7*x^2);
end
```

Output:

```
valor1 = 0.6920
valor2 = 0.3993
valor3 = 0.0995
valor4 = 0.0721
valor5 = 0.0375
y1 = 1×7
    1.0000    0.4717    0.2527    0.0768    0.0612    0.0516    0.0232


error1 = 0.3080
error2 = 0.1534
error3 = 0.6063
error4 = 0.0612
error5 = 0.3877
```

Discussão dos Resultados:

No primeiro programa, percebe-se claramente a vantagem do uso dos nós de Shebychev comparativamente a abscissas equidistantes, pois, quanto maior o grau do polinômio interpolador, maior é o ajuste de curvas e, geralmente, menor é o erro absoluto máximo no intervalo considerado.

Na questão 2, verificamos a aplicabilidade de splines cúbicas interpolantes como método de interpolação, o qual não necessita da função verdadeira dos pontos interpolados; no entanto, notamos os consideráveis erros relativos.