
INTERPOLAÇÃO POLINOMIAL

Considere um experimento ou fenômeno físico que resulte em um conjunto de dados tabelados. Essas informações podem ser tomadas como coordenadas de pontos que descrevem uma certa função f(x). Mas não temos a definição analítica dessa função, temos apenas alguns pontos em um intervalo. O que devemos fazer para estimar o comportamento do fenômeno, ou o valor dessa função, em pontos que não estão tabelados? Há duas alternativas:

1. Interpolação: aplicada quando os dados estão corretos e resulta em uma expressão analítica para a função f(x) que passa exatamente em cima dos pontos fornecidos.
2. Ajuste de curvas: aplicada quando os dados possuem ruído ou erros na medição e resulta em uma curva suave que melhor se ajusta aos dados no sentido dos mínimos quadrados (a soma do quadrado das distâncias entre o dado fornecido e a curva ajustada deve ser tão próxima de zero quanto possível).

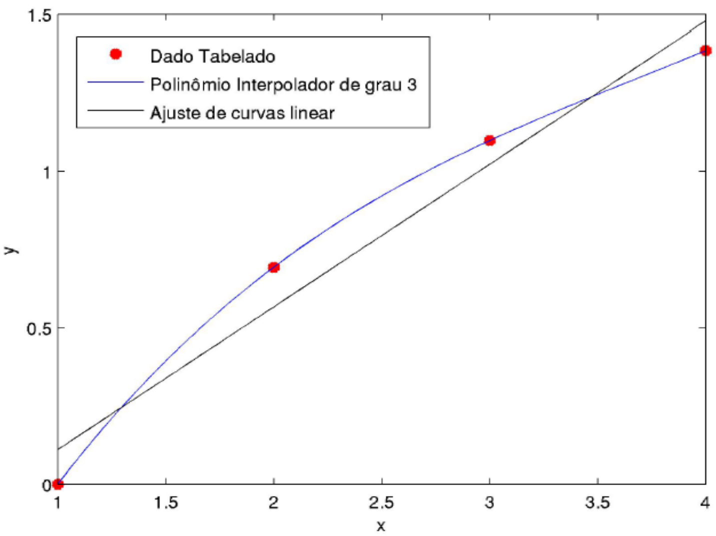


Figura 1: Interpolação versus Ajuste de Curvas.

As funções do Matlab/Octave mais utilizadas na interpolação e ajuste de curvas são:

- **interp1**
- **polyfit**
- **spline**

Interp1

Dados os vetores **x** e **y** (pontos tabelados), '**interp1**' encontra o valor de xx e armazena em yy por meio de interpolação linear (default). A sintaxe é:
yy = interp1(x,y,xx)

em que xx pode ser um único valor ou um vetor de n posições.

Obs: Em muitos casos a interpolação linear é suficiente. Por essa razão é o método usado p
definição no Matlab/Octave. Para mais detalhes dessa função acesse o manual (help interp1).

***Exemplo 1**: Seja a tabela de pontos*

x	*-1*	*0*	*2*
y	*4*	*1*	*-1*

Qual o valor da função para x = 1?

- *Manualmente (ver notas de aula)*
  Por interpolação linear usando as formas de Lagrange ou Newton encontramos o polinôm.
  $P_1(x) = 1 - x$. *Logo,* $f(1) \sim P_1(1) = 1 - 1 = 0$.

- *No Matlab*
  ```
  x = [ -1   0   2];
  y = [  4   1  -1];
  f = interp1(x,y,1)
  ```

- *Comandos para visualização:*
  ```
  clear                      % limpa as variáveis
  x = [ -1   0   2];         % coordenada x do dado
  y = [  4   1  -1];         % coordenada y do dado
  xx = linspace(-1,2);       % vetor auxiliar de 100 posições xx[1],xx[2],...,xx[100]
  yy = interp1(x,y,xx);      % armazena em yy[i] o resultado da interpolação em xx[i]
  figure                     % cria janela gráfica
  plot(x,y,'r*');            % plota o dado tabelado com pontos em vermelho
  hold on                    % comando que fixa o eixo coordenado
  plot(xx ,yy);              % plota o resultado da interpolação linear
  ```

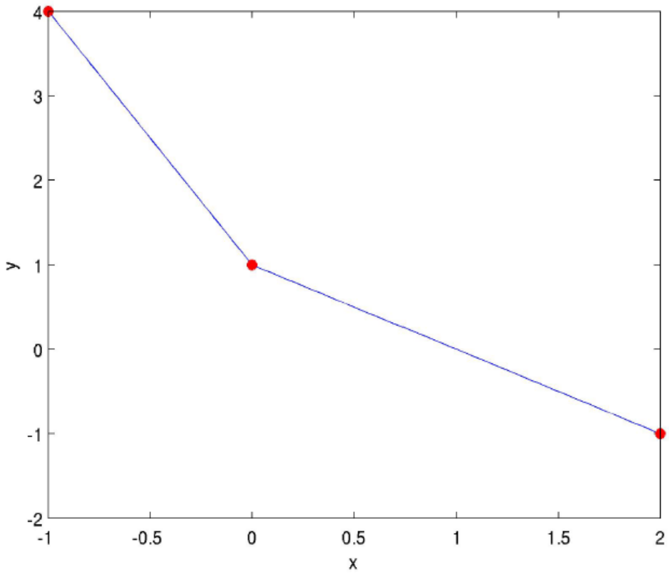


Figura 2: Interpolação linear obtida por 'interp1'.

Polyfit (Mínimos Quadrados)

A função '**polyfit**' encontra os coeficientes do polinômio de grau n ajustado aos dados no sentido dos mínimos quadrados. Se escolhermos n=1 para o grau do polinômio, a melhor reta que ajusta os dados será encontrada (esse caso particular é amplamente conhecido como **Regressão Linear**). Se n = 2, um polinômio quadrático será encontrado, e assim por diante. Sintaxe:

p = polyfit(x,y,n)

em que x e y são as coordenadas do dado e n o grau do polinômio que será ajustado.

A curva ajustada aos pontos por mínimos quadrados é escolhida de maneira que a soma dos quadrados dos erros nos pontos seja minimizada. Embora o ajuste de curvas por mínimos quadrados possa ser feito utilizando qualquer conjunto de funções base, é comum e direto o uso de polinômios. Dessa forma, a função 'polyfit' do Matlab/Octave também utiliza polinômios como seu conjunto de funções base e obviamente retorna como solução um polinômio de grau especificado. Seja n o grau do polinômio desejado, o Matlab/Octave fornece os n+1 coeficientes do polinômio de grau n ajustado aos dados na seguinte ordem: a_n, a_{n-1}, ..., a_2, a_1, a_0. Lembrando que um polinômio de grau n pode ser dado por:

$$Pn(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_2 x^2 + a_1 x + a_0$$

Obs: Quando o conjunto de n+1 pontos tabelados gera uma matriz de Vandermonde associada mal condicionada, não é possível obter um polinômio de grau n por 'polyfit' e é exibida uma mensagem de erro na tela. É preciso reduzir o grau do polinômio que se deseja ajustar. Polinômios de ordem elevada, em geral, não são boas opções.

Exemplo 2*: Seja a tabela de pontos do Exemplo1. Qual a forma geral do polinômio interpolador de grau 2 que fita os dados no sentido dos mínimos quadrados?*

- *No Matlab*
```
x = [ -1   0   2];       % coordenada x do dado
y = [  4   1  -1];       % coordenada y do dado
n = 2;                   % grau do polinômio
p = polyfit(x,y,n);      % armazena os coeficientes an,...,a0 do polinômio de grau n
```
  Note que p será um vetor de 3 (n+1) posições em que p[1] será o coeficiente a_2, p[2] o coeficente a_1 e p[3] o coeficiente a_0 do polinômio $P_2(x) = a_2 x^2 + a_1 x + a_0$.

- *Comandos para visualização:*
```
clear                    % limpa as variáveis
x = [ -1   0   2];       % coordenada x do dado
y = [  4   1  -1];       % coordenada y do dado
n = 2;                   % grau do polinômio interpolador
p = polyfit(x,y,n);      % vetor que armazena os coeficientes an,...,a0 do polinômio de grau n
xx = linspace(-1,2);     % vetor auxiliar de 100 posições xx[1],xx[2],...,xx[100]
yy=polyval(p,xx);        % gera imagem do polinômio em cada posição do xx
figure                   % cria janela gráfica
plot(x,y,'r*');          % plota o dado tabelado com pontos em vermelho
hold on                  % comando que fixa o eixo coordenado
plot(xx,yy);             % plota o resultado do polinômio de grau n obtido por polyfit
```

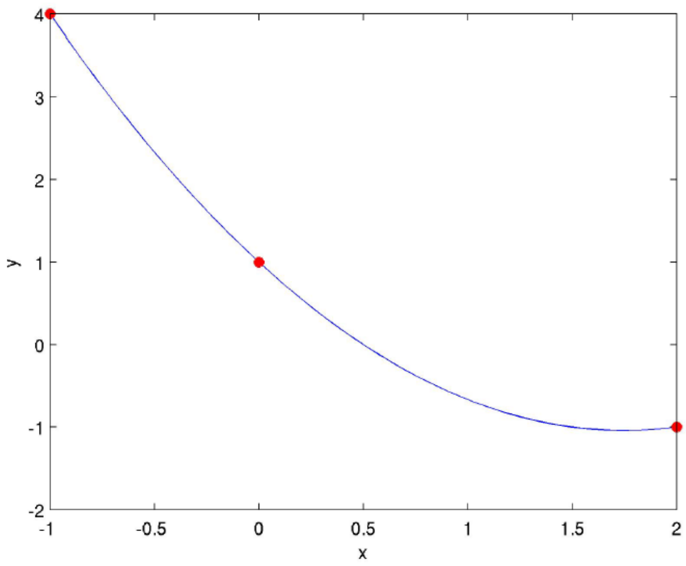


Figura 3: Polinômio de grau 2 no intervalo [-1,2] obtido por 'polyfit' e 'polyval'.

EXERCÍCIO 1: Encontre os coeficientes da reta que melhor ajusta os dados do "Exemplo 1" sentido dos mínimos quadrados. Plote esse ajuste linear.

Spline

Dados os vetores **x** e **y** (pontos tabelados), '**spline**' retorna informações do polinômio por par resultado da spline cúbica interpolante sobre os dados. Para acessar essas informações é preciso us o comando 'ppval' que avalia um polinômio por partes. A sintaxe é:

pp = spline(x,y)
v = ppval(pp,xx)

em que xx é um vetor de n posições.

Exemplo 3: *Seja a tabela de pontos do Exemplo1, plote a spline cúbica interpolante que fita dados.*

- *Comandos:*

```
clear                  % limpa as variáveis
x = [ -1   0   2];     % coordenada x do dado
y = [  4   1  -1];     % coordenada y do dado
s = spline(x,y);       % spline cúbica interpolante para x e y
xx = linspace(-1,2);   % vetor auxiliar de 100 posições
yy=ppval(s,xx);        % gera imagem do polinômio em cada posição do xx
figure                 % cria janela gráfica
plot(x,y,'r*');        % plota o dado tabelado com pontos em vermelho
hold on                % comando que fixa o eixo coordenado
plot(xx,yy);           % plota o resultado do polinômio de grau n obtido por polyfit
```

EXERCÍCIO 2: Considere os dados tabelados abaixo:

0	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**1**
-0.477	**1.978**	**3.28**	**6.16**	**7.08**	**7.34**	**7.66**	**9.56**	**9.48**	**9.30**	**11.2**

Plote em gráficos separados os resultados para:
a) Interpolação Linear
b) Regressão Linear
c) Ajuste por um polinômio de grau 2
d) Ajuste por um polinômio de grau 10
e) Spline Cúbica

Plote em um mesmo eixo coordenado todos os resultados obtidos nos itens a – e.

O que se pode concluir das estratégias de interpolação utilizadas?

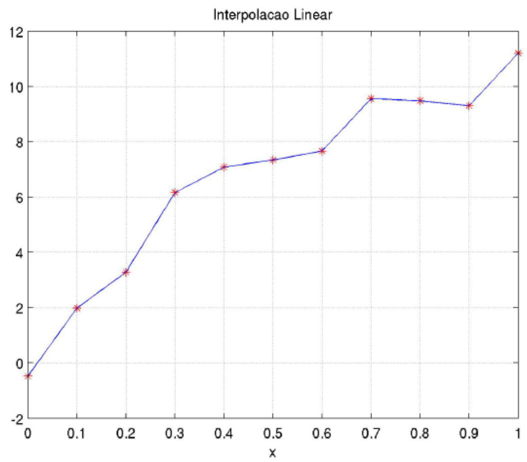
Resolução do Exercício 2

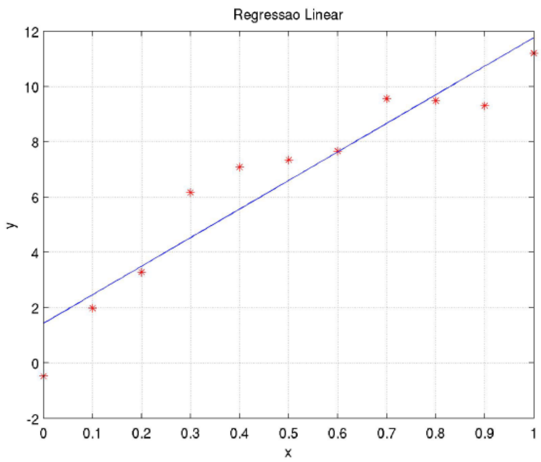
Comandos Matlab/Octave:

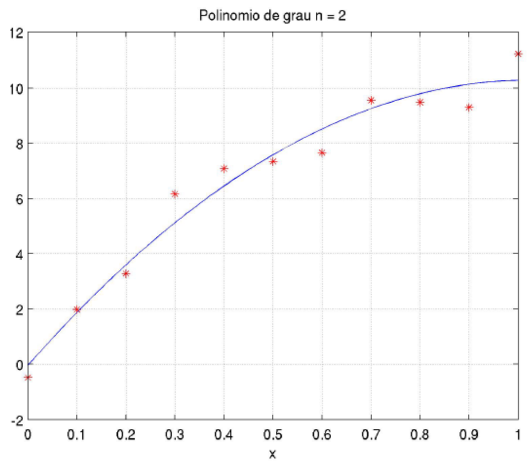
```
clear all
close all
clc

x=0:0.1:1;
y=[-0.4770 1.9780 3.2800 6.1600 7.0800 7.3400 7.6600 9.5600 9.4800 9.3000 11.2000];

%Interpolação Linear
xx=linspace(0,1);
a=interp1(x,y,xx);
figure;
plot(x,y,'r*',xx,a);
grid on
xlabel('x')
ylabel('y')
title('Interpolacao Linear')

%Regressao Linear
p1=polyfit(x,y,1);
b=polyval(p1,xx);
figure;
plot(x,y,'r*',xx,b);
grid on
xlabel('x')
ylabel('y')
title('Regressao Linear')

%Polinômio de grau n = 2
p2=polyfit(x,y,2);
c=polyval(p2,xx);
figure;
plot(x,y,'r*',xx,c);
grid on
xlabel('x')
ylabel('y')
title('Polinomio de grau n = 2')

%Polinômio de grau n = 10
p10=polyfit(x,y,10);
d=polyval(p10,xx);
figure;
plot(x,y,'r*',xx,d);
grid on
xlabel('x')
ylabel('y')
title('Polinomio de grau n = 10')

%Spline Cúbica
```

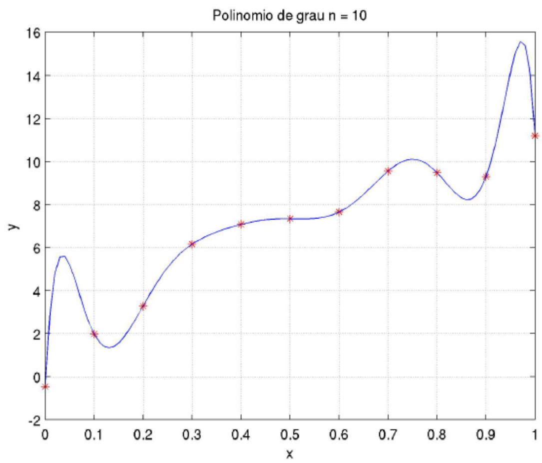
```
s=spline(x,y);
e=ppval(s,xx);
figure;
plot(x,y,'r*',xx,e);
grid on
xlabel('x')
ylabel('y')
title('Spline Cubica')

figure
plot(x,y,'r*')
hold on
plot(xx,a,'b');
plot(xx,b,'y');
plot(xx,c,'g');
plot(xx,d,'k');
plot(xx,e,'m');
grid on
xlabel('x');
ylabel('y');
legend('Dados Tabelados', 'Interpolacao Linear', 'Regressao Linear','Polinomio grau n=2', 'Polinomio grau n = 10', 'Spline Cubica')
```

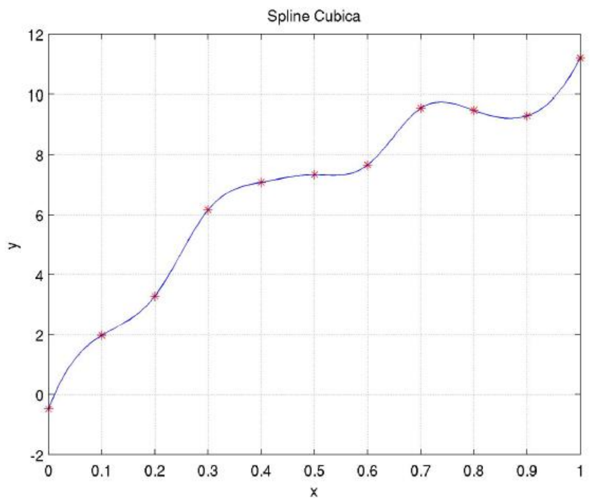
Gráficos obtidos

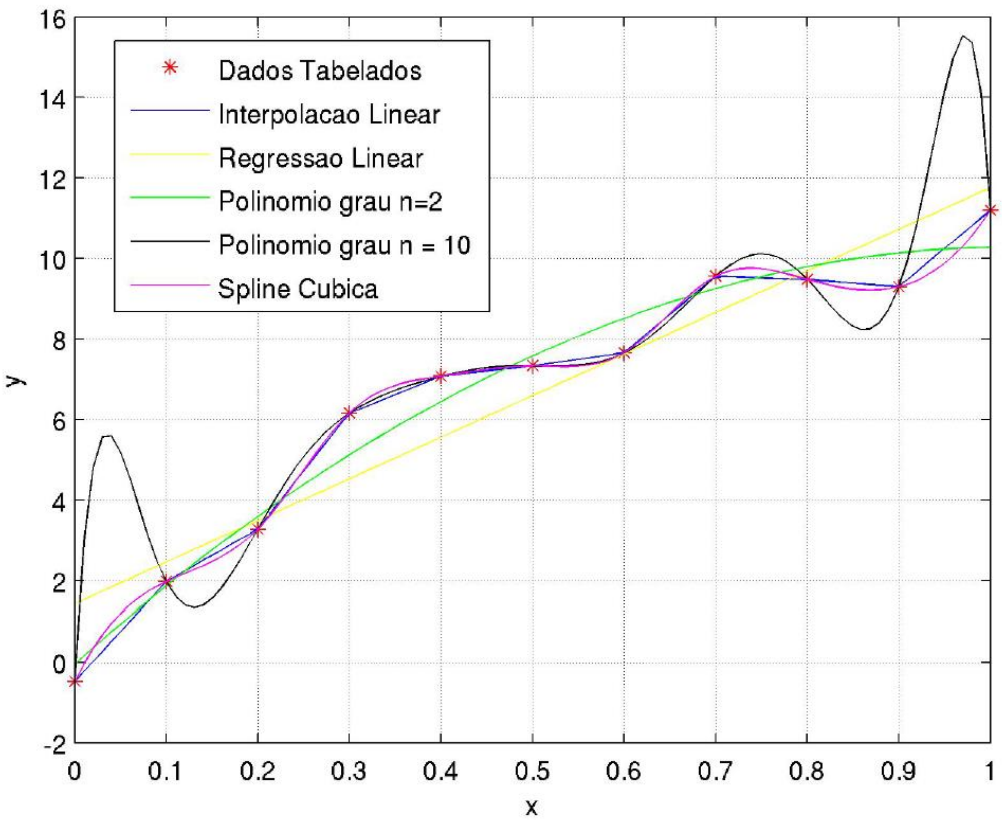









Spline Cubica
12
10
8
6
4
2
0
-2
y
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
x


Dados Tabelados
Interpolacao Linear
Regressao Linear
Polinomio grau n=2
Polinomio grau n = 10
Spline Cubica
16
14
12
10
8
6
4
2
0
-2
y
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
x