

Interpolação Polinomial

Prof. Americo Cunha

Universidade do Estado do Rio de Janeiro – UERJ

americo.cunha@uerj.br

www.americocunha.org



@AmericoCunhaJr



@AmericoCunhaJr



@AmericoCunhaJr



@AmericoCunhaJr



Imagine que você tenha que lidar com uma **função complicada**:



Imagine que você tenha que lidar com uma **função complicada**:

$$f(x) = \frac{\ln x e^{\sqrt{x}}}{\ln(1 + e^x) + x^3}$$

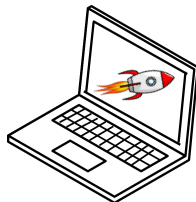
fórmula complexa



Imagine que você tenha que lidar com uma **função complicada**:

$$f(x) = \frac{\ln x e^{\sqrt{x}}}{\ln(1 + e^x) + x^3}$$

fórmula complexa

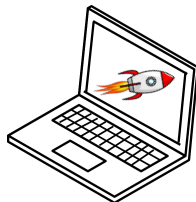


**simulador computacional
(sem fórmula)**

Imagine que você tenha que lidar com uma **função complicada**:

$$f(x) = \frac{\ln x e^{\sqrt{x}}}{\ln(1 + e^x) + x^3}$$

fórmula complexa



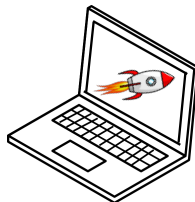
**simulador computacional
(sem fórmula)**

**Como manipular (derivar, integrar etc)
uma função desse tipo na prática?**

Imagine que você tenha que lidar com uma **função complicada**:

$$f(x) = \frac{\ln x e^{\sqrt{x}}}{\ln(1 + e^x) + x^3}$$

fórmula complexa



**simulador computacional
(sem fórmula)**

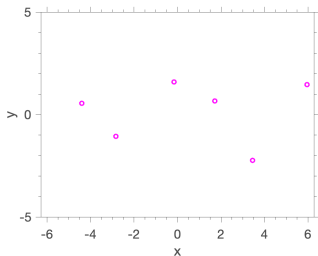
**Como manipular (derivar, integrar etc)
uma função desse tipo na prática?**

**Através de uma função simplista que aproxima o
comportamento de $f(x)$, construída a partir de alguns
pontos onde o valor da função é conhecido!**

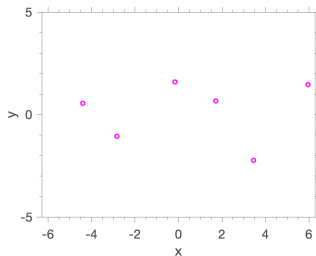


Interpolação × Regressão

Interpolação

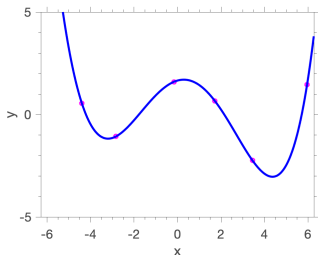


Regressão



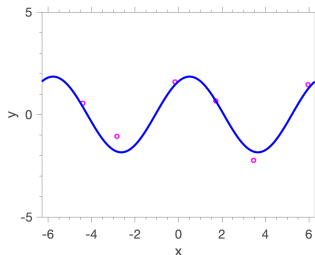
Interpolação × Regressão

Interpolação



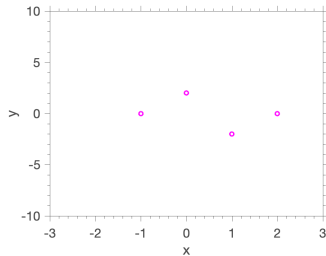
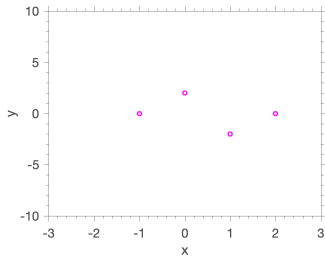
A curva interpolante passa
por todos os pontos.

Regressão

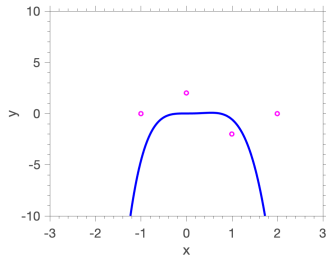
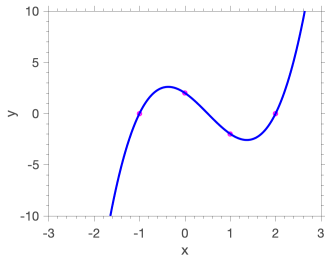


A curva regressora passa
“próxima” de todos os pontos.

Dados bem comportados ...



Dados bem comportados ...



**... podem ser bem aproximado (em geral)
por um processo de interpolação!**

O problema de interpolação

Dado um conjunto com $n + 1$ *pontos* distintos ($x_i \neq x_j$)

$$(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n),$$

encontre uma *função interpolante* $p(x)$ tal

$$p(x_j) = y_j, \quad j = 0, 1, \dots, n$$

i.e., a curva $y = p(x)$ passa pelos $n + 1$ pontos dados.



Processo de interpolação

1. Escolha uma forma para a **função interpolante**

$$p(x) = c_n \phi_n(x) + \cdots + c_1 \phi_1(x) + c_0 \phi_0(x)$$

- ϕ_j - funções linearmente independentes (predefinidas)
- c_j - coeficientes (a serem determinados)

2. Encontre os **coeficientes c_j** (resolvendo um sistema linear) de modo que **$y = p(x)$** seja uma **curva que contenha todos os pontos da amostra**.



Função interpolante



Função interpolante

$$p(x) = c_n \phi_n(x) + \cdots + c_1 \phi_1(x) + c_0 \phi_0(x)$$



Função interpolante

$$p(x) = c_n \phi_n(x) + \cdots + c_1 \phi_1(x) + c_0 \phi_0(x)$$

$$p(x_0) = c_n \phi_n(x_0) + \cdots + c_1 \phi_1(x_0) + c_0 \phi_0(x_0) = y_0$$

$$p(x_1) = c_n \phi_n(x_1) + \cdots + c_1 \phi_1(x_1) + c_0 \phi_0(x_1) = y_1$$

$$\vdots$$
$$\vdots$$

$$p(x_n) = c_n \phi_n(x_n) + \cdots + c_1 \phi_1(x_n) + c_0 \phi_0(x_n) = y_n$$



Função interpolante

$$\underbrace{\begin{bmatrix} \phi_n(x_0) & \cdots & \phi_1(x_0) & \phi_0(x_0) \\ \phi_n(x_1) & \cdots & \phi_1(x_1) & \phi_0(x_1) \\ \vdots & \ddots & \vdots & \vdots \\ \phi_n(x_{n-1}) & \cdots & \phi_1(x_{n-1}) & \phi_0(x_{n-1}) \\ \phi_n(x_n) & \cdots & \phi_1(x_n) & \phi_0(x_n) \end{bmatrix}}_A \underbrace{\begin{bmatrix} c_n \\ c_{n-1} \\ \vdots \\ c_1 \\ c_0 \end{bmatrix}}_x = \underbrace{\begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_1 \\ y_0 \end{bmatrix}}_b$$



$$A\mathbf{x} = \mathbf{b}$$



Função interpolante

$$\underbrace{\begin{bmatrix} \phi_n(x_0) & \cdots & \phi_1(x_0) & \phi_0(x_0) \\ \phi_n(x_1) & \cdots & \phi_1(x_1) & \phi_0(x_1) \\ \vdots & \ddots & \vdots & \vdots \\ \phi_n(x_{n-1}) & \cdots & \phi_1(x_{n-1}) & \phi_0(x_{n-1}) \\ \phi_n(x_n) & \cdots & \phi_1(x_n) & \phi_0(x_n) \end{bmatrix}}_A \underbrace{\begin{bmatrix} c_n \\ c_{n-1} \\ \vdots \\ c_1 \\ c_0 \end{bmatrix}}_x = \underbrace{\begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_1 \\ y_0 \end{bmatrix}}_b$$



$$A\mathbf{x} = \mathbf{b}$$

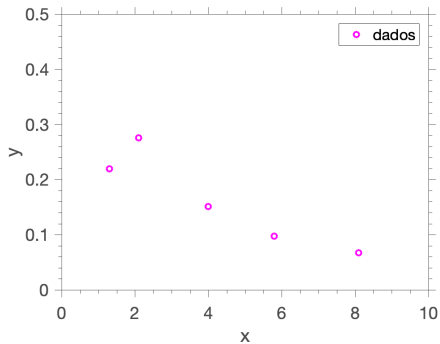
Esse sistema tem uma única solução!



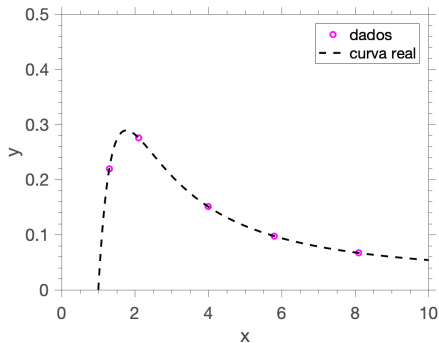
Experimento computacional 1

```
1  clc; clear; close all
2
3  f      = @(x) log(x).*exp(sqrt(x))./(log(1+exp(x))+x.^3);
4  xdat   = [1.3 2.1 4.0 5.8 8.1]; ydat = f(xdat);
5  xgrid  = 0.1:0.01:10; ygrid = f(xgrid);
6
7  A      = [xdat.^4; xdat.^3; xdat.^2; xdat; ones(1,5)]';
8  xsol   = A\ydat';
9  p1     = xsol(1)*xgrid.^4 + xsol(2)*xgrid.^3 + ...
10         xsol(3)*xgrid.^2 + xsol(4)*xgrid + xsol(5);
11  A      = [exp(-xdat.^2); exp(-xdat); xdat.^2; xdat; ones(1,5)]';
12  xsol   = A\ydat';
13  p2     = xsol(1)*exp(-xgrid.^2) + xsol(2)*exp(-xgrid) + ...
14         xsol(3)*xgrid.^2 + xsol(4)*xgrid + xsol(5);
15
16  plot(xdat,ydat,'om','LineWidth',2);
17  hold on
18  plot(xgrid,ygrid,'--k','LineWidth',2);
19  plot(xgrid,p1,'-r','LineWidth',2);
20  plot(xgrid,p2,'-b','LineWidth',2);
21  hold off
22  xlabel('x'); ylabel('y'); set(gca,'FontSize',18);
23  legend('dados','f(x)','p1(x)','p2(x)'); xlim([0 10]); ylim([0 0.5]);
```

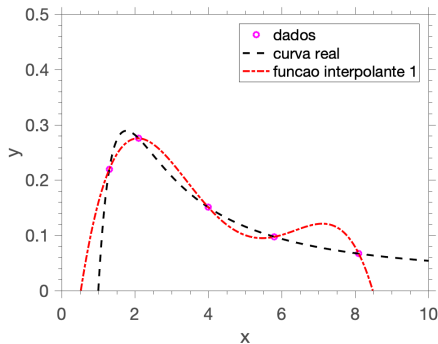




$$f(x) = \frac{\ln x e^{\sqrt{x}}}{\ln(1 + e^x) + x^3}$$

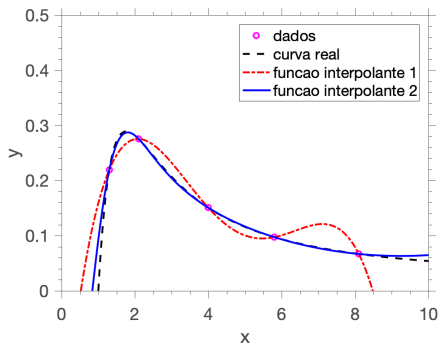


$$f(x) = \frac{\ln x e^{\sqrt{x}}}{\ln(1 + e^x) + x^3}$$



$$p_1(x) = c_4 x^4 + c_3 x^3 + c_2 x^2 + c_1 x + c_0$$

$$f(x) = \frac{\ln x e^{\sqrt{x}}}{\ln(1 + e^x) + x^3}$$



$$p_1(x) = c_4 x^4 + c_3 x^3 + c_2 x^2 + c_1 x + c_0$$

$$p_2(x) = c_4 e^{-x^2} + c_3 e^{-x} + c_2 x^2 + c_1 x + c_0$$



Interpolação polinomial

A **função interpolante** é assumida como sendo

$$p(x) = c_n x^n + \cdots + c_1 x + c_0$$

i.e., as **funções base** são da forma $\phi_j(x) = x^j$ (base monomial).



Interpolação polinomial

A **função interpolante** é assumida como sendo

$$p(x) = c_n x^n + \cdots + c_1 x + c_0$$

i.e., as **funções base** são da forma $\phi_j(x) = x^j$ (base monomial).

$$\begin{bmatrix} x_0^n & \cdots & x_0 & 1 \\ x_1^n & \cdots & x_1 & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_n^n & \cdots & x_n & 1 \end{bmatrix} \begin{bmatrix} c_n \\ c_{n-1} \\ \vdots \\ c_0 \end{bmatrix} = \begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_0 \end{bmatrix}$$

Sistema linear de Vandermonde (solução única)

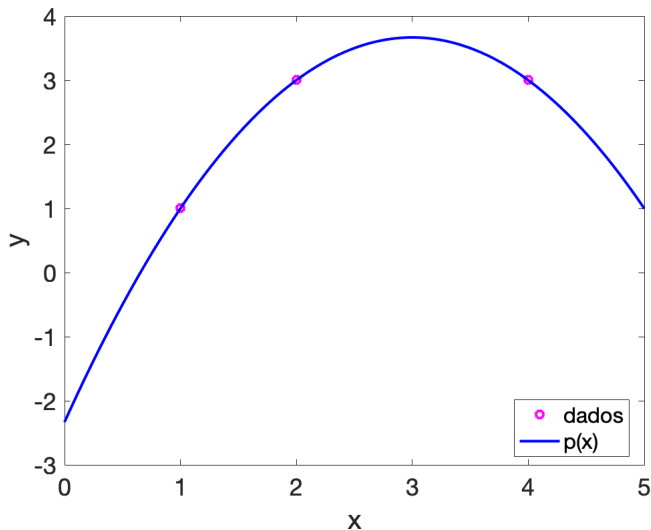


Experimento computacional 2

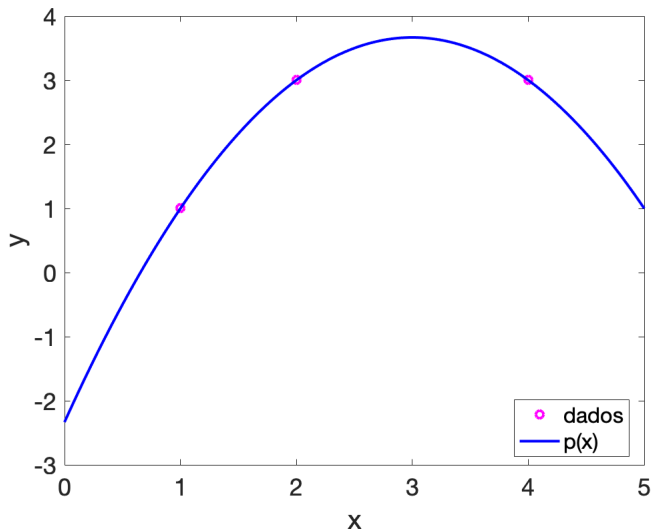
```
1  clc; clear; close all
2
3  xdat = [1;2;4]; ydat = [1;3;3];
4
5  A      = vander(xdat)
6  xsol = A\ydat
7
8  p = @(x) polyval(xsol, x);
9  xgrid = 0:0.01:5; ygrid = p(xgrid);
10
11 plot(xdat,ydat,'om',xgrid,ygrid,'-b','LineWidth',2);
12 xlabel('x'); ylabel('y'); set(gca,'FontSize',18);
13 legend('dados','p(x)','Location','south');
```



Experimento computacional 2



Experimento computacional 2



$$p(x) = -\frac{2}{3}x^2 + 4x - \frac{7}{3}$$



Fundamentação teórica

Teorema (Existência e unicidade do polinômio interpolante)

Para qualquer conjunto real com $n + 1$ pontos (abscissas distintas)

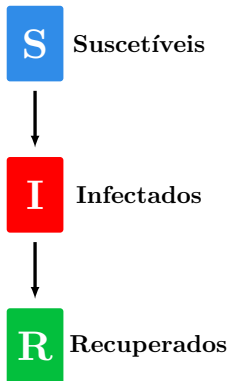
$$(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n),$$

existe um único polinômio $p(x)$ de grau $\leq n$ tal que

$$p(x_j) = y_j, \quad j = 0, 1, \dots, n.$$



Simulador de uma epidemia

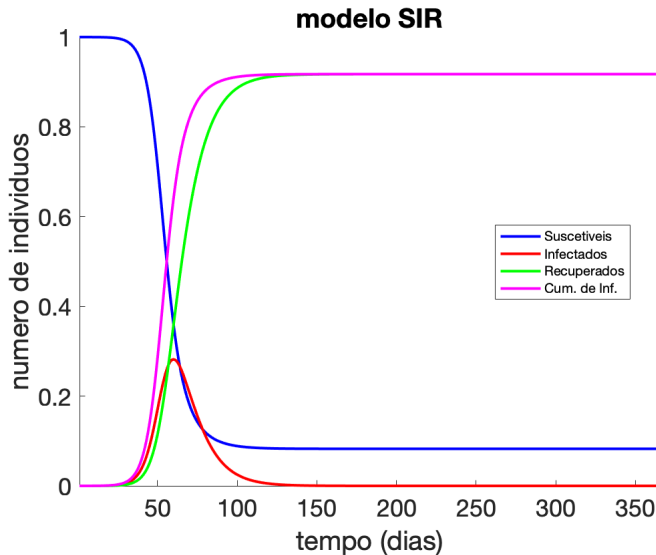


$$\begin{aligned}\frac{dS}{dt} &= -\beta S \frac{I}{N} \\ \frac{dI}{dt} &= \beta S \frac{I}{N} - \gamma I \\ \frac{dR}{dt} &= \gamma I \\ &+ \text{condições iniciais}\end{aligned}$$

*Simulador disponível em <http://www.EpidemicCode.org>

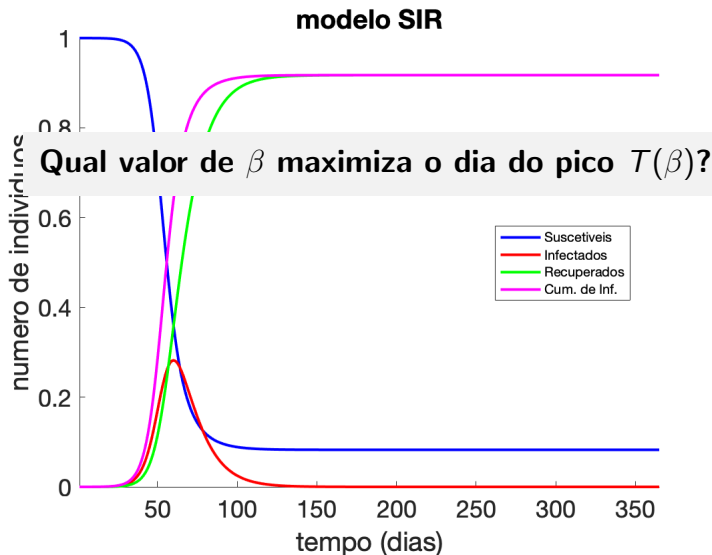


Simulador de uma epidemia



* Simulação com $N = 1$, $\beta = 1/3$, $\gamma = 1/10$, $R(0) = 0$, $I(0) = 10^{-6}$, $S(0) = N - I(0) - R(0)$

Simulador de uma epidemia



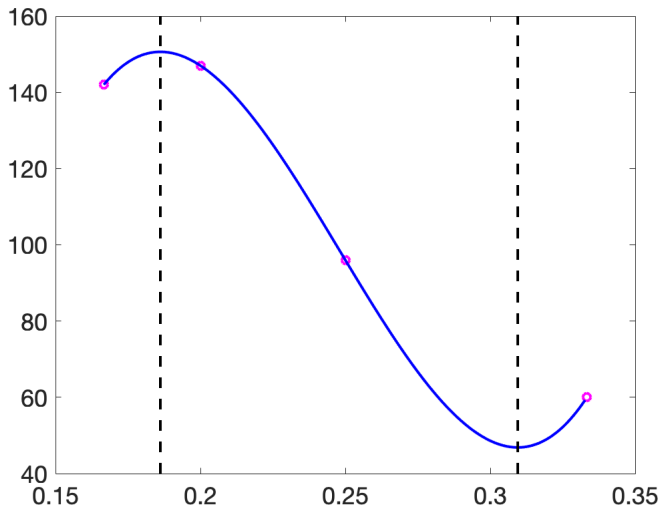
* Simulação com $N = 1$, $\beta = 1/3$, $\gamma = 1/10$, $R(0) = 0$, $I(0) = 10^{-6}$, $S(0) = N - I(0) - R(0)$

Interpolando dados de um simulador

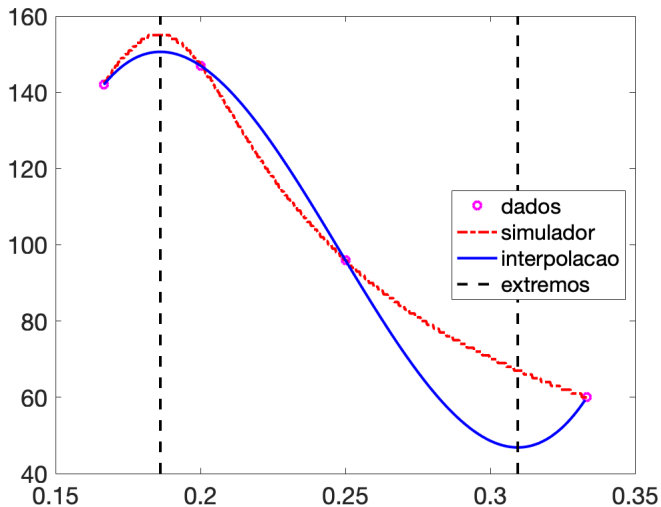
```
1  clc; clear; close all
2
3  beta    = [1/3 1/4 1/5 1/6];
4  T_beta = [60 96 147 142];
5
6  coef_T   = polyfit(beta,T_beta,3);
7  coef_dT  = polyder(coef_T);
8  beta_ast = roots(coef_dT)
9
10 p = @(x) polyval(coef_T,x);
11 x1 = 1/6:0.001:1/3;
12 y1 = p(x1);
13
14 figure(1)
15 plot(beta,T_beta,'om',x1,y1,'-b');
16 hold on
17 plot([beta_ast(1) beta_ast(1)],[40 160],'--k');
18 plot([beta_ast(2) beta_ast(2)],[40 160],'--k');
19 hold off
```



Interpolando dados de um simulador



Interpolando dados de um simulador



Observações sobre interpolação monomial

- Os coeficientes c_j 's não têm uma relação óbvia com os valores y_j , o que dificulta sua interpretação;
- Se adicionarmos um novo ponto aos dados, os coeficientes precisarão ser recalculados para obter o novo polinômio interpolante;
- Quando o intervalo de interpolação for muito amplo, ou n for suficientemente grande, a matriz de Vandermonde pode ser mal condicionada (induzindo erros na solução numérica do sistema);
- Se n for suficientemente grande, o custo computacional de resolver o sistema linear $\sim \frac{2}{3}n^3$, ou a quantidade de memória para armazenar o mesmo $n^2 + 2n$, podem ser questões críticas.



Observações sobre interpolação monomial

- Os coeficientes c_j 's não têm uma relação óbvia com os valores y_j , o que dificulta sua interpretação;
- Se adicionarmos um novo ponto aos dados, os coeficientes precisarão ser recalculados para obter o novo polinômio interpolante;
- Quando o intervalo de interpolação for muito amplo, ou n for suficientemente grande, a matriz de Vandermonde pode ser mal condicionada (induzindo erros na solução numérica do sistema);
- Se n for suficientemente grande, o custo computacional de resolver o sistema linear $\sim 2/3 n^3$, ou a quantidade de memória para armazenar o mesmo $n^2 + 2n$, podem ser questões críticas.

**Todas essas questões podem ser contornadas
utilizando outras funções base!**



Interpolação de Lagrange

A **função interpolante** é assumida como sendo

$$p(x) = y_0 L_0^n(x) + y_1 L_1^n(x) + \cdots + y_n L_n^n(x),$$

onde os **polinômios de Lagrange** são dados por

$$L_j^n(x) = \prod_{\substack{i=0 \\ i \neq j}}^n \frac{x - x_i}{x_j - x_i}.$$

Note que $L_j^n(x_i) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$



Interpolação de Lagrange

$$\begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} c_n \\ c_{n-1} \\ \vdots \\ c_0 \end{bmatrix} = \begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_0 \end{bmatrix}$$

Sistema linear com solução trivial ($c_j = y_j$)



Interpolação de Lagrange

$$\begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} c_n \\ c_{n-1} \\ \vdots \\ c_0 \end{bmatrix} = \begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_0 \end{bmatrix}$$

Sistema linear com solução trivial ($c_j = y_j$)

Não é preciso resolver um sistema linear para construir o polinômio de Lagrange, basta construir as funções base.

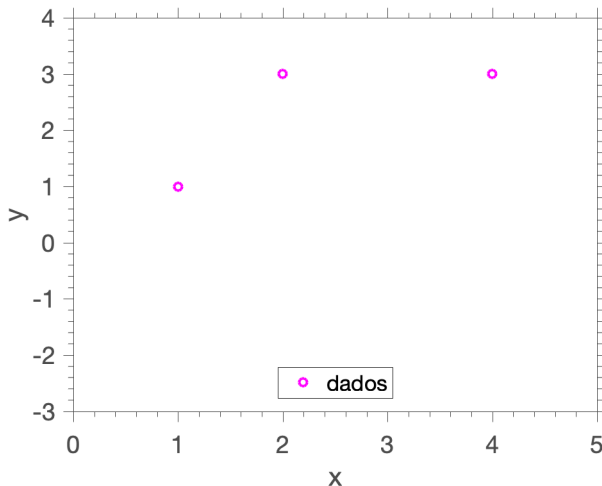


Experimento computacional 3

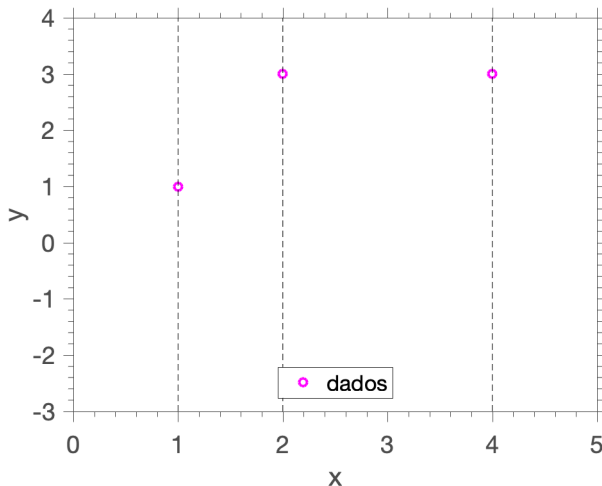
```
1  clc; clear; close all
2
3  xdat = [1;2;4]; ydat = [1;3;3];
4
5  L0 = @(x) (1/3)*(x-2).*(x-4);
6  L1 = @(x) -(1/2)*(x-1).*(x-4);
7  L2 = @(x) (1/6)*(x-1).*(x-2);
8  p = @(x) ydat(3)*L2(x) + ydat(2)*L1(x) + ydat(1)*L0(x);
9
10 xgrid = 0:0.01:5; ygrid = p(xgrid);
11
12 plot(xdat,ydat,'om','LineWidth',2);
13 hold on
14 plot(xgrid,L2(xgrid),'--r','LineWidth',1);
15 plot(xgrid,L1(xgrid),'-.c','LineWidth',1);
16 plot(xgrid,L0(xgrid),' :g','LineWidth',1);
17 plot(xgrid,ygrid,'-b','LineWidth',2);
18 plot([xdat(1),xdat(1)],[-3,4],'--k','LineWidth',0.5);
19 plot([xdat(2),xdat(2)],[-3,4],'--k','LineWidth',0.5);
20 plot([xdat(3),xdat(3)],[-3,4],'--k','LineWidth',0.5);
21 hold off
22 xlabel('x'); ylabel('y'); set(gca,'FontSize',18);
23 legend('dados','L2','L1','L0','p(x)','Location','south');
```



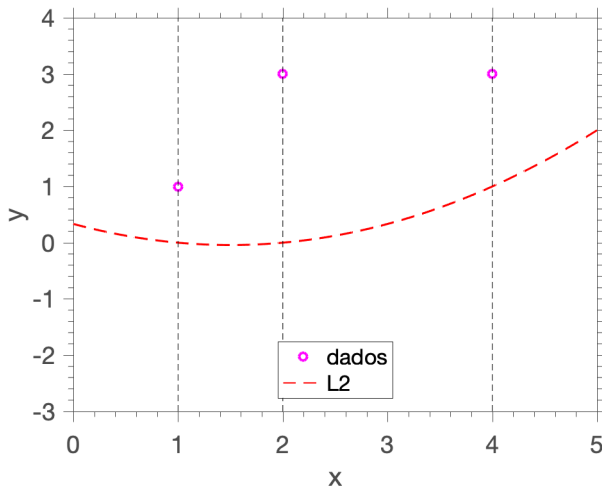
Experimento computacional 3



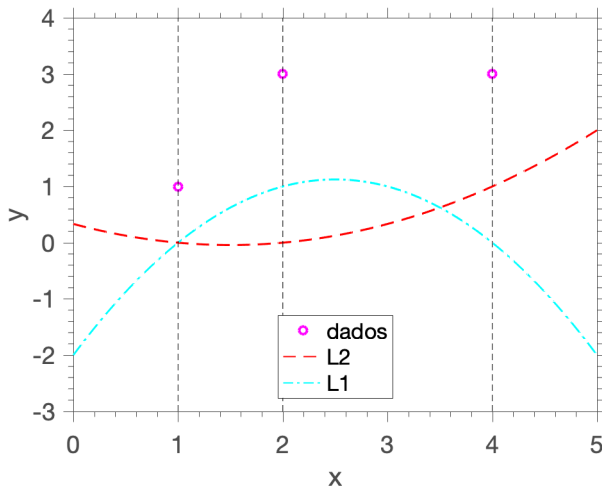
Experimento computacional 3



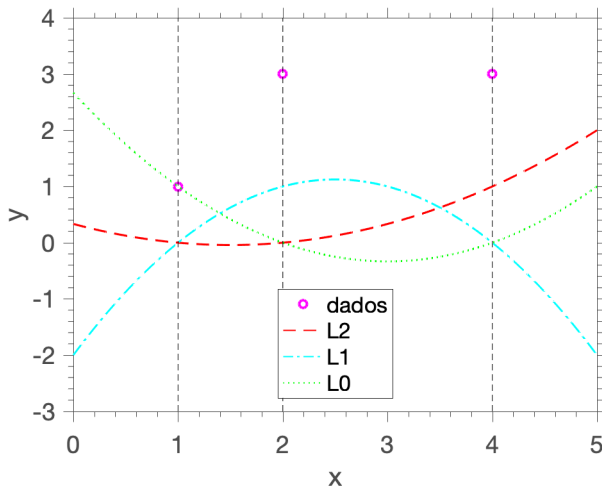
Experimento computacional 3



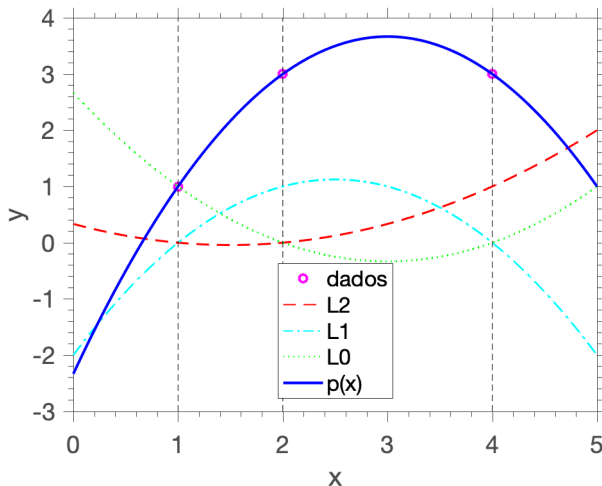
Experimento computacional 3



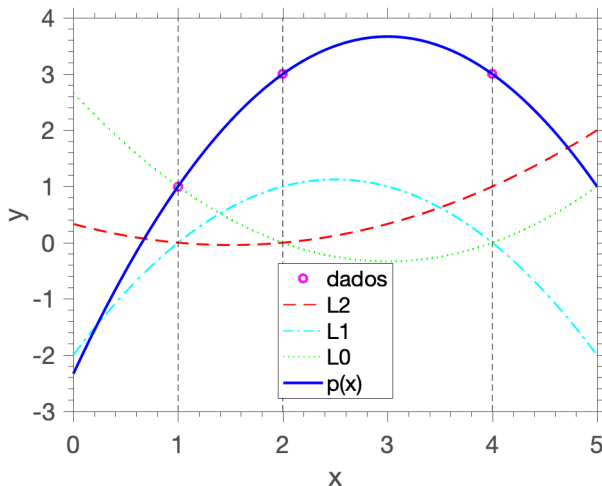
Experimento computacional 3



Experimento computacional 3



Experimento computacional 3



$$p(x) = -\frac{2}{3}x^2 + 4x - \frac{7}{3}$$



Implementação em GNU Octave

```
1 function y = interplagrange(xdat,ydat,x)
2     n = length(xdat);
3     L = ones(n,length(x));
4     for i = 1:n
5         for j = 1:n
6             if i ~= j
7                 L(i,:) = L(i,:).*(x-xdat(j))./(xdat(i)-xdat(j));
8             end
9         end
10    end
11    y = ydat'*L;
12 end
```

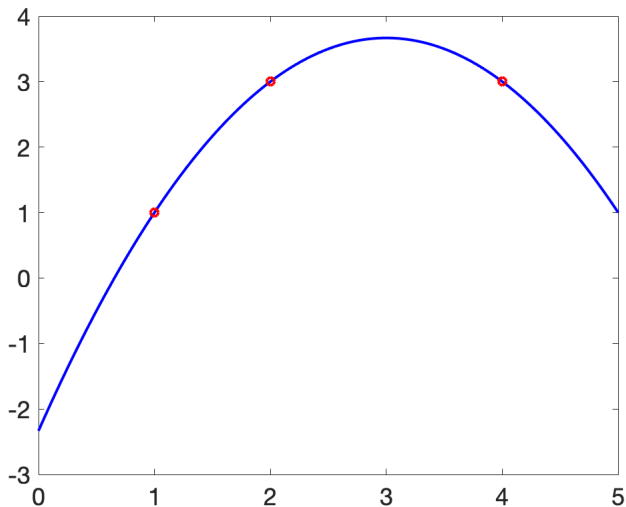


Experimento computacional 3 (revisado)

```
1  clc; clear; close all
2
3  xdat = [1; 2; 4];
4  ydat = [1; 3; 3];
5
6  xgrid = 0:0.01:5;
7  ygrid = interplagrange(xdat,ydat,xgrid);
8
9  plot(xgrid,ygrid,'b',xdat,ydat,'or');
```



Experimento computacional 3 (revisado)



Observações sobre interpolação de Lagrange

- Os coeficientes agora têm um significado claro, eles são iguais aos valores y_j ;
- Se adicionarmos um novo ponto aos dados, os coeficientes precisarão ser recalculados para obter o novo polinômio interpolante;
- Como a matriz do sistema é a identidade, o processo de construção do polinômio é muito estável. Na prática não se resolve o sistema diagonal, existe um algoritmo bem eficiente baseado em interpolação baricêntrica.



Interpolação de Newton

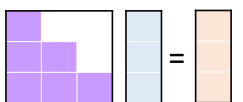
A **função interpolante** é assumida como sendo

$$p(x) = c_0 + c_1(x - x_0) + \cdots + c_n(x - x_0)(x - x_1) \cdots (x - x_{n-1}),$$

i.e., as **funções base** são da forma

$$\phi_j(x) = \prod_{i=0}^{n-1} (x - x_i).$$

Os **coeficientes** são definidos por um **sistema triangular inferior**

$$\begin{array}{ccc} A & \mathbf{x} & \mathbf{b} \\ n \times n & n \times 1 & n \times 1 \end{array}$$


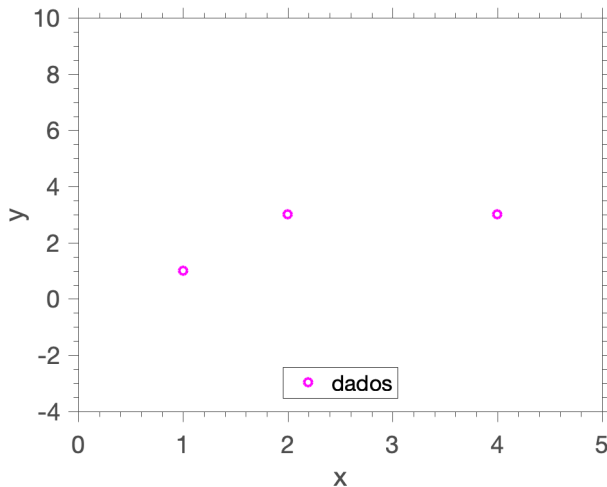


Experimento computacional 4

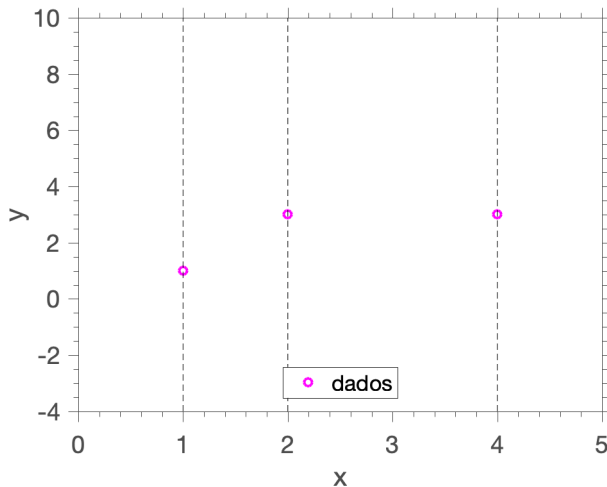
```
1  clc; clear; close all
2
3  xdat = [1;2;4]; ydat = [1;3;3];
4
5  p0 = @(x) 1*ones(size(x));
6  p1 = @(x) p0(x) + 2*(x-1);
7  p2 = @(x) p1(x) - (2/3)*(x-1).*(x-2);
8
9  xgrid = 0:0.01:5;
10
11 plot(xdat,ydat,'om','LineWidth',2);
12 hold on
13 plot(xgrid,p0(xgrid),'--r','LineWidth',2);
14 plot(xgrid,p1(xgrid),'-.c','LineWidth',2);
15 plot(xgrid,p2(xgrid),'-b','LineWidth',2);
16 plot([xdat(1),xdat(1)],[-4,10],'--k','LineWidth',0.5);
17 plot([xdat(2),xdat(2)],[-4,10],'--k','LineWidth',0.5);
18 plot([xdat(3),xdat(3)],[-4,10],'--k','LineWidth',0.5);
19 hold off
20 xlabel('x'); ylabel('y'); set(gca,'FontSize',18);
21 xlim([0 5]); ylim([-4 10]);
22 legend('dados','p_0(x)','p_1(x)','p_2(x)','Location','south')
```



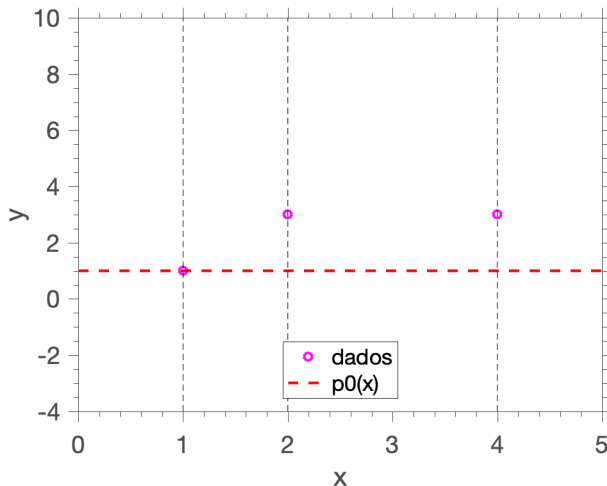
Experimento computacional 4



Experimento computacional 4

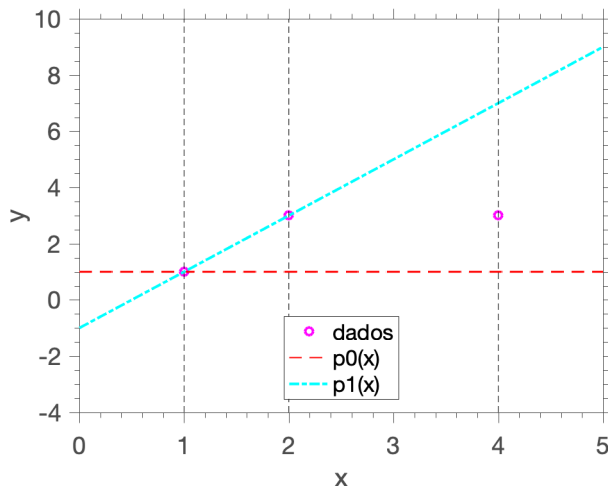


Experimento computacional 4



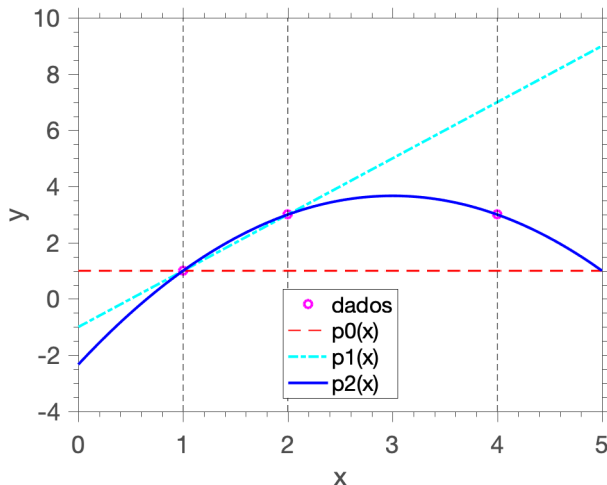
$$p_0(x) = 1$$

Experimento computacional 4



$$p_1(x) = p_0(x) + c_1(x - 1) = 1 + 2(x - 1)$$

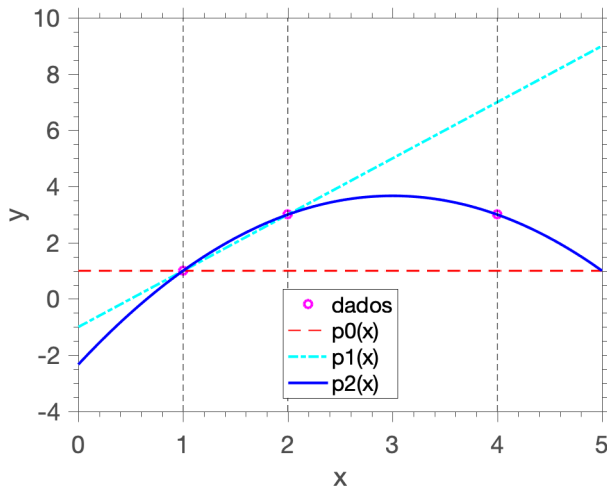
Experimento computacional 4



$$p_2(x) = p_1(x) + c_2(x-1)(x-2) = 1 + 2(x-1) - \frac{2}{3}(x-1)(x-2)$$



Experimento computacional 4



$$p_2(x) = -\frac{2}{3}x^2 + 4x - \frac{7}{3}$$



Implementação em GNU Octave

```
1 function y = interpnewton(x,xdat,coef)
2     n = length(xdat);
3     y = coef(n)*ones(size(x));
4     for j=n-1:-1:1
5         y = y.*(x - xdat(j)) + coef(j);
6     end
7 end
```

```
1 function [coef,tab] = divdif(xdat,ydat)
2     n = length(xdat)-1;
3     tab = zeros(n+1,n+1);
4     xdat = shiftdim(xdat);
5     ydat = shiftdim(ydat);
6     tab(1:n+1,1) = ydat;
7     for k = 2:n+1
8         num = tab(k:n+1,k-1)-tab(k-1:n,k-1);
9         den = xdat(k:n+1)-xdat(1:n+1-k+1);
10        tab(k:n+1,k) = num./den;
11    end
12    coef = diag(tab);
13 end
```



Implementação em GNU Octave

```
1 function [coef,tab] = divdifadd(xdat,ydat,tab)
2     n = length(xdat)-1;
3     tab = [tab zeros(n,1); ydat(n+1) zeros(1,n)];
4     for k=2:n+1
5         num = tab(n+1,k-1)-tab(n,k-1);
6         den = xdat(n+1)-xdat(n+1-k+1);
7         tab(n+1,k) = num/den;
8     end
9     coef = diag(tab);
10 end
```

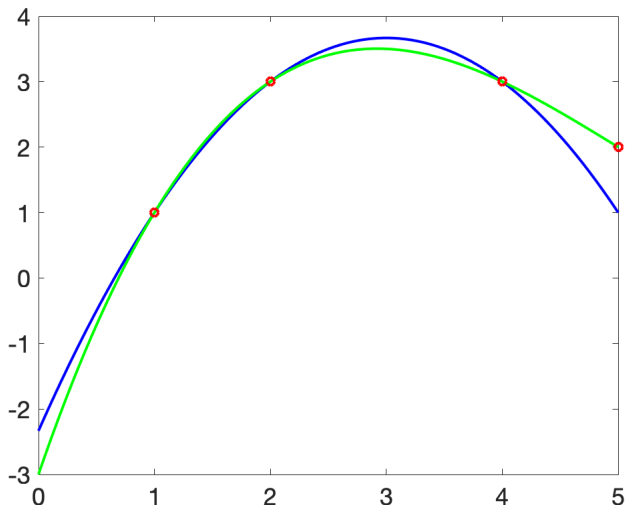


Experimento computacional 4 (revisado)

```
1  clc; clear; close all
2
3  xdat1 = [1; 2; 4];    ydat1 = [1; 3; 3];
4  xdat2 = [1; 2; 4; 5]; ydat2 = [1; 3; 3; 2];
5
6  [c1,tab1] = divdif(xdat1,ydat1);
7  [c2,tab2] = divdifadd(xdat2,ydat2,tab1);
8
9  xgrid = 0:0.01:5;
10 ygrid1 = interpnewton(xgrid,xdat1,c1);
11 ygrid2 = interpnewton(xgrid,xdat2,c2);
12
13 figure(1)
14 plot(xgrid,ygrid1,'b');
15 hold on
16 plot(xgrid,ygrid2,'g',xdat2,ydat2,'or');
17 hold off
```



Experimento computacional 4 (revisado)



Custo e características do problema de interpolação

base	$\phi_j(x)$	custo de construção	custo de avaliação	características notáveis
monomial	x^j	$\sim \frac{2}{3} n^3$	$\sim 2 n$	simplicidade
Lagrange	$\prod_{\substack{i=0 \\ i \neq j}}^n \frac{x - x_i}{x_j - x_i}$	$\sim 2 n^2$	$\sim 5 n$	interpretabilidade e estabilidade
Newton	$\prod_{i=0}^{j-1} (x - x_i)$	$\sim \frac{3}{2} n^2$	$\sim 2 n$	adaptatividade



Erro na interpolação polinomial

Teorema (Erro da interpolação polinomial)

Se o polinômio $p_n(x)$ interpola a função $f \in C^{n+1}([a, b])$ nos pontos $a = x_0 < x_1 < \dots < x_n = b$, então o erro da interpolação é dado por

$$f(x) - p_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{i=0}^n (x - x_i),$$

onde $\xi \in [a, b]$.



Como citar esse material?


A. Cunha Jr, *Interpolação*,
Universidade do Estado do Rio de Janeiro – UERJ, 2021.



 @AmericoCunhaJr

 @AmericoCunhaJr

 @AmericoCunhaJr

 @AmericoCunhaJr

Essas notas de aula podem ser compartilhadas nos termos da licença Creative Commons BY-NC-ND 3.0, com propósitos exclusivamente educacionais.

