CCI-22

Matemática Computacional

Carlos Alberto Alonso Sanches

CCI-22

5) Interpolação

Polinômios interpoladores,
Formas de Lagrange, de Newton e de Newton-Gregory

CCI-22

- Introdução
- Forma de Lagrange
- Forma de Newton
- Forma de Newton-Gregory
- Interpolação inversa
- Estudo do erro
- Convergência
- Funções *splines*

CCI-22

- Introdução
- Forma de Lagrange
- Forma de Newton
- Forma de Newton-Gregory
- Interpolação inversa
- Estudo do erro
- Convergência
- Funções *splines*

Definição

- *Interpolar* significa situar entre dois polos, intercalar, interserir
- Dados alguns valores discretos de uma determinada função, sua interpolação consiste em determinar outra função (em geral, um polinômio) que seja contínua e que coincida nesses pontos
- Exemplo: calor específico da água

°C	20	25	30	35	40	45	50
Calor específico	0,99907	0,99852	0,99826	0,99818	0,99828	0,99849	0,99878

- Esse processo também pode ser útil quando se deseja substituir uma função de difícil integração ou derivação

Formalização

- Dados n+1 valores distintos $x_0, x_1, \ldots, x_n$, chamados nós ou pontos de interpolação, e os respectivos valores $f(x_0), f(x_1), \ldots, f(x_n)$, deseja-se determinar um polinômio interpolador $p_n(x)$ de grau máximo n tal que $p_n(x_i) = f(x_i)$, $0 \le i \le n$
- Como $x_i \ne x_j$, para $0 \le i,j \le n$, $i \ne j$, então $p_n(x)$ é único
- Demonstração:
  - Seja $p_n(x) = a_0 + a_1.x + a_2.x^2 + \ldots + a_n.x^n$

$$Xa = y \quad \text{onde: } X = \begin{bmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \cdots & x_n^n \end{bmatrix} \quad a = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} \quad y = \begin{bmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix}$$

Matriz de Vandermonde

Se os x_i são distintos, então $\det(X) \ne 0$

Modos de se obter $p_n(x)$

- Há três modos de se calcular $p_n(x)$:
  - Resolução do sistema linear com matriz de Vandermonde
  - Forma de Lagrange
  - Forma de Newton (diferenças divididas)
- Uma alternativa mais simples de interpolação é, ao invés de calcular $p_n(x)$, fazer interpolações em cada grupo de dois três pontos. Esses casos são chamados de interpolação *linear* ou *quadrática*, respectivamente

CCI-22

- Introdução
- Forma de Lagrange
- Forma de Newton
- Forma de Newton-Gregory
- Interpolação inversa
- Estudo do erro
- Convergência
- Funções *splines*

Forma de Lagrange

- Sejam n+1 pontos distintos $x_0, x_1, ..., x_n$ e $y_i = f(x_i)$, $0 \le i \le n$. Seja $p_n(x)$ o polinômio de grau máximo n que interpola f nesses pontos
- $p_n(x)$ pode ser representado do seguinte modo:
  - $p_n(x) = y_0.L_0(x) + y_1.L_1(x) + ... + y_n.L_n(x)$
  - $L_k(x)$ são polinômios de grau n, $0 \le k \le n$
- Deseja-se que $p_n(x_i) = y_i$, $0 \le i \le n$
- Há um modo simples de satisfazer essas condições:
  - $L_k(x_i) = 0$, se $k \ne i$
  - $L_k(x_i) = 1$, se $k = i$
- Basta definir:

$$L_k(x) = \frac{(x - x_0)\cdot(x - x_1)\ldots(x - x_{k-1})\cdot(x - x_{k+1})\ldots(x - x_n)}{(x_k - x_0)\cdot(x_k - x_1)\ldots(x_k - x_{k-1})\cdot(x_k - x_{k+1})\ldots(x_k - x_n)}$$

Exemplo

x	-1	0	2
f(x)	4	1	-1

- Forma de Lagrange: $p_2(x) = y_0.L_0(x) + y_1.L_1(x) + y_2.L(x)$
  - $L_0(x) = (x - x_1)(x - x_2)/[(x_0 - x_1)(x_0 - x_2)] = (x^2 - 2x)/3$
  - $L_1(x) = (x - x_0)(x - x_2)/[(x_1 - x_0)(x_1 - x_2)] = (x^2 - x - 2)/(-2)$
  - $L_2(x) = (x - x_0)(x - x_1)/[(x_2 - x_0)(x_2 - x_1)] = (x^2 + x)/6$

$$p_2(x) = 4(x^2 - 2x)/3 + 1(x^2 - x - 2)/(-2) - 1(x^2 + x)/6$$

$$p_2(x) = 1 - (7/3)x + (2/3)x^2$$

CCI-22

- Introdução
- Forma de Lagrange
- Forma de Newton
- Forma de Newton-Gregory
- Interpolação inversa
- Estudo do erro
- Convergência
- Funções *splines*

Operador Diferenças Divididas

- Seja f(x) uma função tabelada em n+1 pontos distintos $x_0, x_1, ..., x_n$
- Definimos o *operador diferenças divididas* por:
  - $f[x_0] = f(x_0)$ — Ordem 0
  - $f[x_0, x_1] = (f[x_1] - f[x_0])/(x_1 - x_0)$ — Ordem 1
  - $f[x_0, x_1, x_2] = (f[x_1, x_2] - f[x_0, x_1])/(x_2 - x_0)$ — Ordem 2
  - $f[x_0, x_1, x_2, x_3] = (f[x_1, x_2, x_3] - f[x_0, x_1, x_2])/(x_3 - x_0)$ — Ordem 3
- Generalizando para a ordem n:
  - $f[x_0, x_1, x_2, ..., x_n] = (f[x_1, x_2, ..., x_n] - f[x_0, x_1, ..., x_{n-1}])/(x_n - x_0)$
- Dizemos que $f[x_0, x_1, ..., x_k]$ é a *diferença dividida de ordem k* da função f(x) sobre os k+1 pontos $x_0, x_1, ..., x_k$
- Prova-se que $f[x_0, x_1, ..., x_k]$ é simétrica nos argumentos
  - Exemplo: $f[x_0, x_1, x_2] = f[x_1, x_0, x_2] = f[x_2, x_1, x_0]$
  - A demonstração baseia-se em que $f[x_i, x_j] = f[x_j, x_i]$

Tabela de Diferenças Dividadas

- As diferenças divididas podem ser calculadas e armazenadas através da seguinte tabela:

x	Ordem 0	Ordem 1	Ordem 2	Ordem 3	...	Ordem n
x_0	$f[x_0]$					
		$f[x_0, x_1]$				
x_1	$f[x_1]$		$f[x_0, x_1, x_2]$			
		$f[x_1, x_2]$		$f[x_0, x_1, x_2, x_3]$		
x_2	$f[x_2]$		$f[x_1, x_2, x_3]$		...	
		$f[x_2, x_3]$		$f[x_1, x_2, x_3, x_4]$		
x_3	$f[x_3]$		$f[x_2, x_3, x_4]$			$f[x_0, x_1, x_2, ..., x_n]$
		$f[x_3, x_4]$		...	...	
x_4	$f[x_4]$		...	$f[x_{n-3}, x_{n-2}, x_{n-1}, x_n]$		
		...	$f[x_{n-2}, x_{n-1}, x_n]$			
...	...	$f[x_{n-1}, x_n]$				
x_n	$f[x_n]$					

Exemplo

x	-1	0	1	2	3
f(x)	1	1	0	-1	-2

x	Ordem 0	Ordem 1	Ordem 2	Ordem 3	Ordem 4
$x_0 = -1$	$f[x_0] = 1$				
		$f[x_0, x_1] = 0$			
$x_1 = 0$	$f[x_1] = 1$		$f[x_0, x_1, x_2] = -1/2$		
		$f[x_1, x_2] = -1$		$f[x_0, x_1, x_2, x_3] = 1/6$	
$x_2 = 1$	$f[x_2] = 0$		$f[x_1, x_2, x_3] = 0$		$f[x_0, x_1, x_2, x_3, x_4] = -1/24$
		$f[x_2, x_3] = -1$		$f[x_1, x_2, x_3, x_4] = 0$	
$x_3 = 2$	$f[x_3] = -1$		$f[x_2, x_3, x_4] = 0$		
		$f[x_3, x_4] = -1$			
$x_4 = 3$	$f[x_4] = -2$				

Forma de Newton

- Seja f(x) contínua e com tantas derivadas contínuas quantas necessárias num intervalo [a,b]. Sejam n+1 pontos nesse intervalo: $a = x_0 < x_1 < ... < x_n = b$
- O polinômio $p_n(x)$ de grau máximo n que interpola f(x) nesses pontos pode ser encontrado de modo construtivo:
  - Calcula-se $p_0(x)$ que interpola f(x) em x_0
  - Calcula-se $p_1(x)$ que interpola f(x) em x_0 e x_1
  - Calcula-se $p_2(x)$ que interpola f(x) em x_0, x_1 e x_2
  - Assim por diante, até $p_n(x)$

Cálculo de $p_0(x)$

- $p_0(x)$ é o polinômio de grau 0 que interpola f(x) em $x = x_0$. Então, $p_0(x) = f(x_0) = f[x_0]$
- Para todo $x \in [a,b]$, $x \neq x_0$:
  - $f[x,x_0] = (f[x_0] - f[x])/(x_0 - x)$
  - $f[x,x_0] = (f(x_0) - f(x))/(x_0 - x)$
  - $(x_0 - x).f[x,x_0] = f(x_0) - f(x)$
  - $f(x) = f(x_0) + (x - x_0).f[x,x_0]$
  - $f(x) = p_0(x) + \underbrace{(x - x_0).f[x,x_0]}_{E_0(x)\text{ : erro de aproximação}}$

Cálculo de $p_1(x)$

- $p_1(x)$ é o polinômio de grau máximo 1 que interpola $f(x)$ em x_0 e x_1
- Para todo $x \in [a,b]$, $x \neq x_0$ e $x \neq x_1$:
  - $f[x,x_0,x_1] = (f[x_0,x_1] - f[x,x_0])/(x_1 - x)$
  - $f[x,x_0,x_1] = (f[x,x_0] - f[x_0,x_1])/(x - x_1)$
  - $f[x,x_0,x_1] = (f[x_0,x] - f[x_1,x_0])/(x - x_1)$
  - $f[x,x_0,x_1] = ((f(x) - f(x_0))/(x - x_0) - f[x_1,x_0])/(x - x_1)$
  - $f[x,x_0,x_1] = (f(x) - f(x_0) - (x - x_0)f[x_1,x_0])/((x - x_0)(x - x_1))$
  - $f(x) = \underbrace{f(x_0) + (x - x_0)f[x_1,x_0]}_{p_1(x)} + \underbrace{(x - x_0)(x - x_1)f[x,x_0,x_1]}_{E_1(x)}$
  - $p_1(x) = \underbrace{f(x_0)}_{p_0(x)} + \underbrace{(x - x_0)f[x_0,x_1]}_{q_1(x)}$

Generalização

- Analogamente, é possível verificar que:
  - $p_2(x) = \underbrace{f(x_0) + (x - x_0)f[x_0,x_1]}_{p_1(x)} + \underbrace{(x - x_0)(x - x_1)f[x_0,x_1,x_2]}_{q_2(x)}$
  - $E_2(x) = (x - x_0)(x - x_1)(x - x_2)f[x,x_0,x_1,x_2]$
- Ao generalizarmos esses resultados, encontramos $p_n(x)$ e seu correspondente erro de aproximação:
  - $p_n(x) = f(x_0) + (x - x_0)f[x_0,x_1] + \ldots + (x - x_0)\ldots(x - x_{n-1})f[x_0,x_1,\ldots,x_n]$
  - $E_n(x) = (x - x_0)(x - x_1)\ldots(x - x_n)f[x,x_0,x_1,\ldots,x_n]$
- Podemos comprovar que, de fato, $p_n(x)$ interpola $f(x)$ em $x_0, x_1, \ldots, x_n$:
  - $f(x) = p_n(x) + E_n(x)$
  - $f(x_k) = p_n(x_k) + E_n(x_k) = p_n(x_k)$, para $0 \le k \le n$

Exemplo (já visto)

x	-1	0	2
$f(x)$	4	1	-1

x	Ordem 0	Ordem 1	Ordem 2
$x_0 = -1$	$f[x_0] = 4$		
		$f[x_0, x_1] = -3$	
$x_1 = 0$	$f[x_1] = 1$		$f[x_0, x_1, x_2] = 2/3$
		$f[x_1, x_2] = -1$	
$x_2 = 2$	$f[x_2] = -1$		

$$p_2(x) = 4 + (x + 1)(-3) + (x + 1)(x - 0)(2/3)$$

$$p_2(x) = (2/3)x^2 - (7/3)x + 1$$

Mesmo resultado

CCI-22

- Introdução
- Forma de Lagrange
- Forma de Newton
- Forma de Newton-Gregory
- Interpolação inversa
- Estudo do erro
- Convergência
- Funções *splines*

Forma de Newton-Gregory

- Quando os nós da interpolação são *igualmente espaçados*, $p_n(x)$ pode ser obtido pela forma de Newton-Gregory
- Sejam $x_0, x_1, \ldots, x_n$ pontos que se sucedem com passo h. Chamamos Δ de *operador de diferenças ordinárias*:
  - $\Delta f(x) = f(x + h) - f(x)$
  - $\Delta^2 f(x) = \Delta f(x + h) - \Delta f(x)$
  - $\Delta^n f(x) = \Delta^{n-1} f(x + h) - \Delta^{n-1} f(x)$
- Naturalmente, $\Delta^0 f(x) = f(x)$

Tabela de Diferenças Ordinárias

- Analogamente às diferenças divididas, podemos usar uma tabela para armazenar as diferenças ordinárias:

x	f(x)	Δf(x)	Δ²f(x)	Δ³f(x)	...	Δⁿf(x)
x_0	$f(x_0)$					
		$\Delta f(x_0)$				
x_1	$f(x_1)$		$\Delta^2 f(x_0)$			
		$\Delta f(x_1)$		$\Delta^3 f(x_0)$		
x_2	$f(x_2)$		$\Delta^2 f(x_1)$		...	
		$\Delta f(x_2)$		$\Delta^3 f(x_1)$		
x_3	$f(x_3)$		$\Delta^2 f(x_2)$			$\Delta^n f(x_0)$
		$\Delta f(x_3)$		...	...	
x_4	$f(x_4)$		...	$\Delta^3 f(x_{n-3})$		
		...	$\Delta^2 f(x_{n-2})$			
...	...	$\Delta f(x_{n-1})$				
x_n	$f(x_n)$					

Exemplo

x	-1	0	1	2	3
f(x)	2	1	2	5	10

x	f(x)	Δf(x)	Δ²f(x)	Δ³f(x)	Δ⁴f(x)
$x_0 = -1$	$f(x_0) = 2$				
		$\Delta f(x_0) = -1$			
$x_1 = 0$	$f(x_1) = 1$		$\Delta^2 f(x_0) = 2$		
		$\Delta f(x_1) = 1$		$\Delta^3 f(x_0) = 0$	
$x_2 = 1$	$f(x_2) = 2$		$\Delta^2 f(x_1) = 2$		$\Delta^4 f(x_0) = 0$
		$\Delta f(x_2) = 3$		$\Delta^3 f(x_1) = 0$	
$x_3 = 2$	$f(x_3) = 5$		$\Delta^2 f(x_2) = 2$		
		$\Delta f(x_3) = 5$			
$x_4 = 3$	$f(x_4) = 10$				

Relação com Diferenças Divididas

- Por indução, é possível demonstrar que, quando os pontos $x_0, x_1, \ldots, x_n$ se sucedem com passo h, então $f[x_0, x_1, \ldots, x_n] = \Delta^n f(x_0)/(h^n n!)$
  - Base: n=1
    - $f[x_0, x_1] = (f(x_1) - f(x_0))/(x_1 - x_0) = \Delta f(x_0)/h = \Delta^1 f(x_0)/(h^1 1!)$
  - Suponhamos que seja válido para n-1:
    - $f[x_0, \ldots, x_{n-1}] = \Delta^{n-1} f(x_0)/(h^{n-1}(n-1)!)$
  - Passo:
    - $f[x_0, x_1, \ldots, x_n] = (f[x_1, \ldots, x_n] - f[x_0, \ldots, x_{n-1}])/(x_n - x_0)$
    - $f[x_0, x_1, \ldots, x_n] = (\Delta^{n-1} f(x_1)/(h^{n-1}(n-1)!) - \Delta^{n-1} f(x_0)/(h^{n-1}(n-1)!))/nh$
    - $f[x_0, x_1, \ldots, x_n] = (\Delta^{n-1} f(x_1) - \Delta^{n-1} f(x_0)/nh.(h^{n-1}(n-1)!))$
    - $f[x_0, x_1, \ldots, x_n] = \Delta^n f(x_0)/(h^n n!)$

Polinômio interpolador

- Desse modo, é possível encontrar uma fórmula específica de $p_n(x)$ para este caso particular:
  - $p_n(x) = f(x_0) + (x - x_0)f[x_0,x_1] + (x - x_0)(x - x_1)f[x_0,x_1,x_2] + ... + (x - x_0)...(x - x_{n-1})f[x_0,x_1,...,x_n]$
  - $p_n(x) = f(x_0) + (x - x_0)\Delta f(x_0)/h + (x - x_0)(x - x_1)\Delta^2 f(x_0)/(2h^2) + ... + (x - x_0)...(x - x_{n-1})\Delta^n f(x_0)/(h^n n!)$
- Uma mudança de variável pode simplificar a expressão de $p_n(x)$:
  - $s = (x - x_0)/h \Rightarrow x = sh + x_0$
  - Para $0 \leq i \leq n$:
    - $x - x_i = sh + x_0 - x_i$
    - $x - x_i = sh + x_0 - (x_0 + ih)$
    - $x - x_i = (s - i)h$
  - $p_n(x) = f(x_0) + s\Delta f(x_0) + s(s-1)\Delta^2 f(x_0)/2 + ... + s(s-1)...(s-n+1)\Delta^n f(x_0)/n!$

Voltando ao exemplo anterior

x	-1	0	1	2	3
f(x)	2	1	2	5	10

x	f(x)	Δf(x)	Δ²f(x)	Δ³f(x)	Δ⁴f(x)
-1	2	-1	2	0	0
0	1	1	2	0	
1	2	3	2		
2	5	5			
3	10				

- $x_0 = -1$, $h = 1 \Rightarrow s = (x - x_0)/h = x+1$
- $p_4(x) = f(x_0) + s\Delta f(x_0) + s(s-1)\Delta^2 f(x_0)/2 + s(s-1)(s-2)\Delta^3 f(x_0)/3! + s(s-1)(s-2)(s-3)\Delta^4 f(x_0)/4!$
- $p_4(x) = 2 + (x+1)(-1) + (x+1)x2/2$
- $p_4(x) = x^2 + 1$
- Repare que o grau desse polinômio é 2...

CCI-22

- Introdução
- Forma de Lagrange
- Forma de Newton
- Forma de Newton-Gregory
- Interpolação inversa
- Estudo do erro
- Convergência
- Funções *splines*

Interpolação inversa

- Seja f(x) uma função tabelada em n+1 pontos distintos $x_0, x_1, ..., x_n$
- Dado $\hat{y} \in (f(x_0), f(x_n))$, o problema da *interpolação inversa* consiste em encontrar x^* tal que $f(x^*) = \hat{y}$
- Há dois modos de se resolver este problema:
  1) Obter $p_n(x)$ que interpola f(x) em $x_0, x_1, ..., x_n$ e em seguida encontrar x^* tal que $p_n(x^*) = \hat{y}$
     - Será preciso encontrar a raiz de um polinômio
  2) Se f(x) for inversível no intervalo que contém $\hat{y}$, fazer a interpolação de $f^{-1}(x)$
     - Isso somente será possível se f(x) for contínua e monotônica nesse intervalo

Exemplo 1)

x	0,5	0,6	0,7	0,8	0,9	1,0
f(x)	1,65	1,82	2,01	2,23	2,46	2,72

- Deseja-se encontrar x^* tal que $f(x^*) = 2$
- Como $2 \in (1{,}82;\ 2{,}01)$, usaremos interpolação linear sobre $x_0 = 0{,}6$ e $x_1 = 0{,}7$
- Através de Lagrange:
  - $p_1(x) = f(x_0)(x - x_1)/(x_0 - x_1) + f(x_1)(x - x_0)/(x_1 - x_0)$
  - $p_1(x) = 1{,}9x + 0{,}68$
- $p_1(x^*) = 2 \Leftrightarrow 1{,}9x^* + 0{,}68 = 2 \Leftrightarrow x^* = 0{,}6947368$
- Neste caso, não é possível fazer nenhuma estimativa sobre o erro cometido...

Exemplo 2)

x	0	0,1	0,2	0,3	0,4	0,5
$y = e^x$	1	1,1052	1,2214	1,3499	1,4918	1,6487

- Deseja-se encontrar x^* tal que $e^{x^*} = 1{,}3165$
- Usaremos interpolação quadrática em $f^{-1}(x)$

	y	Ordem 0	Ordem 1	Ordem 2	Ordem 3
	1	0			
			0,9506		
	1,1052	0,1		-0,4065	
			0,8606		0,1994
y_0	1,2214	0,2		-0,3367	
			0,7782		0,1679
y_1	1,3499	0,3		-0,2718	
			0,7047		
y_2	1,4918	0,4			

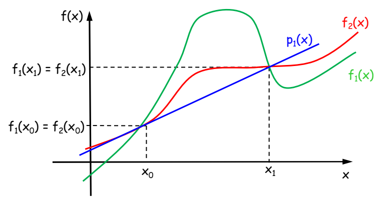
- $p_2(y) = g(y_0) + (y - y_0)g[y_0,y_1] + (y - y_0)(y - y_1)g[y_0,y_1,y_2]$
- $p_2(y) = 0{,}2 + (y - 1{,}2214).0{,}7782 + (y - 1{,}2214)(y - 1{,}3499).(-0{,}2718)$
- $p_2(1{,}3165) = 0{,}27487$
- Pode-se verificar que $e^{0{,}27487} = 1{,}31659$
- Neste caso, é possível estimar o erro, como veremos a seguir...

CCI-22

- Introdução
- Forma de Lagrange
- Forma de Newton
- Forma de Newton-Gregory
- Interpolação inversa
- Estudo do erro
- Convergência
- Funções *splines*

Estudo do erro na interpolação

- Ao aproximarmos uma função $f(x)$ por um polinômio $p_n(x)$ no intervalo $[x_0,x_n]$, comete-se um erro $E_n(x) = f(x) - p_n(x)$
- No exemplo abaixo, considere $p_1(x)$ que interpola $f_1(x)$ e $f_2(x)$ no intervalo $[x_0,x_1]$:



- $E_1^1(x) = f_1(x) - p_1(x)$
- $E_1^2(x) = f_2(x) - p_1(x)$
- $E_1^1(x) > E_1^2(x),\ \forall x \in (x_0,x_1)$

Erro de aproximação

- Teorema: Sejam n+1 pontos $x_0, x_1, \ldots, x_n$. Seja $f(x)$ com derivadas até ordem n+1 para todo x pertencente ao intervalo $[x_0,x_n]$. Seja $p_n(x)$ o polinômio interpolador de $f(x)$ nesses pontos. Então, $\forall x \in [x_0,x_n]$, o erro é dado por $E_n(x) = (x - x_0)(x - x_1)\ldots(x - x_n)f^{(n+1)}(\xi)/(n+1)!$, onde $\xi \in (x_0,x_n)$
- Demonstração:
  - Seja $G(x) = (x - x_0)(x - x_1)\ldots(x - x_n)$, $\forall x \in [x_0,x_n]$
  - Lembrando que $E_n(x) = f(x) - p_n(x)$, seja:
    - $H(t) = E_n(x)G(t) - E_n(t)G(x)$, $t \in [x_0,x_n]$
  - $H(t)$ possui derivadas até ordem n+1, pois $f(t)$ – por hipótese -, $p_n(t)$ e $G(t)$ possuem derivadas até essa ordem
  - $H(t)$ possui pelo menos n+2 raízes em $[x_0,x_n]$:
    - $x_0, x_1, \ldots, x_n$ são raízes de $H(t)$
    - x é raiz de $H(t)$

Demonstração (continuação)

- No intervalo $[x_0,x_n]$, $H(t)$ está definida, possui derivadas até ordem n+1, e tem pelo menos n+2 raízes. Portanto, podemos aplicar sucessivamente o Teorema de Rolle a $H(t), H'(t), \ldots, H^{(n)}(t)$:
  - $H'(t)$ possui pelo menos n+1 raízes em (x_0,x_n)
  - $H''(t)$ possui pelo menos n raízes em (x_0,x_n)
  - ...
  - $H^{(n+1)}(t)$ possui pelo menos uma raiz em (x_0,x_n)
- $H(t) = E_n(x)G(t) - E_n(t)G(x) \Rightarrow H^{(n+1)}(t) = E_n(x)G^{(n+1)}(t) - E_n^{(n+1)}(t)G(x)$
- $E_n(x) = f(x) - p_n(x) \Rightarrow E_n^{(n+1)}(t) = f^{(n+1)}(t) - p_n^{(n+1)}(t) = f^{(n+1)}(t)$
- $G(t) = (t - x_0)(t - x_1)\ldots(t - x_n) \Rightarrow G^{(n+1)}(t) = (n+1)!$
- Substituindo:
  - $H^{(n+1)}(t) = E_n(x)(n+1)! - f^{(n+1)}(t)G(x)$
- Seja $\xi \in (x_0,x_n)$ uma raiz de $H^{(n+1)}(t)$:
  - $H^{(n+1)}(\xi) = E_n(x)(n+1)! - f^{(n+1)}(\xi)G(x) = 0$
  - $E_n(x) = f^{(n+1)}(\xi)G(x)/(n+1)!$
  - $E_n(x) = (x - x_0)(x - x_1)\ldots(x - x_n)f^{(n+1)}(\xi)/(n+1)!$

Algumas conclusões

- Sabemos que $E_n(x) = (x - x_0)(x - x_1)\ldots(x - x_n) f^{(n+1)}(\xi)/(n+1)!$, onde $\xi \in (x_0,x_n)$
- Como há $(n+1)!$ no denominador de $E_n(x)$, parece que, quando n aumenta, o erro de aproximação tende a diminuir...
- No entanto, raramente é possível calcular $f^{(n+1)}(x)$, e ξ nunca é conhecido...
- Veremos a seguir como esse erro pode ser estimado através das diferenças divididas de ordem n+1

Estimativa para o erro

- Pelo teorema anterior:
  - $E_n(x) = (x - x_0)(x - x_1)\ldots(x - x_n)f^{(n+1)}(\xi)/(n+1)!$
  - $|E_n(x)| \leq |(x - x_0)(x - x_1)\ldots(x - x_n)|.M_{n+1}/(n+1)!$, onde $M_{n+1} = \text{máx}_{x \in I} |f^{(n+1)}(x)|$ e $I = [x_0,x_n]$
- Pela forma de Newton:
  - $E_n(x) = (x - x_0)(x - x_1)\ldots(x - x_n)f[x,x_0,x_1,\ldots,x_n]$
- Conclusões:
  - $f[x,x_0,x_1,\ldots,x_n] = f^{(n+1)}(\xi)/(n+1)!$, onde $\xi \in (x_0,x_n)$
  - $M_{n+1}/(n+1)! = \text{máx}_{x \in I} |f[x,x_0,x_1,\ldots,x_n]|$
  - Seja D o máximo dos módulos das diferenças divididas de ordem n+1 que foram calculadas
  - $|E_n(x)| \approx |(x - x_0)(x - x_1)\ldots(x - x_n)|.D$

Exemplo

x	0,2	0,34	0,4	0,52	0,6	0,72
f(x)	0,16	0,22	0,27	0,29	0,32	0,37

- Deseja-se obter f(0,47) usando um polinômio de grau 2, com uma estimativa para o erro

	x	Ordem 0	Ordem 1	Ordem 2	Ordem 3
	0,2	0,16			
			0,4286		
	0,34	0,22		2,0235	
			0,8333		-17,8963
x_0	0,4	0,27		-3,7033	
			0,1667		18,2494
x_1	0,52	0,29		1,0415	
			0,375		-2,6031
x_2	0,6	0,32		0,2085	
			0,4167		
	0,72	0,37			

- $p_2(x) = f(x_0) + (x - x_0)f[x_0,x_1] + (x - x_0)(x - x_1)f[x_0,x_1,x_2]$
- $p_2(x) = 0,27 + (x - 0,4)0,1667 + (x - 0,4)(x - 0,52)1,0415$
- $p_2(0,47) = \mathbf{0,2780} \approx f(0,47)$
- $|E_2(0,47)| \approx |(0,47 - 0,4)(0,47 - 0,52)(0,47 - 0,6)|.|18,2494|$
- $|E_2(0,47)| \approx \mathbf{8,303.10^{-3}}$

Grau do polinômio interpolador

- A tabela de *Diferenças Divididas* (ou *Diferenças Ordinárias*) pode nos ajudar na escolha do grau do polinômio interpolador
- Uma vez montada a tabela, examina-se a vizinhança do ponto de interesse: se as *Diferenças* de ordem k forem praticamente constantes (ou seja, se as *Diferenças* de ordem k+1 forem quase nulas), então o polinômio interpolador de grau k será a melhor aproximação para a função nessa região
- Vide um dos exemplos anteriores...

CCI-22

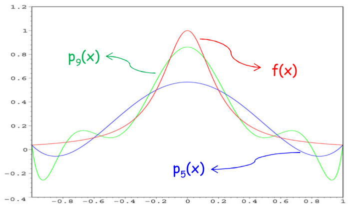
- Introdução
- Forma de Lagrange
- Forma de Newton
- Forma de Newton-Gregory
- Interpolação inversa
- Estudo do erro
- Convergência
- Funções *splines*

Convergência

- Sejam o intervalo [a,b] coberto pelos pontos $a=x_0, x_1, ..., x_n=b$, o valor da função f nesses pontos e $p_n(x)$ o polinômio interpolador de f(x)
- Uma questão importante:
  - Vale sempre a pena utilizar o polinômio interpolador de grau máximo?
  - Em outras palavras, à medida que aumenta o número de pontos de interpolação, ou seja, quando $n \rightarrow \infty$, $p_n(x)$ sempre converge para f(x) nesse intervalo?
- Teorema: Para qualquer sequência de pontos de interpolação $a=x_0, x_1, ..., x_n=b$ no intervalo [a,b], existe uma função contínua f(x) tal que $p_n(x)$ *não converge* para f(x) quando $n \rightarrow \infty$

Fenômeno de Runge

- No caso em que os pontos de interpolação são igualmente espaçados, essa divergência pode ser ilustrada através de um caso conhecido como *Fenômeno de Runge*
- Seja, por exemplo, $f(x) = 1/(1+25x^2)$ tabelada no intervalo [-1;1] nos pontos $x_i = -1 + 2i/n$, $0 \leq i \leq n$
- Veja abaixo f(x) com duas interpolações polinomiais:



À medida que aumenta o número de pontos de interpolação, $|f(x) - p_n(x)|$ torna-se arbitrariamente grande nesse intervalo

Alternativas

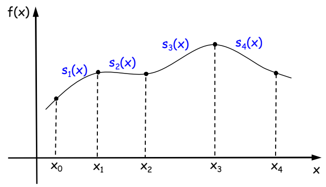
- Em casos como esse, há três alternativas:
  - Não aproximar f(x) através de polinômios, mas com outro tipo de funções
  - Trocar a aproximação em pontos igualmente espaçados pela aproximação em nós de Chebyshev, que distribui o erro mais homogeneamente: $x_i = (x_0 + x_n)/2 + (x_n - x_0)\xi_i/2$, $0 \leq i \leq n$, onde $\xi_i = \cos((2i+1)\pi/(2n+2))$
  - Usar funções *splines*, com convergência garantida

CCI-22

- Introdução
- Forma de Lagrange
- Forma de Newton
- Forma de Newton-Gregory
- Interpolação inversa
- Estudo do erro
- Convergência
- Funções *splines*

Funções *splines*

- *Splines* são hastes flexíveis (de plástico ou de madeira), fixadas em certos pontos de uma mesa de desenho, para traçar curvas suaves
- A ideia deste método é interpolar a função em grupos de poucos pontos (geralmente, dois a dois), e ao mesmo tempo impor condições para que a aproximação e suas derivadas (até certa ordem) sejam contínuas. Desse modo, serão obtidos polinômios de grau menor



Veremos apenas as *splines* cúbicas, isto é, formadas por polinômios de grau 3

Splines cúbicas

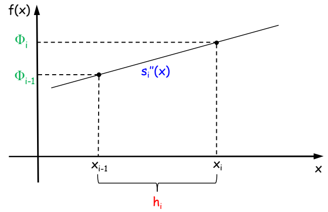
- Dados n+1 pontos distintos $a=x_0, x_1, \ldots, x_n=b$ e seus respectivos valores $f(x_0) = y_0$, $f(x_1) = y_1$, ..., $f(x_n) = y_n$, a função *spline* s será formada por n polinômios cúbicos $s_i(x)$, $0<i\leq n$, com as seguintes propriedades:
  - $s(x) = s_i(x),\ x \in [x_{i-1},x_i],\ 0<i\leq n$
  - $s(x_i) = y_i,\ 0\leq i\leq n$
  - $s_i(x_i) = s_{i+1}(x_i),\ 0<i<n$

  Coincidem nos pontos de interpolação

  - $s_i'(x_i) = s_{i+1}'(x_i),\ 0<i<n$
  - $s_i''(x_i) = s_{i+1}''(x_i),\ 0<i<n$

  Garantem que a curva s não tenha picos, nem troque a curvatura nos nós
- A partir dessas condições, é possível deduzir os 4 coeficientes de cada um dos n polinômios cúbicos $s_i(x)$

Cálculo da *spline*



As derivadas segundas são retas

Exigência da *spline*:
$s_i''(x_i) = s_{i+1}''(x_i) = \Phi_i$

Equação da reta:
$s_i''(x) = \Phi_{i-1} + (\Phi_i - \Phi_{i-1})(x - x_{i-1})/h_i$

- Desenvolvendo a equação da reta:
  - $s_i''(x) = \Phi_{i-1} + (\Phi_i x - \Phi_i x_{i-1} - \Phi_{i-1}x + \Phi_{i-1}x_{i-1})/h_i$
  - $s_i''(x) = (\Phi_{i-1}(x_i - x_{i-1}) + (\Phi_i x - \Phi_i x_{i-1} - \Phi_{i-1}x + \Phi_{i-1}x_{i-1}))/h_i$
  - $s_i''(x) = \Phi_{i-1}(x_i - x)/h_i + \Phi_i(x - x_{i-1})/h_i$
- Integrando:
  - $s_i'(x) = \Phi_{i-1}(x_i - x)^2/2h_i + c_i + \Phi_i(x - x_{i-1})^2/2h_i - d_i$

Cálculo da *spline*

- $s_i'(x) = \Phi_{i-1}(x_i - x)^2/2h_i + c_i + \Phi_i(x - x_{i-1})^2/2h_i - d_i$
- Integrando novamente:

  Sem perda de generalidade
- $s_i(x) = \Phi_{i-1}(x_i - x)^3/6h_i + \Phi_i(x - x_{i-1})^3/6h_i + c_i(x - x_{i-1}) + d_i(x_i - x)$
- Substituindo x por x_{i-1}:
- $s_i(x_{i-1}) = \Phi_{i-1}h_i^2/6 + d_i h_i$
- Sabemos que $s_i(x_{i-1}) = y_{i-1}$:
- $d_i = y_{i-1}/h_i - h_i\Phi_{i-1}/6$
- Idem para x_i:
- $s_i(x_i) = \Phi_i h_i^2/6 + c_i h_i$
- Sabemos que $s_i(x_i) = y_i$:
- $c_i = y_i/h_i - h_i\Phi_i/6$
- Substituindo c_i e d_i na fórmula de $s_i(x)$:
- $s_i(x) = \Phi_{i-1}(x_i - x)^3/6h_i + \Phi_i(x - x_{i-1})^3/6h_i + (y_i/h_i - h_i\Phi_i/6)(x - x_{i-1}) + (y_{i-1}/h_i - h_i\Phi_{i-1}/6)(x_i - x)$

Cálculo da *spline*

- $s_i(x) = \Phi_{i-1}(x_i - x)^3/6h_i + \Phi_i(x - x_{i-1})^3/6h_i + (y_i/h_i - h_i\Phi_i/6)(x - x_{i-1}) + (y_{i-1}/h_i - h_i\Phi_{i-1}/6)(x_i - x)$
- Derivando $s_i(x)$:
  - $s_i'(x) = -\Phi_{i-1}(x_i - x)^2/2h_i + \Phi_i(x - x_{i-1})^2/2h_i + y_i/h_i - h_i\Phi_i/6 - y_{i-1}/h_i + h_i\Phi_{i-1}/6$
- Substituindo x por x_i:
- $s_i'(x_i) = h_i\Phi_i/3 + y_i/h_i - y_{i-1}/h_i + h_i\Phi_{i-1}/6$
- Calculemos também $s_{i+1}'(x_i)$:
- $s_{i+1}'(x) = -\Phi_i(x_{i+1} - x)^2/2h_{i+1} + \Phi_{i+1}(x - x_i)^2/2h_{i+1} + y_{i+1}/h_{i+1} - h_{i+1}\Phi_{i+1}/6 - y_i/h_{i+1} + h_{i+1}\Phi_i/6$
- $s_{i+1}'(x_i) = -h_{i+1}\Phi_i/3 + y_{i+1}/h_{i+1} - h_{i+1}\Phi_{i+1}/6 - y_i/h_{i+1}$
- Sabemos que $s_i'(x_i) = s_{i+1}'(x_i)$:
- $h_i\Phi_i/3 + y_i/h_i + h_i\Phi_{i-1}/6 - y_{i-1}/h_i = -h_{i+1}\Phi_i/3 + y_{i+1}/h_{i+1} - h_{i+1}\Phi_{i+1}/6 - y_i/h_{i+1}$
- $h_i\Phi_{i-1} + 2(h_i + h_{i+1})\Phi_i + h_{i+1}\Phi_{i+1} = 6((y_{i+1} - y_i)/h_{i+1} - (y_i - y_{i-1})/h_i)$

Cálculo da *spline*

- $h_i\Phi_{i-1} + 2(h_i + h_{i+1})\Phi_i + h_{i+1}\Phi_{i+1} = 6((y_{i+1} - y_i)/h_{i+1} - (y_i - y_{i-1})/h_i)$
- Essa igualdade é válida para $0<i<n$:
  - n-1 equações
  - n+1 incógnitas: $\Phi_0, \ldots, \Phi_n$
- Basta acrescentar duas condições: $\Phi_0 = 0$ e $\Phi_n = 0$
- Sistema tridiagonal de equações de ordem n-1:

$$\begin{bmatrix} 2(h_1+h_2) & h_2 & & & & \\ h_2 & 2(h_2+h_3) & h_3 & & & \\ & h_3 & 2(h_3+h_4) & h_4 & & \\ & & & \ddots & & \\ & & & h_{n-3} & 2(h_{n-3}+h_{n-2}) & h_{n-2} \\ & & & & h_{n-2} & 2(h_{n-2}+h_{n-1}) \end{bmatrix} . \begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \vdots \\ \Phi_{n-2} \\ \Phi_{n-1} \end{bmatrix} = \begin{bmatrix} b_1 - b_0 \\ b_2 - b_1 \\ b_3 - b_2 \\ \vdots \\ b_{n-2} - b_{n-3} \\ b_{n-1} - b_{n-2} \end{bmatrix}$$

onde $b_i = 6(y_{i+1} - y_i)/h_{i+1}$, $0 \le i < n$

Exemplo

- Deseja-se obter f(0,25) através de *spline* cúbica:

x	0	0,5	1,0	1,5	2,0
f(x)	3	1,8616	-0,5571	-4,1987	-9,0536

- Será preciso determinar $s_1(x)$, $s_2(x)$, $s_3(x)$ e $s_4(x)$
- Nesse exemplo, $h_i = 0,5$, para $0<i\le4$
- Sistema tridiagonal:

$$\begin{bmatrix} 2 & 0,5 & 0 \\ 0,5 & 2 & 0,5 \\ 0 & 0,5 & 2 \end{bmatrix} . \begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \end{bmatrix} = \begin{bmatrix} -15,3636 \\ -14,6748 \\ -14,5598 \end{bmatrix} \Rightarrow \begin{array}{l} \Phi_3 = -6,252 \\ \Phi_2 = -4,111 \\ \Phi_1 = -6,654 \end{array}$$

- Como se deseja obter uma aproximação para f(0,25), basta calcular $s_1(0,25)$:
  - $s_i(x) = \Phi_{i-1}(x_i - x)^3/6h_i + \Phi_i(x - x_{i-1})^3/6h_i + (y_i/h_i - h_i\Phi_i/6)(x - x_{i-1}) + (y_{i-1}/h_i - h_i\Phi_{i-1}/6)(x_i - x)$
  - $s_1(0,25) = \Phi_1(0,25 - 0)^3/6.0,5 + (y_1/0,5 - 0,5\Phi_1/6)(0,25 - 0) + (y_0/0,5)(0,5 - 0,25)$
  - $s_1(0,25) = 2,5348 \approx f(0,25)$

Implementação

- No cálculo da *spline* cúbica, é preciso resolver um sistema linear tridiagonal
- De modo análogo à Eliminação de Gauss, é possível triangularizar este sistema de uma forma muito eficiente
- Vejamos um caso particular (n=5):

$$\begin{bmatrix} 2(h_1+h_2) & h_2 & 0 \\ h_2 & 2(h_2+h_3) & h_3 \\ 0 & h_3 & 2(h_3+h_4) \end{bmatrix} = \begin{bmatrix} c_1 & h_2 & 0 \\ h_2 & c_2 & h_3 \\ 0 & h_3 & c_3 \end{bmatrix} \Leftarrow \quad L_2 = L_2 - (h_2/c_1).L_1$$

$$\begin{bmatrix} c_1 & h_2 & 0 \\ 0 & c'_2 & h_3 \\ 0 & h_3 & c_3 \end{bmatrix} \Leftarrow \quad L_3 = L_3 - (h_3/c'_2).L_2$$

$$\begin{bmatrix} c_1 & h_2 & 0 \\ 0 & c'_2 & h_3 \\ 0 & 0 & c'_3 \end{bmatrix}$$

Algoritmo

- Cálculo de Φ_i, $0<i<n$:
  - $c_1 = 2(h_1 + h_2)$
  - $c_i = 2(h_i + h_{i+1}) - (h_i)^2/c_{i-1}$, para $1<i<n$
  - $d_1 = b_1 - b_0$
  - $d_i = (b_i - b_{i-1}) - (h_i d_{i-1})/c_{i-1}$, para $1<i<n$
  - $\Phi_{n-1} = d_{n-1}/c_{n-1}$
  - $\Phi_i = (d_i - h_{i+1}\Phi_{i+1})/c_i$, para $n-1>i>0$

Tempo: O(n)

- Uma vez obtidos os valores de Φ_i, $0<i<n$, o cálculo de s(x) também pode ser realizado em tempo O(n)