A FIRST COURSE IN NUMERICAL METHODS

CAPÍTULO 11 - INTERPOLAÇÃO POR PARTES

- métodos robustos para a interpolação de funções que funcionam até quando o número de pontos é grande, ou quando as localizações das abcissas não estão em nosso controle, ou quando o intervalo da função aproximada é grande.

SEÇÃO 11.1 - O CASO PARA INTERPOLAÇÃO POR PARTES

- consideramos a interpolação de $n+1$ pontos

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

que procura uma função que satisfaça

$$v(x_i) = y_i, \quad i = 0, 1, \dots, n$$

- também continuamos considerando uma função subjacente $f(x)$ que deve ser aproximada em um intervalo $[a,b]$ contendo a abcissa x_i.
- a função $f(x)$ é desconhecida, exceto por seus valores $f(x_i) = y_i$, $i = 0, 1, \dots, n$.
- apenas consideramos interpolantes em forma linear que podem ser escritos como:

$$v(x) = \sum_{j=0}^{n} c_j \phi_j(x)$$

onde $\phi_j(x)$ são funções bases dadas e c_j são coeficientes desconhecidos a serem determinados.

INSUFICIÊNCIAS DA INTERPOLAÇÃO POLINOMIAL

Um interpolante da forma discutida no cáp 10 nem sempre é adequado pelos seguintes motivos:

- O erro $f(x) - p_n(x)$ pode não ser pequeno.
- Polinômios de ordem grande tendem a oscilar "irracionalmente"
- As derivadas de ordem grande podem explodir, o que produz um erro grande.
- "Sem localidade": alterar qualquer valor de dados pode alterar drasticamente todo o interpolador.

INTERPOLAÇÃO POR PARTES

- devemos encontrar uma maneira de reduzir o termo do erro sem aumentar o grau n, fazemos isso diminuindo o tamanho do intervalo $b-a$. Assim, recorremos ao uso de partes do polinômio apenas localmente.
- Globalmente, usamos a interpolação polinomial por partes. Assim, dividimos o intervalo em vários subintervalos menores (ou elementos) pela partição

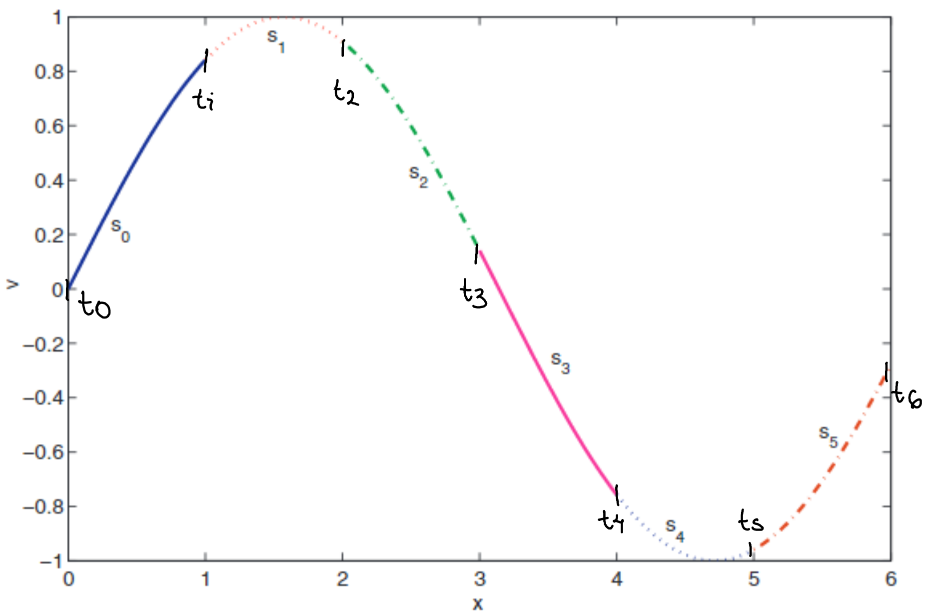
$$a = t_0 < t_1 < \dots < t_r = b$$

e usamos uma interpolação polinomial (grau relativamente baixo) em cada um desses subintervalos $[t_i, t_{i+1}]$, $i = 0, \dots, r-1$.

- Essas partes do polinômio, $s_i(x)$, são então remendadas para formar uma curva $v(x)$ de interpolação global contínua (ou C^1 ou C^2) que satisfaz

$$v(x) = s_i(x), \quad t_i \leq x \leq t_{i+1}, \quad i = 0, \dots, r-1$$

- Os pontos $t_0, t_1, \dots, t_r$ são chamados de break points



SEÇÃO 11.2 - INTERPOLAÇÃO BROKEN LINE (LINEAR) E HERMITE POR PARTES

INTERPOLAÇÃO LINEAR (BROKEN LINE)

O exemplo mais simples de interpolação polinomial contínua por partes é linear por partes. Assim, as partes polinomiais são lineares e o interpolador linear por partes é contínuo (mas não continuamente diferenciável) em todos os lugares.

- EXEMPLO 11.1.

Um exemplo de interpolação linear é fornecido pela figura:

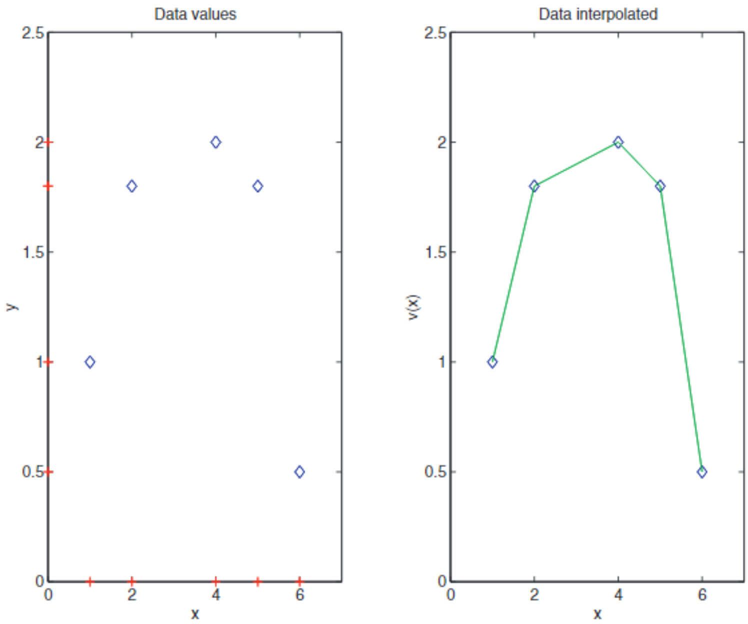


Figure 11.2. *Data and their broken line interpolation.*

Consiste simplesmente em conectar valores de dados por linhas retas. Pela fórmula de Newton para um interpolador polinomial linear, podemos escrever:

$$\boxed{v(x) = s_i(x) = f(x_i) + f[x_i, x_{i+1}](x - x_i), \quad x_i \le x \le x_{i+1}, \quad 0 \le i \le 4}$$

Uma grande vantagem da interpolação linear por partes, além de sua óbvia simplicidade, é que os valores de máximo e mínimo estão nos pontos: nenhum novo ponto extremo é "inventado" pelo interpolante, o que é importante para uma rotina black-box de interpolação de propósito geral.

O MATLAB usa essa interpolação como a opção padrão para plotagem.

LIMITE DE ERRO (ERROR BOUND) PARA INTERPOLAÇÃO POR PARTES LINEAR

Continuando com a notação do exemplo 11.1, seja $n = r$, $t_i = x_i$ e

$$h = \max_{1 \leq i \leq r} (t_i - t_{i-1})$$

não é difícil mostrar que o erro para interpolação linear é limitado por

$$|f(x) - v(x)| \leq \frac{h^2}{8} \max_{a \leq \xi \leq b} |f''(\xi)|$$

para qualquer x no intervalo $[a,b]$. De fato, para qualquer $x \in [a,b]$ existe um índice i, $1 \leq i \leq r$ tal que $t_{i-1} \leq x \leq t_i$. O interpolador é linear neste subintervalo, então aplicando a fórmula de erro da seção 10.5 para interpolação polinomial a este segmento linear, temos

$$f(x) - v(x) = \frac{f''(\xi)}{2!}(x - t_{i-1})(x - t_i)$$

$$\boxed{|f(x) - v(x)| = \frac{f''(\xi)}{2!}(x - t_{i-1})(x - t_i) \leq \frac{h^2}{8} \max_{a \leq \xi \leq b} |f''(\xi)|}$$

Assim, pelo menos para pontos de dados igualmente espaçados, à medida que n aumenta o erro diminui, ā taxa $O(h^2) = O(n^{-2})$

Observe que n não é mais o grau das partes do polinômio.

INTERPOLAÇAO POR PARTES CONSTANTE

Frequentemente, os break points t_i são os dados pontos da abcissa x_i, ordenados em ordem crescente, caso em que temos $r = n$. Tal é o caso da interpolação linear por partes.

Uma exceção simples é a interpolação constante por partes: não sabemos nada sobre a "suavidade" de f – nem mesmo se for contínua – podemos querer construir a aproximação mais simples, e uma dessas é dada por:

$$\boxed{v(x) = s_i(x) = f(x_i), \quad t_i \leq x \leq t+1}$$

no qual $t_i = \frac{1}{2}(x_i + x_{i+1})$, $i = 1, 2, \dots, n$

$t_0 = a \leq x_0$

$t_r = b \leq x_n$

$r = n + 1$

INTERPOLAÇÃO POR PARTES CÚBICA

Mas a interpolação linear por partes geralmente não é suave o suficiente. (Com certeza, o que é "suficiente" depende da aplicação) Da figura 11.2 fica claro que esse interpolante tem primeiras derivadas descontínuas nos pontos dados.

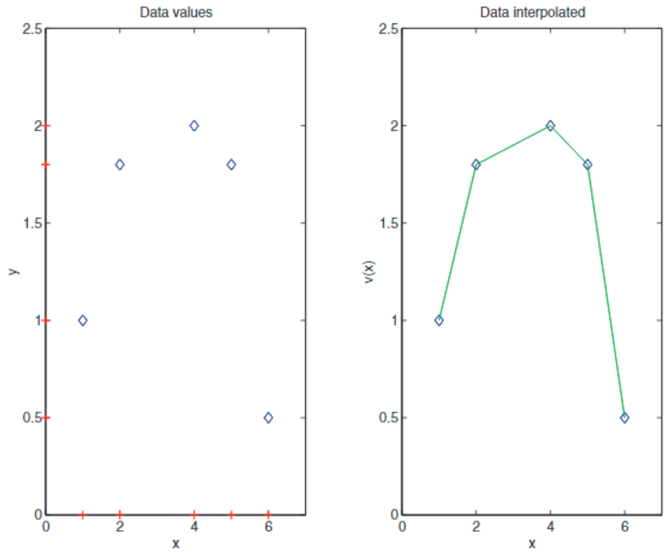


Figure 11.2. *Data and their broken line interpolation.*

Para ter mais suavidade, digamos, C^1 ou C^2, devemos aumentar o grau de cada peça polinomial.

A interpolação polinomial por partes mais popular é com cúbicos. Aqui podemos escrever:

$$v(x) = s_i(x) = a_i + b_i(x-t_i) + c_i(x-t_i)^2 + d_i(x-t_i)^3,$$

$$t_i \leq x \leq t_{i+1}, \quad i = 0, \dots, r-1$$

Claramente, nós temos $4r$ incógnitas (coeficientes). Para fixar esses parâmetros, precisamos de 4 condições algébricas. Existem dois tipos de condições:

- condições de interpolação
- condições de continuidade

No caso linear por partes, essas condições são lidas:

$$s_i(t_i) = f(t_i), \quad s_i(t_{i+1}) = f(t_{i+1}), \quad i = 0, 1, \dots, r-1$$

no qual $t_i = x_i$, $i = 0, 1, \dots, r$. Isso fornece $2r$ condições, que esgotam toda a liberdade que temos no caso linear por partes porque temos apenas $2r$ coeficientes para determinar. A continuidade está implícita em $s_i(t_{i+1}) = f(t_{i+1}) = s_{i+1}(t_{i+1})$. Mas no caso cúbico por partes podemos impor $2r$ condições adicionais. Agora, procuramos uma maneira de fazer isso, que mantém o "charme" da construção local.

Outra maneira importante de definir essas condições é considerada na seção 11.3

INTERPOLAÇÃO DE HERMITE CÚBICA POR PARTES

O caso mais simples, ou mais limpo, é onde também são fornecidos valores de $f'(t_i)$. Assim, $n+1 = 2(r+1)$ e as abcissas são:

$$(x_0, x_1, x_2, \dots, x_{n-1}, x_n) = (t_0, t_0, t_1, t_1, \dots, t_r, t_r)$$

A correspondência destes fornece precisamente mais $2r$ condições, escritas como:

$$s_i'(t_i) = f'(t_i), \quad s_i'(t_{i+1}) = f'(t_{i+1}), \qquad i = 0, 1, \dots, r-1$$

Esta é a interpolação de Hermite cúbica por partes. Observe que $v(x)$ está claramente em C^1, ou seja, tanto v quanto v' são contínuos em todos os lugares.

Esse interpolador pode ser construído localmente, parte por parte, aplicando o algoritmo de interpolação osculante da Seção 10.7 para o polinômio cúbico em cada subintervalo separadamente.

Especificamente, para qualquer x no subintervalo $[t_i, t_{i+1}]$ o interpolador $v(x)$ coincide com o correspondente polinômio cúbico de Hermite $s_i(x)$ neste intervalo, dado explicitamente por:

$$\boxed{\begin{aligned} s_i(x) = {} & f_i + (h_i f_i')\tau + (3(f_{i+1} - f_i) - h_i(f_{i+1}' + 2f_i'))\tau^2 \\ & + (h_i(f_{i+1}' + f_i') - 2(f_{i+1} - f_i))\tau^3 \end{aligned}}$$

no qual $h_i = t_{i+1} - t_i$, $f_i = f(t_i)$, $f_i' = f'(t_i)$, $\tau = \dfrac{x - t_i}{h_i}$

Observe também que alterar os dados em um ponto de dados altera o valor do interpolante apenas nos dois subintervalos associados a este ponto.

ERROR BOUND PARA INTERPOLAÇÃO HERMITE CÚBICA POR PARTES

TEOREMA: Erro da Interpolação Polinomial por Partes

Seja $v(x)$ o interpolador de $f(x)$ nos $n+1$ pontos $x_0 < x_1 < \dots < x_n$. Defina $h = \max_{1 \le i \le n} (x_i - x_{i-1})$ e assuma que $f(x)$ tenha quantas derivadas forem necessárias no intervalo $[a, b]$ que contém x_i, $i = 0, \dots, n$.

Então usando uma interpolação local linear, constante ou Hermite cúbica, para cada $x \in [a,b]$ o error bound da interpolação é:

- $|f(x) - v(x)| \le \frac{h}{2} \max_{a \le \xi \le b} |f'(\xi)|$ constante por partes
- $|f(x) - v(x)| \le \frac{h^2}{8} \max_{a \le \xi \le b} |f''(\xi)|$ linear por partes
- $|f(x) - v(x)| \le \frac{h^4}{384} \max_{a \le \xi \le b} |f''''(\xi)|$ Hermite cúbica por partes

SEÇÃO 11.3 - SPLINE CÚBICA

A principal desvantagem de trabalhar com Hermite cúbica por partes é que precisamos de valores para $f'(t_i)$, o que é pedir muito, especialmente quando não há f, apenas valores de dados discretos. Outra desvantagem potencial é que existem muitas aplicações em que um requisito geral C^1 não fornece suavidade suficiente.

Suponha que temos apenas valores de dados (x_i, y_i), $i = 0, \dots, n$, onde $x_0 < x_1 < \dots < x_{n-1} < x_n$ são distintos e $y_i = f(x_i)$ para alguma função f que pode não estar explicitamente disponível. Defina também $a = x_0$ e $b = x_n$.

Identificamos x_i com os breakpoints t_i e $n = r$. Tendo usado $2n$ parâmetros para satisfazer as condições de interpolação por um interpolador contínuo, agora usamos os $2n$ parâmetros restantes para exigir que $v(x) \in C^2[a, b]$. O resultado é muitas vezes referido como uma spline cúbica.

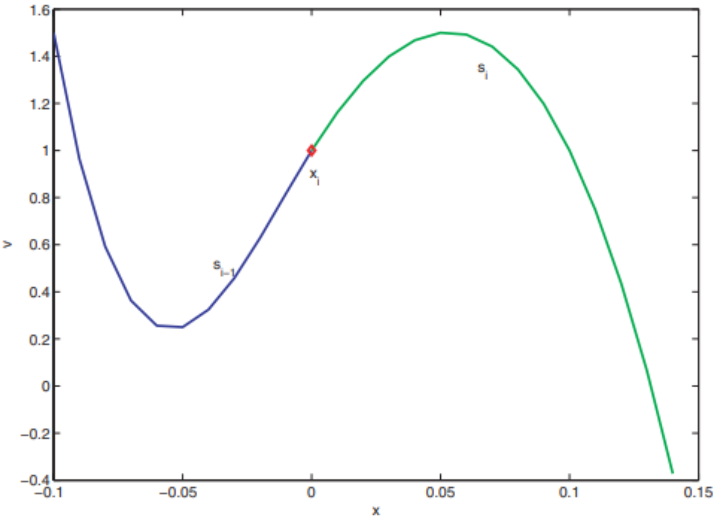
Assim, as condições a serem satisfeitas pela spline cúbica são:

$s_i(x_i) = f(x_i)$,	$i = 0, \dots, n-1$	11.1a
$s_i(x_{i+1}) = f(x_{i+1})$,	$i = 0, \dots, n-1$	11.1b
$s_i'(x_{i+1}) = s_{i+1}'(x_{i+1})$,	$i = 0, \dots, n-2$	11.1c
$s_i''(x_{i+1}) = s_{i+1}''(x_{i+1})$,	$i = 0, \dots, n-2$	11.1d

Note que as condições de correspondência são aplicadas apenas nas abscissas internas, não incluindo o primeiro e o último pontos. Por esta razão, existem apenas $n-1$ dessas condições para a primeira e segunda derivadas.

EXEMPLO. considere os dados:

i	0	1	2
x_i	0.0	1.0	2.0
$f(x_i)$	1.1	0.9	2.0



• observe que $n = 2$, assim, temos três pontos de dados e apenas um break point em $x_1 = 1.0$

A spline cúbica interpoladora é escrita como:

$$v(x) = \begin{cases} s_0(x) = a_0 + b_0(x - 0.0) + c_0(x - 0.0)^2 + d_0(x - 0.0)^3, & x < 1.0 \\ s_1(x) = a_1 + b_1(x - 1.0) + c_1(x - 1.0)^2 + d_1(x - 1.0)^3, & x > 1.0 \end{cases}$$

Agora devemos determinar os oito coeficientes $a_0, b_0, c_0, d_0, a_1, b_1, c_1$ e d_1.

As condições de interpolação (11.1a) e (11.b), avaliadas em suas extremidades $i = 0$, dão

$$1.1 = v(0.0) = s_0(0.0) = a_0$$

$$0.9 = v(1.0) = s_1(1.0) = a_1$$

que determinam $a_0 = 1.1$ e $a_1 = 0.9$

Avaliando essas duas condições em $i = 1$, temos também

$$0.9 = v(1.0) = s_0(1.0) = 1.1 + b_0 + c_0 + d_0$$

$$2.0 = v(2.0) = s_1(2.0) = 0.9 + b_1 + c_1 + d_1$$

Assim, temos duas relações entre os seis coeficientes restantes. Observe como igualar $s_0(1.0)$ e $s_1(1.0)$ ao mesmo valor de $f(1.0)$ implica continuidade do interpolante construído $v(x)$.

Em seguida, avaliamos (11.1c) e (11.1d) em $i = 0$ (o único valor de i para o qual eles são definidos aqui), ou seja, especificamos que também $s_0'(x_1) = s_1'(x_1)$ e $s_0''(x_1) = s_1''(x_1)$, onde $x_1 = 1.0$. Isto dá

$$b_0 + 2c_0 + 3d_0 = b_1$$

$$2c_0 + 6d_0 = 2c_1$$

No total, então, temos quatro equações para os seis coeficientes b_0, b_1, c_0, c_1, d_0 e d_1. Exigimos mais duas condições para completar as especificações de $v(x)$.

DUAS CONDIÇÕES ADICIONAIS

No total, acima existem $4n-2$ condições para 4 incógnitas, então ainda temos duas condições para especificar. Especificamos uma em cada um dos limites x_0 e x_n:

1. opção 1: uma escolha popular é a de free boundary ou spline natural

$$v''(x_0) = v''(x_n) = 0$$

Essa condição é um tanto arbitrária, porém, e pode causar deteriorização geral na qualidade da aproximação porque não há razão a priori para supor que f também desaparece nos pontos finais.

2. opção 2: se $f'(x_0)$ e $f'(x_n)$ são conhecidas, podemos impor (clamped boundary):

$$v'(x_0) = f'(x_0), \qquad v'(x_n) = f'(x_n)$$

O interpolador resultante também é conhecido como spline completo.

3. opção 3: not-a-Knot. Na ausência de informações sobre as derivadas de f nas extremidades, usamos os dois parâmetros restantes para garantir a continuidade da terceira derivada do interpolador spline nos break points internos mais próximos, x_1 e x_{n-1}.

Ao contrário das condições free boundary, a not-a-Knot não requer que o interpolador satisfaça algo que a função interpolada não satisfaça, portanto, a qualidade do erro não se deteriora.

$$v_0'''(x_1) = v_1'''(x_1)$$

$$v_{n-2}'''(x_{n-1}) = v_{n-1}'''(x_{n-1})$$

PROPRIEDADES

Interpolant	Local?	Order	Smooth?	Selling features
Piecewise constant	yes	1	bounded	Accommodates general f
Broken line	yes	2	C^0	Simple, max and min at data values
Piecewise cubic Hermite	yes	4	C^1	Elegant and accurate
Spline (not-a-knot)	not quite	4	C^2	Accurate, smooth, requires only f data

CAPÍTULO 14 - DIFERENCIAÇÃO NUMÉRICA

SEÇÃO 14.1 - FÓRMULAS DERIVADAS DE SÉRIES DE TAYLOR

1. Two-points formula

$$\boxed{f'(x_0) = \frac{f(x_0) - f(x_0 - h)}{h} + \frac{h}{2} f''(\xi)}, \qquad x_0 - h \leq \xi \leq x_0$$

2. Three point formulas

a) centered formula

$$\boxed{f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{h} - \frac{h^2}{6} f'''(\xi)}, \qquad x_0 - h \leq \xi \leq x_0 + h$$

b) higher order one sided formula

$$\boxed{f'(x_0) = \frac{1}{2h}(-3f(x_0) + 4f(x_0 + h) - f(x_0 + 2h)) + \frac{h^2}{3} f'''(\xi)}, \quad x_0 \leq \xi \leq x_0 + 2h$$

3. Five point formula

$$\boxed{f'(x_0) \approx \frac{1}{12h}(f(x_0 - 2h) - 8f(x_0 - h) + 8f(x_0 + h) - f(x_0 + 2h))}$$

erro: $e(h) = \frac{h^4}{30} f^{(v)}(\xi)$

4. Three point formula for the second derivative

$$\boxed{f''(x_0) = \frac{1}{h^2}(f(x_0 - h) - 2f(x_0) + f(x_0 + h)) - \frac{h^2}{12} f^{(iv)}(\xi)}, \quad x_0 - h \leq \xi \leq x_0 + h$$

SEÇÃO 11.2 EXTRAPOLAÇÃO DE RICHARDSON

A extrapolação de Richardson é um mecanismo simples e eficaz para gerar métodos numéricos de ordem superior a partir de métodos de ordem inferior. Dados dois métodos da mesma ordem, podemos explorar a relação entre seus termos de erro principais para eliminar tal termo.

$$f''(x_0) = \frac{1}{12h^2}\left(-f(x_0-2h) + 16f(x_0-h) - 30f(x_0) + 16f(x_0+h) - f(x_0+2h)\right)$$

$$+ \frac{h^4}{90} f^{(VI)}(x_0) + O(h^5)$$

erro: $e(h) = \frac{h^4}{90} f^{(VI)}(\xi)$, $x_0 - 2h \le \xi \le x_0 + 2h$

SEÇÃO 11.3 - FÓRMULAS DERIVADAS DO POLINÔMIO INTERPOLADOR DE LAGRANGE

- esperamos obter fórmulas em termos de valores de $f(x_i)$

Considere a derivação de uma fórmula envolvendo os pontos $x_0, x_1, \dots, x_n$, não ordenados de nenhuma maneira particular. O polinômio interpolador de grau no máximo n é:

$$p(x) = \sum_{j=0}^{n} f(x_j)\, L_j(x)$$

no qual o polinômio de Lagrange é

$$L_j(x) = \frac{(x-x_0)\dots(x-x_{j-1})(x-x_{j+1})\dots(x-x_n)}{(x_j-x_0)\dots(x_j-x_{j-1})(x_j-x_{j+1})\dots(x_j-x_n)}$$

derivando de $p(x)$ e substituindo $x = x_0$, obtém-se a fórmula geral para diferenciação:

$$p'(x_0) = \sum_{j=0}^{n} f(x_j)\, L_j'(x_0)$$

essa fórmula não requer que os pontos $x_0, x_1, \dots, x_n$ sejam equidistantes.

DIFERENCIAÇÃO USANDO PONTOS EQUIDISTANTES

Para obter expressões mais limpas, vamos assumir um espaçamento equidistante. Assim, suponha que os pontos x_i sejam distribuídos em torno de x_0 e dados como $x_0 - lh, x_0 - (l-1)h, \dots, x_0 - h, x_0, x_0 + h, \dots, x_0 + uh$, onde l e u são inteiros não negativos e $n = l + u$. Há uma mudança óbvia no índice envolvido, de 0 para $-l$, porque queremos enfatizar que os pontos $x_i = x_0 + ih$ estão geralmente em ambos os lados de x_0, onde buscamos aproximar $f'(x_0)$.

$$L_j'(x_0) = \frac{1}{jh} \prod_{\substack{k=-l \\ k \neq 0 \\ k \neq j}}^{u} \left(\frac{-k}{j-k}\right), \text{ para } j \neq 0 \qquad a_j = hL_j'(x_0), \quad j = -l, \dots, u$$

$$\boxed{p'(x_0) = h^{-1} \sum_{j=-l}^{u} a_j f(x_j)}$$

erro: $f'(x_0) - p_n'(x_0) = \left[\frac{f^{(n+1)}(\xi)}{(n+1)!}\, l!\, u!\right] h^n$

para $x_{-l} \leq \xi \leq x_u$

CAPÍTULO 15 - INTEGRAÇÃO NUMÉRICA

SEÇÃO 15.1 - REGRAS BÁSICAS DE QUADRATURA

As regras básicas de quadratura são baseadas na interpolação polinomial de baixo grau. Dada uma função $f(x)$ em um intervalo curto $[a,b]$, podemos escolher um conjunto de nós $x_0, x_1, \dots, x_n \in [a,b]$ e construir um interpolador polinomial $p_n(x)$. Disto segue-se que

$$\int_a^b p_n(x)\,dx \quad \text{aproxima} \quad \int_a^b f(x)\,dx$$

REGRAS BÁSICAS

Suponha que as abcissas $x_0, x_1, \dots, x_n$ tenham sido especificadas de alguma forma. Então o polinômio interpolador na forma de Lagrange é:

$$p_n(x) = \sum_{j=0}^{n} f(x_j)\, L_j(x)$$

onde,

$$L_j(x) = \frac{(x-x_0)\cdots(x-x_{j-1})(x-x_{j+1})\cdots(x-x_n)}{(x_j-x_0)\cdots(x_j-x_{j-1})(x_j-x_{j+1})\cdots(x_j-x_n)} = \prod_{\substack{k=0\\k\neq j}}^{n} \frac{(x-x_k)}{(x_j-x_k)}$$

isso leva a

$$\int_a^b f(x)\,dx \approx \int_a^b p_n(x)\,dx = \int_a^b \sum_{j=0}^{n} f(x_j)\, L_j(x)\,dx = \sum_{j=0}^{n} f(x_j) \int_a^b L_j(x)\,dx$$

portanto

$$a_j = \int_a^b L_j(x)\,dx$$

REGRA DO TRAPÉZIO

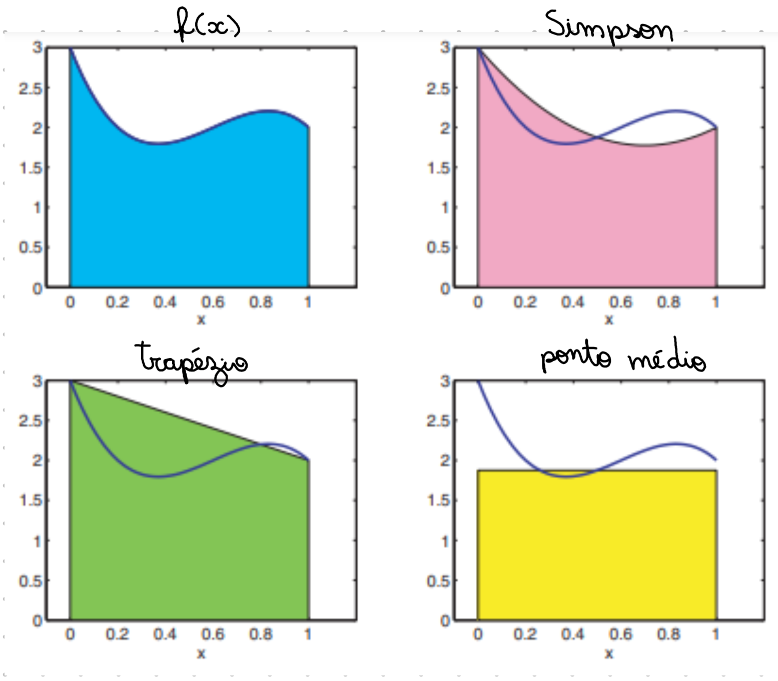
$$I_f \approx I_{trap} = \frac{b-a}{2}\left[f(a) + f(b)\right]$$

REGRA DE SIMPSON

$$I_f \approx I_{simp} = \frac{b-a}{6}\left[f(a) + 4f\left(\frac{b+a}{2}\right) + f(b)\right]$$

REGRA DO PONTO MÉDIO

$$I_f \approx I_{mid} = (b-a)\, f\left(\frac{a+b}{2}\right)$$



Quadrature	Rule	Error
Midpoint	$(b-a)f(\frac{a+b}{2})$	$\frac{f''(\xi_1)}{24}(b-a)^3$
Trapezoidal	$\frac{b-a}{2}[f(a)+f(b)]$	$-\frac{f''(\xi_2)}{12}(b-a)^3$
Simpson	$\frac{b-a}{6}[f(a)+4f(\frac{b+a}{2})+f(b)]$	$-\frac{f''''(\xi_3)}{90}(\frac{b-a}{2})^5$

SEÇÃO 15.2 - INTEGRAÇÃO NUMÉRICA COMPOSTA

Composite Quadrature Methods.
With $rh = b-a$, where r is a positive integer (must be even in the Simpson case), we have the formulas

$$\begin{aligned} \int_a^b f(x)dx &\approx \frac{h}{2}\left[f(a)+2\sum_{i=1}^{r-1} f(a+ih)+f(b)\right], \quad \text{trapezoidal} \\ &\approx \frac{h}{3}\left[f(a)+2\sum_{k=1}^{r/2-1} f(t_{2k})+4\sum_{k=1}^{r/2} f(t_{2k-1})+f(b)\right], \quad \text{Simpson} \\ &\approx h\sum_{i=1}^{r} f(a+(i-1/2)h), \quad \text{midpoint.} \end{aligned}$$

Theorem: Quadrature Errors.
Let f be sufficiently smooth on $[a,b]$, and consider a composite method using a mesh $a = t_0 < t_1 < \cdots < t_r = b$ with $h_i = t_i - t_{i-1}$. Denote $h = \max_{1\le i\le r} h_i$. In the case of the Simpson method assume that $h_{i+1} = h_i$ for all i odd.

Then the error in the composite trapezoidal method satisfies

$$|E(f)| \le \frac{\|f''\|_\infty}{12}(b-a)h^2,$$

the error in the composite midpoint method satisfies

$$|E(f)| \le \frac{\|f''\|_\infty}{24}(b-a)h^2,$$

and the error in the composite Simpson method satisfies

$$|E(f)| \le \frac{\|f''''\|_\infty}{180}(b-a)h^4.$$