Approssimazioni di dati e funzioni

In molti problemi applicativi e nella costruzione stessa di alcuni metodi numerici di base emerge l'esigenza di dover approssimare una funzione.
Approssimare una funzione vuol dire sostituire a una funzione “complicata” una funzione semplice, “facilmente calcolabile”, scelta in uno spazio di dimensione finita e quindi esprimibile come combinazione di un numero finito di funzioni che sia facile calcolare, derivare, integrare,..., mediante algoritmi robusti ed efficienti.

Esaminiamo due casi in cui una funzione appare “complicata”:

1. una funzione può essere nota solo per punti, perché deriva da misurazioni sperimentali oppure perché è soluzione numerica di un problema matematico (funzione empirica). In altre parole, non si ha una espressione analitica della funzione e si deve risolvere un problema di rappresentazione di dati.

Approssimazioni di dati e funzioni

Ad esempio, se si vuole studiare la solubilità del nitrato di sodio $NaNO_3$ rispetto alla temperatura dell'acqua, si può effettuare un esperimento da cui si ottiene una tabella che descrive il comportamento del nitrato di sodio in funzione della temperatura dell'acqua. Effettuando m misurazioni del fenomeno si costruisce una **tabella** con due colonne, nella prima delle quali si riportano i punti di osservazione, ossia le temperature dell'acqua scelte per rilevare il fenomeno, e nella seconda le parti di nitrato di sodio disciolto in 100 parti di acqua alla temperatura assegnata, ossia le osservazioni.

Temperatura dell'acqua (punti di osservazione) x	Parti di $NaNO_3$ disciolte in 100 parti d'acqua (osservazioni) y
0°	66.7
4°	71.0
10°	76.3
15°	80.6
21°	85.7
29°	92.9
36°	99.4

Approssimazioni di dati e funzioni

Se si assume che esista una legge quantitativa che lega i punti di osservazione e le osservazioni, si può descrivere tale relazione quantitativa con

$$y = f(x)$$

dove

- x è la variabile indipendente che assume i valori x_i corrispondenti ai punti di osservazione ed esprime la variazione di temperatura;
- y è la variabile dipendente, che corrisponde alle osservazioni, ossia ai valori osservati di parti di $NaNO_3$ disciolti in acqua.

Si dice che y è funzione di x. Si vuole costruire un modello matematico che descriva sufficientemente bene il fenomeno e consenta di fare delle previsioni per valori di x diversi dai punti di osservazione (**problemi di simulazione numerica**).

La scelta del modello è condizionata da considerazioni legate al problema da risolvere, ma anche da considerazioni numeriche dovute al fatto che la funzione deve essere facilmente calcolabile.

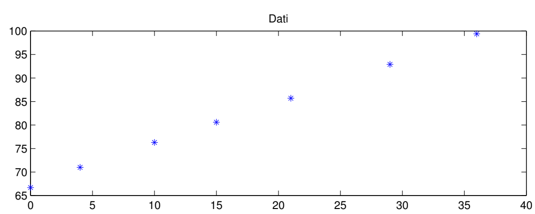
Approssimazioni di dati e funzioni

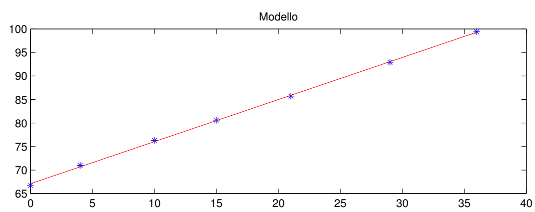
In questo caso il fenomeno può essere descritto con sufficiente accuratezza da una funzione lineare

$$y = ax + b$$

con $a = 0.89$ e $b = 67.10$. In tal modo si ottiene:

Temperatura dell'acqua x	Parti di $NaNO_3$ disciolte in 100 parti d'acqua y
0°	66.7
4°	71.0
10°	76.3
15°	80.6
21°	85.7
29°	92.9
36°	99.4





2. nel secondo caso si suppone di dover operare su una funzione $f(x)$ nota analiticamente, ma con una espressione che è difficile da calcolare in un qualsiasi x appartenente al dominio di definizione, o per cui è difficile calcolare l'integrale o la derivata con i soli strumenti dell'Analisi. Allora si "sostituisce" $f(x)$ con una funzione più semplice $f_n(x)$ appartenente a uno spazio di dimensione finita, su cui è possibile operare analiticamente e dedurre un'approssimazione del risultato richiesto entro i limiti di una tolleranza prefissata.

   Questo tipo di approssimazione è richiesto anche per scrivere routine di calcolo che valutino una funzione nei punti del suo dominio di definizione, oppure la sua derivata o il suo integrale (librerie scientifiche e compilatori).

   Esempio: calcolo di $\displaystyle\int_{-1}^{1} \frac{e^{-x}}{\sqrt{1-x^2}}\,dx$.

In questo ambito dell'Analisi Numerica, si individuano due grandi capitoli:

- **approssimazione**: tecniche e metodi per il trattamento di dati con errore;
- **interpolazione**: tecniche e metodi per il trattamento dei dati che si assume siano esatti.

Il **problema lineare** affrontato dai metodi di interpolazione è il seguente:

- in uno spazio funzionale lineare $\mathcal{F}$ di dimensione finita (costituito da *funzioni facilmente calcolabili*), generato dalle funzioni di base $\phi_0(\boldsymbol{x}), \ldots, \phi_n(\boldsymbol{x})$, per cui ogni elemento $f(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ si esprime come combinazione lineare della base:
$$f(\boldsymbol{x}) = a_0\phi_0(\boldsymbol{x}) + a_1\phi_1(\boldsymbol{x}) + \ldots + a_n\phi_n(\boldsymbol{x})$$
dove $a_0, \ldots, a_n$ sono le incognite e $\boldsymbol{x} \in \mathbb{R}^d$,
- assegnato un insieme di dati $(\boldsymbol{x}_i, y_i)$, $i = 0, \ldots, n$

si vuole determinare l'elemento $f(x) \in \mathcal{F}$ che soddisfa le **condizioni di interpolazione**:
$$f(\boldsymbol{x}_i) = y_i \qquad i = 0, \ldots, n$$

Si tratta di determinare $a_0, \ldots, a_n$.

I punti $\boldsymbol{x}_0, \ldots, \boldsymbol{x}_n$ si dicono **punti di collocazione**, mentre $a_0, \ldots, a_n$ si dicono **gradi di libertà**. In generale $\boldsymbol{x}_i \in \mathbb{R}^d$ con $d \geqslant 1$.

Approssimazione di dati e funzioni

Nel linguaggio della Geometria Analitica in $\mathbb{R}^2$ ciò vuol dire: *assegnati* $n+1$ *punti* $(\boldsymbol{x}_i, y_i) \in \mathbb{R}^{n+1}$, $i = 0, \dots, n$ *nel piano cartesiano, si determini la curva di equazione* $y = f(\boldsymbol{x})$ *che* ***onora*** *tutti i punti assegnati*.

Il numero dei punti deve coincidere con il numero dei gradi di libertà.

Si possono richiedere anche condizioni sulle derivate di f in opportuni punti.

Si assume che gli y_i siano accurati (non affetti da errore).

A seconda dello spazio funzionale $\mathcal{F}$ scelto, ci sono diversi tipi di interpolazione lineare (interpolazione polinomiale, polinomiale a tratti, trigonometrica, . . .).

Considereremo il caso $d = 1$, cioè $x_i \in \mathbb{R}$.

Interpolazione polinomiale

Polinomio di interpolazione di Lagrange

Assegnati $n+1$ punti di osservazione distinti $x_0, x_1, \dots, x_n \in [a, b]$ chiuso e limitato ed $n+1$ osservazioni $y_0, y_1, \dots, y_n \in \mathbb{R}$, si vuole determinare il **polinomio** di grado al più n tale che (condizioni di interpolazione):

$$p_n(x_i) = y_i \qquad i = 0, \dots, n$$

dove $p_n(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n$ (*rappresentazione di un polinomio nella base standard delle potenze di* x*: forma canonica di un polinomio*).
I valori $y_0, \dots, y_n$ possono essere considerati come i valori noti in $n+1$ punti distinti di una funzione $F(x)$ definita nell'intervallo $[a, b]$, che si vuole interpolare in tale dominio:

$$p_n(x_i) = F(x_i) = y_i \qquad i = 0, \dots, n$$

Nel linguaggio della Geometria Analitica, si vuole determinare la **curva algebrica** di equazione $y = p_n(x)$ che "onora" gli $n+1$ punti distinti (x_i, y_i), $i = 0, \dots, n$, del piano cartesiano.

Interpolazione polinomiale

Usando la base delle potenze, si tratta di determinare $a_0, \dots, a_n$, risolvendo un sistema di $n+1$ equazioni in $n+1$ incognite, ottenuto dalle condizioni di interpolazione (**metodo dei coefficienti indeterminati**):

$$\begin{cases} a_0 + a_1 x_0 + a_2 x_0^2 + \ldots + a_n x_0^n = y_0 \\ a_0 + a_1 x_1 + a_2 x_1^2 + \ldots + a_n x_1^n = y_1 \\ \vdots \\ a_0 + a_1 x_n + a_2 x_n^2 + \ldots + a_n x_n^n = y_n \end{cases}$$

La matrice dei coefficienti è la **matrice di Vandermonde**: è non singolare se e solo se gli x_i sono distinti

$$V = \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{pmatrix}$$

con $\det(V) = \Pi_{i>j}(x_i - x_j)$. Se gli x_i sono distinti, allora esiste una e una sola soluzione ($\det(V) \neq 0$), ossia esiste uno e un solo polinomio di interpolazione di grado al più n che onora tutte le osservazioni ($a_0, \dots, a_n$ sono unici).

Interpolazione polinomiale

Tuttavia **la matrice di Vandermonde è mal condizionata**.

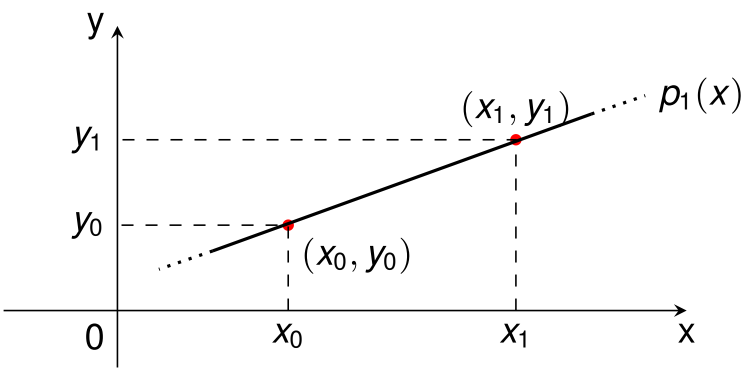
Per esempio se si prendono 10 punti equispaziati nell'intervallo $[0, 1]$, la matrice di Vandermonde di ordine 10 ha numero di condizione

$$\|V\|_\infty \|V^{-1}\|_\infty \approx 4.81 \cdot 10^7$$

Pertanto, si calcola il polinomio di interpolazione di grado al più n relativo a $n+1$ punti distinti usando una rappresentazione del polinomio diversa da quella canonica, che consenta anche di risparmiare in termini di complessità computazionale. Una di queste rappresentazioni è il **polinomio di Lagrange**.

Esempio

Dati (x_0, y_0), (x_1, y_1) distinti, si vuole calcolare il polinomio di grado 1 tale che $p_1(x_i) = y_i$, $i = 0, 1$, ossia, geometricamente, l'equazione della retta passante per i punti (x_0, y_0), (x_1, y_1) del piano cartesiano:



$$p_1(x) = \frac{x - x_1}{x_0 - x_1} y_0 + \frac{x - x_0}{x_1 - x_0} y_1$$

Vale $p_1(x_0) = y_0$, $p_1(x_1) = y_1$.
Se si pongono $L_0(x) = \dfrac{x - x_1}{x_0 - x_1}$ ed $L_1(x) = \dfrac{x - x_0}{x_1 - x_0}$, si ha

$$\begin{array}{ll} L_0(x_0) = 1 & L_1(x_0) = 0 \\ L_0(x_1) = 0 & L_1(x_1) = 1 \end{array}$$

dunque

$$p_1(x) = L_0(x) y_0 + L_1(x) y_1$$

Derivazione del polinomio di Lagrange

Generalizziamo il caso $n = 1$ appena visto nell'esempio ad un qualunque intero $n > 1$. Nel caso generale si deve costruire per ogni punto k-esimo, con $k = 0, \ldots, n$, un polinomio $L_k(x)$ di grado n tale che

$$\begin{cases} L_k(x_i) = 0 & \text{per } i \neq k \\ L_k(x_k) = 1 \end{cases}$$

Allora $x_0, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n$ sono zeri di $L_k(x)$:

$$L_k(x) = \alpha (x - x_0) \cdots (x - x_{k-1})(x - x_{k+1}) \cdots (x - x_n)$$

e, poiché $L_k(x_k) = 1$, deve essere

$$\alpha = \frac{1}{(x_k - x_0) \cdots (x_k - x_{k-1})(x_k - x_{k+1}) \cdots (x_k - x_n)}$$

Ne segue che l'espressione di $L_k(x)$ è

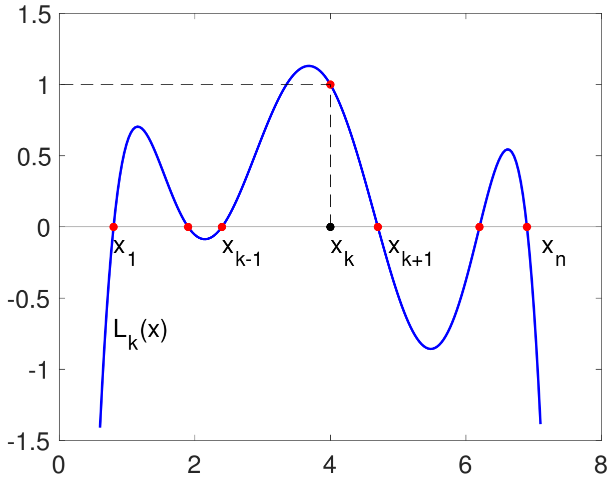
$$L_k(x) = \frac{(x - x_0) \cdots (x - x_{k-1})(x - x_{k+1}) \cdots (x - x_n)}{(x_k - x_0) \cdots (x_k - x_{k-1})(x_k - x_{k+1}) \cdots (x_k - x_n)} = \prod_{\substack{i=0,\ldots,n \\ i \neq k}} \frac{x - x_i}{x_k - x_i}$$

Polinomio di interpolazione di Lagrange

Allora il polinomio di interpolazione (polinomio di Lagrange) è esprimibile come

$$p_n(x) = y_0 L_0(x) + y_1 L_1(x) + \ldots + y_n L_n(x) = \sum_{k=0}^{n} y_k L_k(x)$$

con $L_k(x) = \dfrac{(x-x_0)(x-x_1)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_n)}{(x_k-x_0)(x_k-x_1)\cdots(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_n)}$ (base di Lagrange).



Esistenza e unicità del polinomio di interpolazione

Teorema

Assegnati $n+1$ punti distinti $x_0, \ldots, x_n \in [a, b]$, chiuso e limitato, e $n+1$ valori $y_0, \ldots, y_n \in \mathbb{R}$, esiste uno e un solo polinomio di grado al più n tale che $p_n(x_i) = y_i$, $\forall i = 0, \ldots, n$. Esso si può esprimere come **polinomio di Lagrange** e vale che $L_0(x) + L_1(x) + \ldots + L_n(x) = 1$.

Dim.

- Il polinomio di Lagrange è combinazione lineare di polinomi di grado n, $L_k(x)$, e dunque è un polinomio di grado al più n.
- Poiché $L_k(x_i) = 0$ per $i \neq k$ e $L_k(x_k) = 1$, si ha che il polinomio di Lagrange è polinomio di interpolazione ossia:

$$p_n(x_i) = y_0 L_0(x_i) + \ldots + y_i L_i(x_i) + \ldots + y_n L_n(x_i) = 1 \cdot y_i = y_i \qquad \forall i = 0, \ldots, n$$

- Il polinomio di Lagrange è l'unico polinomio di interpolazione. Infatti, supposto che esista un altro polinomio $g_n(x)$ di grado al più n tale che $g_n(x_i) = y_i$ $\forall i = 0, \ldots, n$, allora $p_n(x) - g_n(x)$ è un polinomio di grado al più n che si annulla in $n+1$ valori x_i, $i = 0, \ldots, n$; per il teorema fondamentale dell'algebra $p_n(x) - g_n(x) \equiv 0$. Questo implica, per il principio di indentità dei polinomi, che $g_n(x)$ e $p_n(x)$ coincidono.

Esistenza e unicità del polinomio di interpolazione

Per l'unicità del polinomio di interpolazione, il polinomio di interpolazione di grado n di un polinomio di grado minore o uguale a n è esso stesso.

Per dimostrare che $1 = \sum_{i=0}^{n} L_i(x)$ basta osservare che, se $g(x)$ è un polinomio di grado al più n tale che $g(x_i) = y_i$, $\forall i = 0, \ldots, n$, segue che $g(x) = p_n(x)$ $\forall x \in [a, b]$.
Se $g(x) = 1$ e $g(x_i) = 1 = y_i$, $i = 0, \ldots, n$, allora, per ogni $x \in [a, b]$, vale che

$$1 = p_n(x) = 1 \cdot L_0(x) + \ldots + 1 \cdot L_n(x)$$

$$1 = \sum_{i=0}^{n} L_i(x)$$

Esistenza e unicità del polinomio di interpolazione

In pratica il polinomio di Lagrange si esprime come combinazione lineare di una diversa base dell'insieme dei polinomi di grado n: $L_0(x), L_1(x), \ldots, L_n(x)$ sono una **base** per l'insieme dei polinomi di grado n (detta base di Lagrange).

Poiché l'insieme dei polinomi di grado n è uno spazio vettoriale di dimensione $n+1$, basta trovare $n+1$ polinomi linearmente indipendenti che generano lo spazio per trovare una base.
Dimostriamo che $L_i(x)$, $i = 0, \ldots, n$ formano una base dimostrando che essi sono linearmente indipendenti.
Considerata una combinazione lineare di $L_i(x)$ e posta tale combinazione uguale alla funzione nulla, si ha

$$\sum_{i=0}^{n} \alpha_i L_i(x) = 0 \qquad \text{per ogni } x \in [a, b]$$

In particolare, per x_j, $j = 0, \ldots, n$, si ha

$$\sum_{i=0}^{n} \alpha_i L_i(x_j) = \alpha_j L_j(x_j) = \alpha_j = 0$$

Pertanto $\{L_i(x)\}_{i=0,\ldots,n}$ è una base.

Esistenza e unicità del polinomio di interpolazione

Se si rappresenta il polinomio interpolante come $p_n(x) = \sum_{j=0}^{n} a_j L_j(x)$, ossia come combinazione lineare della base di Lagrange, imponendo le condizioni di interpolazione $p_n(x_i) = y_i$, $i = 0, \ldots, n$, il problema dell'interpolazione lineare si esprime mediante il seguente sistema di equazioni lineari

$$\sum_{j=0}^{n} a_j L_j(x_i) = y_i \qquad i = 0, \ldots, n,$$

dove la matrice dei coefficienti è l'identità (ben condizionata)

$$\begin{pmatrix} L_0(x_0) & L_1(x_0) & \ldots & L_n(x_0) \\ \vdots & \vdots & & \vdots \\ L_0(x_n) & L_1(x_n) & \ldots & L_n(x_n) \end{pmatrix} \begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix}$$

e $a_j = y_j$, $j = 0, \ldots, n$.

Esempio

Dati $x_0 = 2$, $x_1 = 2.5$, $x_2 = 4$, si vuole trovare il polinomio di interpolazione di secondo grado di $f(x) = 1/x \quad \Rightarrow \quad y_0 = 1/x_0 = 0.5$,, $y_1 = 1/x_1 = 0.4$, $y_2 = 1/x_2 = 0.25$.
Dal punto di vista geometrico, $p_2(x)$ è una parabola.

$$L_0(x) = \frac{(x-2.5)(x-4)}{(2-2.5)(2-4)} = x^2 - 6.5x + 10$$

$$L_1(x) = \frac{(x-2)(x-4)}{(2.5-2)(2.5-4)} = (-4x^2 + 24x - 32)/3$$

$$L_2(x) = \frac{(x-2)(x-2.5)}{(4-2)(4-2.5)} = (x^2 - 4.5x + 5)/3$$

$$\begin{aligned} p_2(x) &= 0.5(x^2 - 6.5x + 10) + 0.4(-4x^2 + 24x - 32)/3 + 0.25(x^2 - 4.5x + 5)/3 \\ &= 0.05x^2 - 0.425x + 1.15 \end{aligned}$$

I coefficienti di $p_2(x)$ si possono ottenere anche risolvendo il sistema:

$$\begin{pmatrix} 1 & 2 & 4 \\ 1 & 2.5 & 6.25 \\ 1 & 4 & 16 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} 0.5 \\ 0.4 \\ 0.25 \end{pmatrix}$$

$$p_2(3) = 0.05 \cdot 9 - 0.425 \cdot 3 + 1.15 = 0.325 \quad \text{invece di } 1/3 = 0.3333\ldots = 0.\overline{3}$$

Si è commesso un errore detto **errore di interpolazione**

Esempio

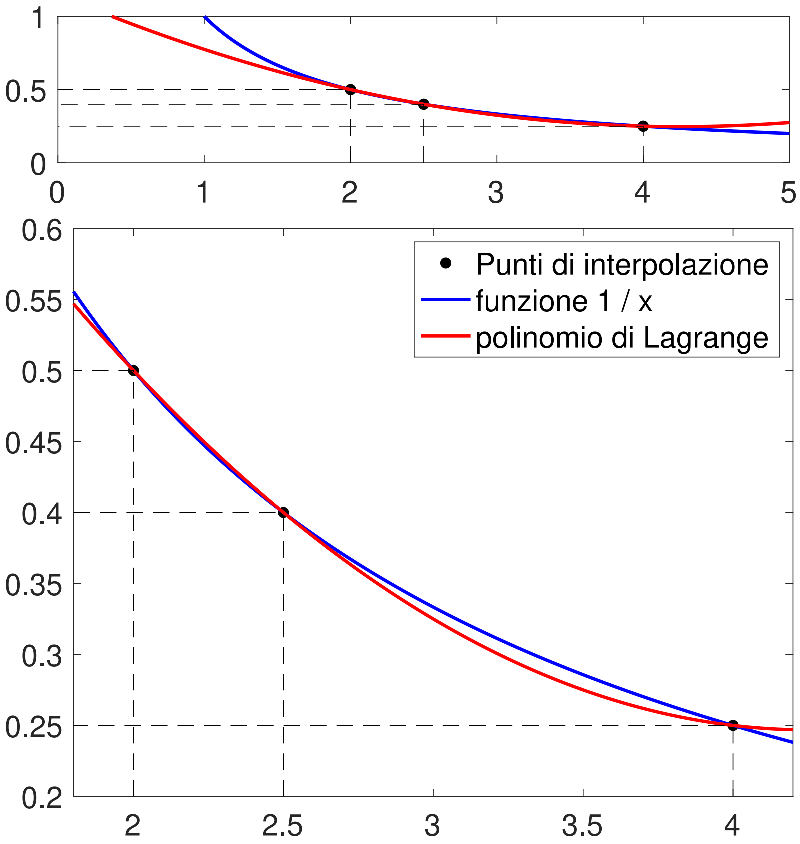


Grafico in proporzione (stessa scala su ascisse e ordinate)

Zoom in proporzioni scalate per evidenziare l'errore di interpolazione

Calcolo del polinomio di Lagrange

Alcuni termini del polinomio di Lagrange

$$p_n(x) = y_0 L_0(x) + y_1 L_1(x) + \ldots + y_n L_n(x)$$

con

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

non dipendono da x e si possono calcolare una volta per tutte; sono dati da:

$$\text{coeff}_k = \frac{y_k}{(x_k - x_0)(x_k - x_1)\cdots(x_k - x_{k-1})(x_k - x_{k+1})\cdots(x_k - x_n)} \qquad k = 0, \ldots, n$$

La complessità computazionale per il loro calcolo è $\mathcal{O}(n^2)$.

Calcolo del polinomio di Lagrange

```
function [p, coeff] = polyLagrange(x, y, punti)
% POLYLAGRANGE - Polinomio interpolante nella forma di Lagrange
% INPUT
%   x     (double array) - vettore dei nodi o punti di osservazione
%   y     (double array) - vettore delle osservazioni
%   punti (double array) - vettore dei punti in cui calcolare il
%                          polinomio di Lagrange
% OUTPUT
%   p     (double array) - valore del polinomio nei punti
%   coeff (double array) - coefficienti del polinomio di Lagrange
  n1 = numel(y); coeff = zeros(size(x)); p = zeros(size(punti));
  for k = 1 : n1
    coeff(k) = y(k) / prod( x(k) - x([1:k-1, k+1:n1]) );
  end
  for j = 1 : numel(punti)
    ij = find( punti(j) == x );
    if isempty(ij)
      % calcolo del polinomio di Lagrange
      p(j) = prod( punti(j) - x ) * sum( coeff ./ (punti(j) - x) );
    else
      p(j) = y( ij(1) );
    end
  end
end
```


Errore di interpolazione

Generalmente (a meno di non interpolare un polinomio di grado non superiore a n), il polinomio $p_n(x)$ assume valori diversi dalla funzione $f(x)$ che si vuole interpolare con il polinomio di Lagrange, ossia tale che $p_n(x_i) = f(x_i) = y_i$, $i = 0, \dots, n$.
Si dice **errore di interpolazione** la funzione

$$R(x) = f(x) - p_n(x) \qquad x \in [a, b]$$

che è tale che $R(x_i) = 0$, $i = 0, \dots, n$, ma è $R(x) \neq 0$, per $x \neq x_i$.
Se non si sa nulla della $f(x)$, non si può dire nulla su $R(x)$. Ci sono **infinite** funzioni $f(x)$, tali che $f(x_i) = y_i$ e sono tutte interpolate dal medesimo polinomio.
Se si possiede una conoscenza qualitativa delle derivate di $f(x)$, allora è possibile calcolare $R(x)$, $x \in [a, b]$.

Errore di interpolazione

Teorema

Sia $f(x) \in C^{n+1}([a,b])$. Siano $x_0, \ldots, x_n$ punti distinti in $[a,b]$ e sia $p_n(x)$ il polinomio di grado al più n che interpola $f(x)$ in $[a,b]$. Allora esiste un punto $\xi \in [a,b]$, dipendente da $x, x_0, \ldots, x_n$ e da $f(x)$, tale che

$$R(x) = f(x) - p_n(x) = \frac{(x-x_0)\cdots(x-x_n)}{(n+1)!} f^{(n+1)}(\xi)$$
$$= \frac{\omega_{n+1}(x)}{(n+1)!} f^{(n+1)}(\xi)$$

dove $\omega_{n+1}(x) = (x-x_0)(x-x_1)\cdots(x-x_n)$.

Osservazione

Poiché il punto $\xi \in [a,b]$ dipendente da x, per $x_0, \ldots, x_n$ fissati è incognito, la formula dell'errore di interpolazione ha significato teorico. Tuttavia, poiché $f^{n+1}(x)$ è continua in $[a,b]$ chiuso e limitato, esiste una costante M tale che

$$\left|f^{(n+1)}(x)\right| \leqslant M \qquad x \in [a,b]$$

Pertanto

$$|R(x)| = \left|\frac{f^{(n+1)}(\xi)}{(n+1)!}\omega_{n+1}(x)\right| = \frac{\left|f^{(n+1)}(\xi)\right|}{(n+1)!}|\omega_{n+1}(x)| \leqslant \frac{M}{(n+1)!}|\omega_{n+1}(x)|$$

Inoltre, poiché $\omega_{n+1}(x)$ è continua in $[a,b]$, esiste ω^* tale che

$$|\omega_{n+1}(x)| \leqslant \omega^* \qquad \forall x \in [a,b]$$

Pertanto

$$|R(x)| \leqslant \frac{M\omega^*}{(n+1)!}$$

Assegnata $\epsilon \geqslant \dfrac{M\omega^*}{(n+1)!}$, $p_n(x)$ è una approssimazione di $f(x)$ entro la tolleranza ϵ.

Estrapolazione

Se x non appartiene all'intervallo $[a, b]$, si parla di estrapolazione anziché di interpolazione. In tal caso occorre ampliare $[a, b]$ al più piccolo intervallo che contiene anche x e richiedere che $f(x)$ sia di classe C^{n+1} anche in tale intervallo.

L'errore di estrapolazione è comunque maggiore o uguale a quello di interpolazione. In tal caso infatti $(x - x_i)$, $i = 0, \ldots, n$, può essere molto più grande.

Esempio

Sia $f(x) = \ln(x)$, $[a, b] = [0.4, 0.8]$. Dati $x_0 = 0.4$, $x_1 = 0.5$, $x_2 = 0.7$, $x_3 = 0.8$ e

$$\begin{aligned} y_0 &= \ln(x_0) = -0.916291 \\ y_1 &= \ln(x_1) = -0.693147 \\ y_2 &= \ln(x_2) = -0.356675 \\ y_3 &= \ln(x_3) = -0.223144 \end{aligned}$$

si vuole determinare il polinomio di interpolazione di grado 3 di $\ln(x)$ in $[0.4, 0.8]$:

$$\begin{aligned} L_0(x) &= \frac{(x-0.5)(x-0.7)(x-0.8)}{(0.4-0.5)(0.4-0.7)(0.4-0.8)} \\ L_1(x) &= \frac{(x-0.4)(x-0.7)(x-0.8)}{(0.5-0.4)(0.5-0.7)(0.5-0.8)} \\ L_2(x) &= \frac{(x-0.4)(x-0.5)(x-0.8)}{(0.7-0.4)(0.7-0.5)(0.7-0.8)} \\ L_3(x) &= \frac{(x-0.4)(x-0.5)(x-0.7)}{(0.8-0.4)(0.8-0.5)(0.8-0.7)} \end{aligned}$$

$$p_3(x) = -0.916291 \cdot L_0(x) - 0.693147 \cdot L_1(x) - 0.356675 \cdot L_2(x) - 0.223144 \cdot L_3(x)$$

Esempio

Per $x = 0.6$ si ha

$$p_3(0.6) = -0.509975 \text{ invece di } -0.510826$$
$$R(0.6) = \ln(0.6) - p_3(0.6) = -0.000851$$

Dalla formula dell'errore di interpolazione si può trovare una maggiorazione di $R(0.6)$ se si conosce la derivata quarta di $\ln(x)$, che è $d^4(\ln(x))/dx^4 = -6/x^4$.

$$R(x) = \frac{(x-0.4)(x-0.5)(x-0.7)(x-0.8)}{4!}\left(-\frac{6}{\xi^4}\right)$$

con $\xi \in]0.4, 0.8[$ dipendente da x. Poiché $6/x^4$ è decrescente in tale intervallo con valore massimo in 0.4, si ha

$$\left|-\frac{6}{x^4}\right| \leqslant \frac{6}{(0.4)^4} = 234.4 \quad \Rightarrow \quad \left|R(x)\right| \leqslant \left|(x-0.4)(x-0.5)(x-0.7)(x-0.8)\right| \cdot \frac{234.4}{24}$$

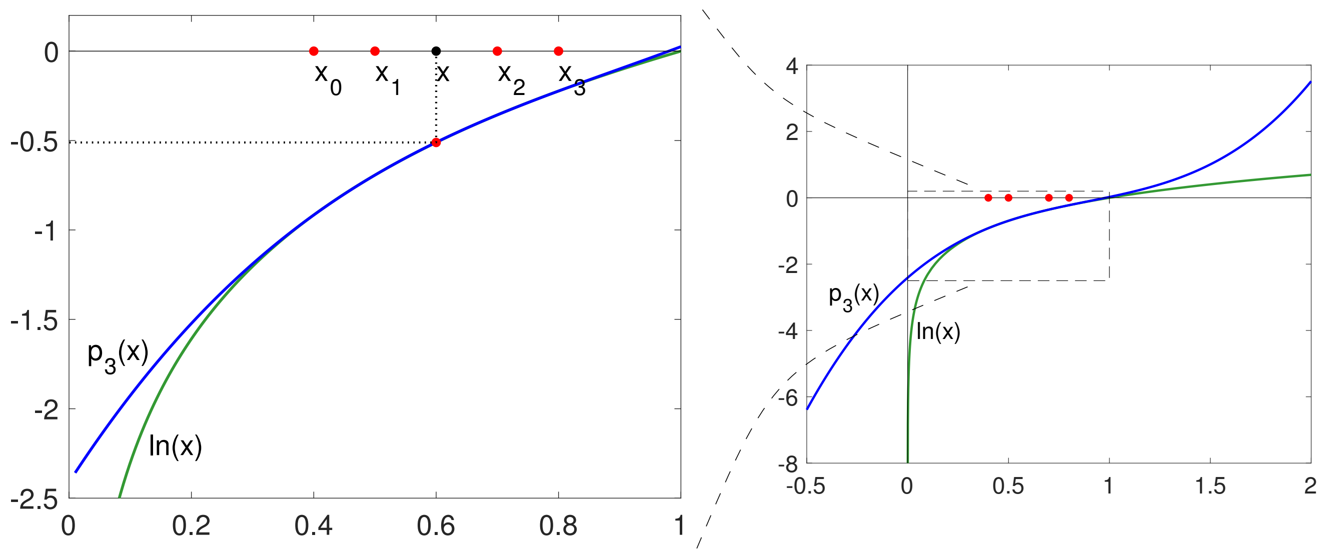
Per $x = 0.6$ si ha

$$\left|R(0.6)\right| \leqslant 0.0039$$

che è una sovrastima di 0.000851.

Esempio

Anche graficamente si nota che il polinomio $p_3(x)$ è una buona approssimazione di $f(x) = \ln(x)$ nell'intervallo $[x_0, x_3]$, ma fuori da tale intervallo le due curve si discostano molto e l'approssimazione diventa man mano peggiore.



Esempio

Sia $f(x)$ una funzione di classe C^2. Se si considera il polinomio di interpolazione $p_1(x)$ di $f(x)$ tale che $p_1(x_0) = f(x_0)$, $p_1(x_1) = f(x_1)$, $x_0 < x_1$, quali sono i valori ammissibili della tolleranza ϵ per poter considerare $p_1(x)$ una approssimazione di $f(x)$ in $[x_0, x_1]$?

$$R(x) = \frac{(x - x_0)(x - x_1)}{2} f''(\xi) \quad \xi \in (x_0, x_1)$$

Sia M tale che $|f''(x)| \leqslant M$, $x \in [x_0, x_1]$ e sia $\omega^* = \max_{x\in[x_0,x_1]} |(x - x_0)(x - x_1)|$. Deve essere

$$\epsilon \geqslant \frac{\omega^* M}{2}$$

$(x - x_0)(x - x_1)$ è una parabola che interseca l'asse x in x_0, x_1 e ha vertice in $\dfrac{x_0 + x_1}{2}$. In tale punto la parabola assume valore $-(x_0 - x_1)^2/4$. Allora $\omega^* = \dfrac{(x_0 - x_1)^2}{4}$. Pertanto

$$\epsilon \geqslant \frac{(x_1 - x_0)^2 M}{8}.$$

Polinomi di Chebyshev

Sono polinomi di grado crescente $T_0(x), T_1(x), \ldots, T_n(x)$ definiti nell'intervallo $[-1, 1]$ nel seguente modo:

$$T_n(x) = \cos\big(n \arccos(x)\big)$$

per ogni $x \in [-1, 1]$. Allora:

$$T_0(x) = 1$$
$$T_1(x) = x$$

Inoltre, posto $\theta = \arccos(x)$ (e dunque $\cos(\theta) = x$), $T_n\big(x(\theta)\big) = \cos(n\theta)$,

$$\begin{aligned} T_{n+1}(\theta) &= \cos\big((n+1)\theta\big) = \cos(n\theta)\cos(\theta) - \sin(n\theta)\sin(\theta) \\ T_{n-1}(\theta) &= \cos\big((n-1)\theta\big) = \cos(n\theta)\cos(\theta) + \sin(n\theta)\sin(\theta) \\ &\Rightarrow T_{n+1}(\theta) = 2\cos(n\theta)\cos(\theta) - T_{n-1}(\theta) \end{aligned}$$

Pertanto vale la relazione di ricorrenza (tornando alla variabile x):

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$$

Allora:

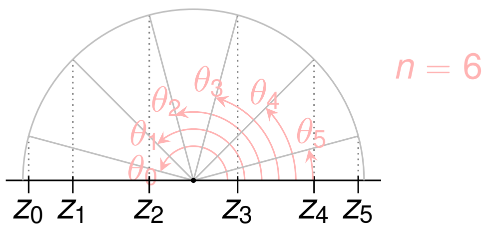
$$T_2(x) = 2x^2 - 1$$
$$T_3(x) = 4x^3 - 3x$$
$$T_4(x) = 8x^4 - 8x^2 + 1$$
$$T_5(x) = 16x^5 - 20x^3 + 5x$$
$$\vdots$$

Si tratta di polinomi di grado crescente tali che

- il coefficiente di x^n in $T_n(x)$ vale 2^{n-1};
- $-1 \leqslant T_n(x) \leqslant 1$ (perché si tratta di una funzione coseno);
- sono funzioni pari per n pari e dispari per n dispari (infatti $T_n(-x) = \cos\big(n \arccos(-x)\big) = \cos\Big(n\big(\pi - \arccos(x)\big)\Big) = \cos(n\pi)\cos\big(n \arccos(x)\big) = (-1)^n T_n(x)$.
  Dunque per n pari $T_n(-x) = T_n(x)$ e per n dispari $T_n(-x) = -T_n(x)$).

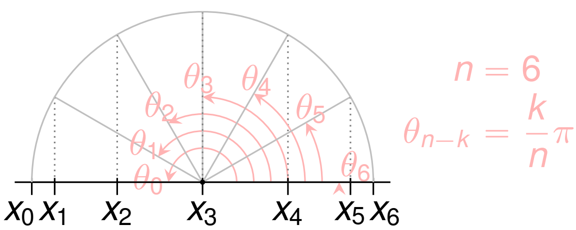
Poiché $T_n(x) = \cos\big(n \arccos(x)\big)$, segue che $T_n(x)$ ha n zeri reali distinti dati da

$$n \arccos(x) = \frac{\pi}{2} + k\pi\,, \quad k = n-1, \dots, 1, 0$$
$$z_{n-k-1} = \cos\left(\frac{2k+1}{2n}\pi\right), \quad k = n-1, \dots, 1, 0$$



$T_n(x)$, essendo un coseno, assume valore massimo 1 e minimo -1 negli $n+1$ punti critici:

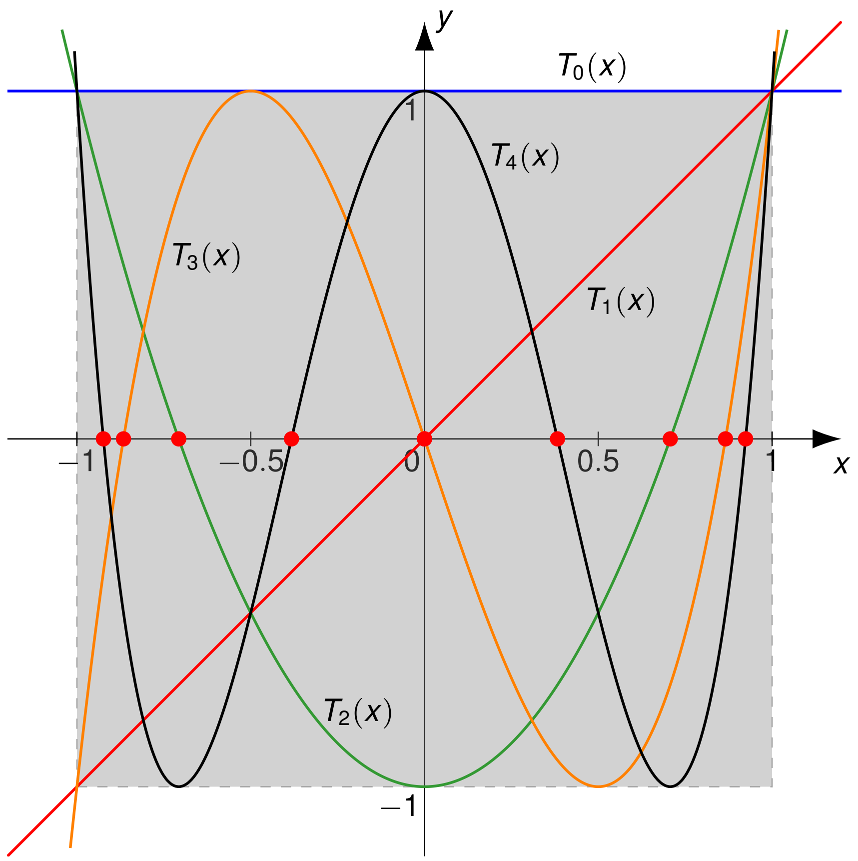
$$n \arccos(x) = k\pi\,, \qquad k = n, n-1, \dots, 1, 0$$
$$x_{n-k} = \cos\left(\frac{k}{n}\pi\right), \qquad k = n, n-1, \dots, 1, 0$$



Vale che $T_n(x_k) = (-1)^k$.

Nelle figure che seguono, i polinomi di grado pari sono indicati con la linea blu, quelli di grado dispari con la linea rossa.

Polinomi di Chebyshev



Polinomi di Chebyshev

In generale per disegnare un polinomio di Chebyshev si disegna una semiconferenza goniometrica (di centro l'origine degli assi e raggio 1) e la si divide in $2n$ archi di uguale lunghezza. Le proiezioni dei $2n+1$ estremi sull'asse x sono, da destra a sinistra, punto di massimo, zero, punto di minimo, zero, ... di $T_n(x)$:

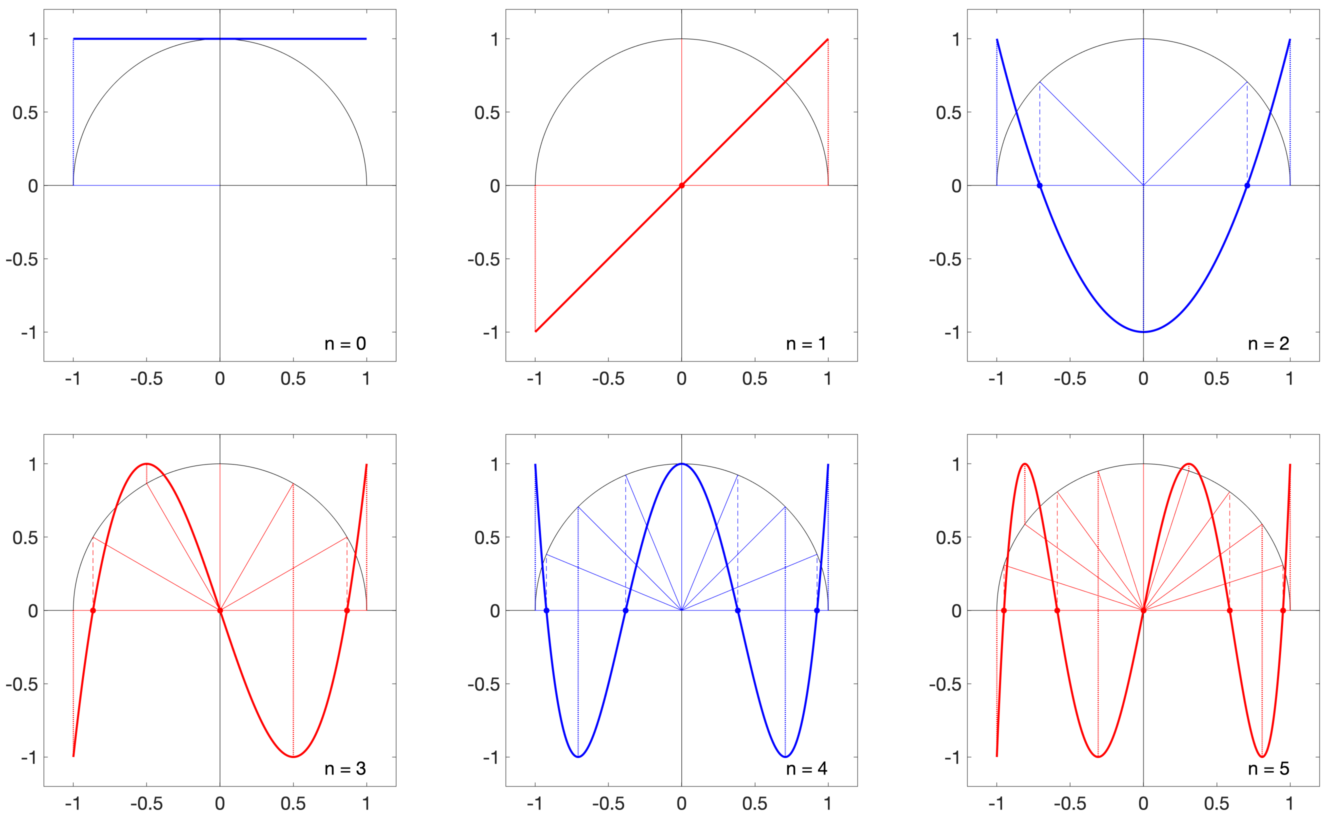
$n = 1$: $x = -1$ punto di minimo, $x = 1$ punto di massimo, $x = 0$ zero.

$n = 2$: $x = \{-1, 1\}$ punti di massimo, $x = 0$ punto di minimo, $x = \left\{\mp\frac{1}{\sqrt{2}}\right\}$ zeri.

$n = 3$: $x = \{-1/2, 1\}$ punti di massimo, $x = \{-1, 1/2\}$ punti di minimo, $x = \left\{\mp\frac{\sqrt{3}}{2}, 0\right\}$ zeri.

Polinomi di Chebyshev

Determinazione geometrica degli zeri dei primi sei polinomi di Chebyshev:



Polinomi di Chebyshev

Polinomi di grado crescente sono linearmente indipendenti. Anche i polinomi di Chebyshev di grado crescente sono linearmente indipendenti.
Poichè **polinomi di grado crescente sono linearmente indipendenti, anche i polinomi di Chebyshev di grado crescente sono linearmente indipendenti**.
Dunque è possibile esprimere un polinomio $p_n(x)$ definito in $[a, b]$ mediante polinomi di Chebyshev usando la seguente mappa:

$$\mu : [a, b] \longrightarrow [-1, 1]$$
$$x \longrightarrow t = \frac{2}{b-a}x - \frac{a+b}{b-a}$$

la cui mappa inversa è

$$\mu^{-1} : [-1, 1] \longrightarrow [a, b]$$
$$t \longrightarrow x = \frac{b-a}{2}t + \frac{a+b}{2}\,.$$

Allora il generico polinomio di grado n si può scrivere come

$$p_n(x) = a_0 T_0(\mu(x)) + a_1 T_1(\mu(x)) + \ldots + a_n T_n(\mu(x))\,.$$

Si tratta comunque di determinare $n+1$ coefficienti.

Proprietà di minimo

I polinomi di Chebyshev soddisfano una importante proprietà di minimo.

Si ricorda che per le funzioni reali continue su un intervallo chiuso e limitato $[a, b]$ si può dare la definizione di norma uniforme (o norma infinito) nel seguente modo:

$$\|f\|_\infty = \max_{x\in[a,b]} |f(x)|$$

Teorema

Tra tutti i polinomi monici (ossia con coefficiente di x^n uguale a 1) di grado n definiti in $[-1, 1]$, il polinomio avente norma infinito minima è $\frac{1}{2^{n-1}} T_n(x)$ e vale che

$$\min_{p_n(x)\in\mathcal{P}_n} \left\|x^n + a_{n-1}x^{n-1} + \ldots + a_0\right\|_\infty$$

$$= \min_{p_n(x)\in\mathcal{P}_n} \max_{x\in[-1,1]} \left|x^n + a_{n-1}x^{n-1} + \ldots + a_0\right|$$

$$= \max_{x\in[-1,1]} \left|\frac{1}{2^{n-1}} T_n(x)\right| = \left\|\frac{1}{2^{n-1}} T_n(x)\right\|_\infty = \frac{1}{2^{n-1}}$$

Nodi di Chebyshev

Supponiamo che ci sia libertà di scegliere i nodi in $[a, b]$ in modo da rendere l'errore di interpolazione minimo possibile. La distribuzione dei nodi $x_0, \ldots, x_n$ che rende minima la quantità

$$\omega^* = \max_{x\in[a,b]} |\omega_{n+1}(x)| = \|\omega_{n+1}(x)\|_\infty$$

corrisponde a trovare un polinomio di grado $n + 1$ per cui in $[a, b]$ è minima la norma del massimo. In tal modo nella maggiorazione dell'errore di interpolazione ω^* ha il valore minimo possibile. Dunque i nodi che rendono ω^* minimo sono i corrispondenti nell'intervallo $[a, b]$ degli zeri di un polinomio di Chebyshev di grado $n + 1$.
Infatti, si consideri la mappa

$$\mu : [a, b] \to [-1, 1]$$

$$x \mapsto \mu(x) = t = \frac{2}{b-a}x - \frac{a+b}{b-a}$$

Nodi di Chebyshev

Presi gli zeri di un polinomio di Chebyshev di grado $n+1$, $t_{n-i} = \cos\left(\frac{(2i+1)\pi}{2(n+1)}\right)$, si ha che

$$x = \frac{b-a}{2}t + \frac{a+b}{2}$$
$$x_i = \frac{b-a}{2}t_i + \frac{a+b}{2} = \frac{b-a}{2}\cos\left(\frac{2i+1}{2(n+1)}\pi\right) + \frac{a+b}{2}$$

per $i = n, n-1, \dots, 1, 0$. Allora

$$x - x_i = \frac{b-a}{2}(t - t_i)$$

Pertanto

$$\omega_{n+1}(x) = (x-x_0)\cdots(x-x_n) = \left(\frac{b-a}{2}\right)^{n+1}(t-t_0)\cdots(t-t_n)$$
$$= \left(\frac{b-a}{2}\right)^{n+1}\frac{1}{2^n}T_{n+1}(t)$$

da cui

$$\|\omega_{n+1}(x)\|_\infty = \left(\frac{b-a}{2}\right)^{n+1}\frac{1}{2^n} = 2\left(\frac{b-a}{4}\right)^{n+1}$$

Esempio

Si vuole trovare il numero di nodi necessari per determinare il polinomio di interpolazione della funzione $f(x) = e^{-x}$ relativo a tali nodi in $[0, 2]$, usando come nodi gli zeri di un opportuno polinomio di Chebyshev, in modo che l'errore sia inferiore a 10^{-3}.
Poiché le derivate della funzione sono date da $(-1)^{n+1}e^{-x}$, si ha che

$$\left|f^{(n+1)}(x)\right| \leqslant 1 \qquad \forall x \in [0, 2]$$

La maggiorazione dell'errore di interpolazione relativa a nodi di Chebyshev fornisce

$$\frac{2}{(n+1)!}\left(\frac{2}{4}\right)^{n+1} \leqslant 10^{-3}$$

Il primo intero n per cui la disequaglianza è verificata è 4. Dunque occorrono 5 nodi che sono gli zeri del polinomio di Chebyshev di grado 5:

$$x_{4-i} = \cos\left(\frac{(2i+1)}{10}\pi\right) + 1 \qquad i = 4, 3, 2, 1, 0$$

Differenze divise e polinomio di interpolazione di Newton

Si deriva una **diversa rappresentazione** del polinomio di interpolazione di Lagrange, che consenta di calcolarlo con una minore complessità e, se si aggiungono coppie di dati (x_i, y_i), di derivare da un polinomio $p_n(x)$ di grado n uno di grado superiore riusando i calcoli eseguiti.

Esempio. Dato (x_0, y_0), $p_0(x) = y_0$.
Se si considera oltre a (x_0, y_0) anche (x_1, y_1), si ha

$$\begin{aligned}
p_1(x) &= \frac{x - x_1}{x_0 - x_1} y_0 + \frac{x - x_0}{x_1 - x_0} y_1 = \frac{x_1 y_0 - x y_0 + x y_1 - x_0 y_1}{x_1 - x_0} \\
&= \frac{x_1 y_0 - x_0 y_1}{x_1 - x_0} + \frac{y_1 - y_0}{x_1 - x_0} x + \frac{y_1 - y_0}{x_1 - x_0} x_0 - \frac{y_1 - y_0}{x_1 - x_0} x_0 \\
&= \frac{x_1 y_0 - \cancel{x_0 y_1} + \cancel{x_0 y_1} - y_0 x_0}{x_1 - x_0} + \frac{y_1 - y_0}{x_1 - x_0}(x - x_0) \\
&= y_0 + \frac{y_1 - y_0}{x_1 - x_0}(x - x_0) = p_0(x) + \frac{y_1 - y_0}{x_1 - x_0}(x - x_0)
\end{aligned}$$

Definizione di differenze divise

Si chiama differenza divisa di $f(x)$ relativa ai nodi x_0, x_1 la quantità

$$f[x_0, x_1] = \frac{f(x_0) - f(x_1)}{x_0 - x_1} = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = f[x_1, x_0]$$

In pratica $f[x_0, x_1]$ è un rapporto incrementale.

Si chiama differenza divisa di ordine 2 di $f(x)$ relativa agli argomenti x_0, x_1, x_2 la quantità

$$f[x_0, x_1, x_2] = \frac{f[x_0, x_1] - f[x_1, x_2]}{x_0 - x_2} = \frac{f[x_2, x_1] - f[x_1, x_0]}{x_2 - x_0} = f[x_2, x_1, x_0]$$

Definizione ricorsiva

Detta **differenza divisa di ordine 0** relativa all'argomento x_0 di $f(x)$ la quantità $f[x_0] = f(x_0)$, si definisce **differenza divisa di ordine** m ($m \geqslant 1$) di $f(x)$ relativa a $m + 1$ argomenti $x_0, x_1, \dots, x_m$ la quantità

$$f[x_0, x_1, \dots, x_m] = \frac{f[x_0, x_1, \dots, x_{m-1}] - f[x_1, x_2, \dots, x_m]}{x_0 - x_m}.$$

Tabella delle differenze divise

Questi risultati si schematizzano nella seguente

Tabella delle differenze divise

$$
\begin{array}{lllllllll}
x_0 \rightarrow f[x_0] & & & & & & & & \\
& \searrow & & & & & & & \\
x_1 \rightarrow f[x_1] & \rightarrow & f[x_0,x_1] & & & & & & \\
& \searrow & & \searrow & & & & & \\
x_2 \rightarrow f[x_2] & \rightarrow & f[x_1,x_2] & \rightarrow & f[x_0,x_1,x_2] & & & & \\
& \searrow & & \searrow & & \searrow & & & \\
x_3 \rightarrow f[x_3] & \rightarrow & f[x_2,x_3] & \rightarrow & f[x_1,x_2,x_3] & \rightarrow & f[x_0,x_1,x_2,x_3] & & \\
& \searrow & & \searrow & & \searrow & & \searrow & \\
x_4 \rightarrow f[x_4] & \rightarrow & f[x_3,x_4] & \rightarrow & f[x_2,x_3,x_4] & \rightarrow & f[x_1,x_2,x_3,x_4] & \rightarrow & f[x_0,x_1,x_2,x_3,x_4] \\
\vdots \quad \vdots & & \vdots & & \vdots & & \vdots & & \vdots
\end{array}
$$

Esempio. $x_0 = 0$, $x_1 = 1$, $x_2 = 3$; $f(x_0) = 1$, $f(x_1) = 3$, $f(x_2) = 2$

$$f[x_0,x_1] = \frac{1-3}{0-1} = 2$$

$$f[x_1,x_2] = \frac{3-2}{1-3} = -\frac{1}{2} \qquad \Longrightarrow$$

$$f[x_0,x_1,x_2] = \frac{2+\frac{1}{2}}{0-3} = -\frac{5}{6}$$

x_i	$f[x_i]$	$f[x_i,x_{i+1}]$	$f[x_i,x_{i+1},x_{i+2}]$
0	1		
1	3	2	
3	2	$-1/2$	$-5/6$

Codice Matlab

```
function [d] = tabDiff(x, y)
% TABDIFF - Tabella delle differenze divise sui nodi x e i valori y
  n = length(x);  % numero dei punti di interpolazione
                  % = grado polinomio + 1
  d = y;
  for k = 2 : n
    d(k:n) = ( d(k:n) - d(k-1 : n-1) ) ./ ( x(k:n) - x(1 : n-k+1) );
  end
end
```

In `d` viene calcolata di volta in volta una colonna della tabella, iniziando a memorizzare le differenze divise di ordine k dall'elemento $k+1$ fino all'ultimo.
In output `d` contiene la diagonale della tabella delle differenze divise.

	$\mathtt{k} = 1$	$\mathtt{k} = 2$	$\mathtt{k} = 3$	$\mathtt{k} = 4$
$\mathtt{d(1)}$	$\mathtt{y(1)} = f(\mathtt{x(1)})$			
$\mathtt{d(2)}$	$\mathtt{y(2)} = f(\mathtt{x(2)})$	$f[\mathtt{x(1)},\mathtt{x(2)}]$		
$\mathtt{d(3)}$	$\mathtt{y(3)} = f(\mathtt{x(3)})$	$f[\mathtt{x(2)},\mathtt{x(3)}]$	$f[\mathtt{x(1)},\mathtt{x(2)},\mathtt{x(3)}]$	
$\mathtt{d(4)}$	$\mathtt{y(4)} = f(\mathtt{x(4)})$	$f[\mathtt{x(3)},\mathtt{x(4)}]$	$f[\mathtt{x(2)},\mathtt{x(3)},\mathtt{x(4)}]$	$f[\mathtt{x(1)},\mathtt{x(2)},\mathtt{x(3)},\mathtt{x(4)}]$

Proprietà delle differenze divise

Teorema

Le differenze divise sono funzioni simmetriche dei loro argomenti, ossia se $(i_0, i_1, \ldots, i_n)$ è una permutazione di $(0, 1, \ldots, n)$, allora

$$f[x_{i_0}, x_{i_1}, \ldots, x_{i_n}] = f[x_0, x_1, \ldots, x_n].$$

Inoltre

$$f[x_0, x_1, \ldots, x_n] = \sum_{j=0}^{n} \frac{f(x_j)}{\prod\limits_{\substack{i=0,\ldots,n \\ i\neq j}} (x_j - x_i)}$$

Proprietà. Dati $n+1$ nodi **distinti** $x_0, x_1, \ldots, x_n$ e assegnati valori $y_i = f(x_i)$ di $f(x)$ in x_i, $i = 0, \ldots, n$, è noto che esiste uno e un solo polinomio di interpolazione di grado al più n tale che $f(x_i) = y_i = p_n(x_i)$ $\forall i = 0, \ldots, n$. Nella rappresentazione di Lagrange, il coefficiente di x^n in $p_n(x)$ è

$$\frac{f(x_0)}{\prod\limits_{i=1,\ldots,n} (x_0 - x_i)} + \frac{f(x_1)}{\prod\limits_{\substack{i=0,\ldots,n \\ i\neq 1}} (x_1 - x_i)} + \ldots + \frac{f(x_n)}{\prod\limits_{i=0,\ldots,n-1} (x_n - x_i)} = \sum_{j=0}^{n} \frac{f(x_j)}{\prod\limits_{\substack{i=0,\ldots,n \\ i\neq j}} (x_j - x_i)}$$

$$= f[x_0, \ldots, x_n]$$

Osservazione

Sia $f(x) = p_n(x)$ un polinomio di grado al più n. Allora:

$$f[x, x_0] = \frac{p_n(x) - p_n(x_0)}{x - x_0} = q_{n-1}(x), \text{ quoziente di } p_n(x) \text{ e } (x - x_0),$$

è un polinomio di grado al più $n-1$.

$$f[x, x_0, x_1] = \frac{f[x, x_0] - f[x_0, x_1]}{x - x_1} = \frac{q_{n-1}(x) - q_{n-1}(x_1)}{x - x_1} = q_{n-2}(x)$$

è un polinomio di grado al più $n-2$

$$\vdots$$

$$f[x, x_0, x_1, \ldots, x_{n-1}] = \text{costante}$$

$$f[x, x_0, x_1, \ldots, x_n] = 0$$

La differenza divisa di ordine n $f[x, x_0, \ldots, x_{n-1}]$ di un polinomio di grado n è una costante; le differenze divise di ordine maggiore di n sono nulle.

Polinomio di Newton

$$p_n(x) = f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) + \dots$$
$$\dots + f[x_0, x_1, \dots, x_n](x - x_0)(x - x_1) \cdots (x - x_{n-1})$$

L'espressione del polinomio di Newton è conseguenza del seguente teorema:

Teorema

Sia $p_k(x)$ in polinomio di grado k tale che $p_k(x_i) = y_i$, $i = 0, 1, \dots, k$. Allora

$$p_{k+1}(x) = p_k(x) + f[x_0, x_1, \dots, x_{k+1}](x - x_0) \cdots (x - x_k)$$

è un polinomio di grado $k + 1$ tale che $p_{k+1}(x_i) = y_i$, $i = 0, 1, \dots, k + 1$.

Dim. Sia $g(x)$ il polinomio di grado $k + 1$ di interpolazione in $x_0, \dots, x_k, x_{k+1}$, ossia tale che $g(x_i) = y_i \ \forall i = 0, \dots, k + 1$. Si vuole far vedere che $p_{k+1}(x)$ coincide con $g(x)$. Per fare questo, si osserva che $p_{k+1}(x_i) = p_k(x_i) = y_i$, $\forall i = 0, \dots, k$.

Polinomio di Newton

Inoltre il coefficiente di x^{k+1} in $p_{k+1}(x)$ è

$$f[x_0, \dots, x_{k+1}] = \sum_{j=0}^{k+1} \frac{f(x_j)}{\displaystyle\prod_{\substack{i=0,\dots,k+1 \\ i \neq j}} (x_j - x_i)}$$

come quello di $g(x)$. Allora $p_{k+1}(x) - g(x)$ è un polinomio di grado k (il coefficiente di x^{k+1} vale 0), che si annulla in $k + 1$ punti x_0, x_1, ..., x_k, ossia il polinomio $p_{k+1}(x) - g(x)$ è identicamente nullo. Dunque $p_{k+1}(x)$ coincide con $g(x)$ ossia è polinomio di interpolazione relativo ai punti x_0, x_1, ..., x_k, x_{k+1}.

Polinomio di Newton

Dunque si ha

$$
\begin{aligned}
p_0(x) &= y_0 = f[x_0] \\
p_1(x) &= p_0(x) + f[x_0, x_1](x - x_0) \\
&= f[x_0] + f[x_0, x_1](x - x_0) \\
p_2(x) &= p_1(x) + f[x_0, x_1, x_2](x - x_0)(x - x_1) \\
&= f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) \\
p_3(x) &= p_2(x) + f[x_0, x_1, x_2, x_3](x - x_0)(x - x_1)(x - x_2) \\
&= f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) \\
&\quad + f[x_0, x_1, x_2, x_3](x - x_0)(x - x_1)(x - x_2) \\
&\vdots
\end{aligned}
$$

Polinomio di Newton

Un altro modo per ricavare il polinomio di Newton è:

$$
\begin{aligned}
f[x, x_0, \dots, x_n] &= \frac{f[x, x_0, \dots, x_{n-1}] - f[x_0, \dots, x_n]}{x - x_n} \\
&\Rightarrow f[x, x_0, \dots, x_{n-1}] = (x - x_n) f[x, x_0, \dots, x_n] + f[x_0, \dots, x_n] \\
f[x, x_0, \dots, x_{n-1}] &= \frac{f[x, x_0, \dots, x_{n-2}] - f[x_0, \dots, x_{n-1}]}{x - x_{n-1}} \\
&\Rightarrow f[x, x_0, \dots, x_{n-2}] = (x - x_{n-1}) f[x, x_0, \dots, x_{n-1}] + f[x_0, \dots, x_{n-1}] \\
f[x, x_0, \dots, x_{n-2}] &= \frac{f[x, x_0, \dots, x_{n-3}] - f[x_0, \dots, x_{n-2}]}{x - x_{n-2}} \\
&\Rightarrow f[x, x_0, \dots, x_{n-3}] = (x - x_{n-2}) f[x, x_0, \dots, x_{n-2}] + f[x_0, \dots, x_{n-2}] \\
&\vdots \\
f[x, x_0, x_1, x_2] &= \frac{f[x, x_0, x_1] - f[x_0, x_1, x_2]}{x - x_2} \quad \Rightarrow \quad f[x, x_0, x_1] = (x - x_2) f[x, x_0, x_1, x_2] + f[x_0, x_1, x_2] \\
f[x, x_0, x_1] &= \frac{f[x, x_0] - f[x_0, x_1]}{x - x_1} \quad \Rightarrow \quad f[x, x_0] = (x - x_1) f[x, x_0, x_1] + f[x_0, x_1] \\
f[x, x_0] &= \frac{f(x) - f(x_0)}{x - x_0} \quad \Rightarrow \quad f(x) = (x - x_0) f[x, x_0] - f(x_0)
\end{aligned}
$$

Polinomio di Newton

Allora

$$\begin{aligned}
f(x) &= f(x_0) + (x - x_0)f[x, x_0] \\
&= f(x_0) + (x - x_0)f[x_0, x_1] + (x - x_0)(x - x_1)f[x, x_0, x_1] \\
&= f(x_0) + (x - x_0)f[x_0, x_1] + (x - x_0)(x - x_1)f[x_0, x_1, x_2] + \\
&\quad + (x - x_0)(x - x_1)(x - x_2)f[x, x_0, x_1, x_2] \\
&\ \vdots \\
&= f(x_0) + (x - x_0)f[x_0, x_1] + (x - x_0)(x - x_1)f[x_0, x_1, x_2] + \\
&\quad + (x - x_0)(x - x_1)(x - x_2)f[x_0, x_1, x_2, x_3] + \\
&\quad + (x - x_0)(x - x_1)(x - x_2)(x - x_3)f[x_0, x_1, x_2, x_3, x_4] + \ldots + \\
&\quad + (x - x_0)(x - x_1)\cdots(x - x_{n-1})f[x_0, \ldots, x_n] + \\
&\quad + \omega(x)f[x, x_0, x_1, \ldots, x_n]
\end{aligned}$$

con $\omega(x) = \prod_{i=0}^{n}(x - x_i)$

Polinomio di Newton

Detto $p_0(x) = f(x_0)$ e $p_1 = p_0(x) + f[x_0, x_1](x - x_0)$ si ha

$$p_k(x) = p_{k-1}(x) + f[x_0, x_1, \ldots x_k](x - x_0)\cdots(x - x_{k-1})$$

e inoltre

$$f(x) = p_n(x) + f[x, x_0, \ldots, x_n]\omega(x)$$

$$\begin{aligned}
p_n(x) &= f(x_0) + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) + \ldots + \\
&\quad + f[x_0, \ldots, x_n](x - x_0)\cdots(x - x_{n-1}) \\
&= \sum_{i=0}^{n}\left(f[x_0, \ldots, x_i]\prod_{j=0}^{i-1}(x - x_j)\right)
\end{aligned}$$

$p_n(x)$ si dice **polinomio di Newton** ed è il polinomio di interpolazione di grado al più n di $f(x)$ in $x_0, \ldots, x_n$.
Infatti, se $f(x)$ fosse un polinomio di grado al più n, allora $f[x, x_0, \ldots, x_n]$ essendo una differenza divisa di ordine $n + 1$ di un polinomio di grado n, sarebbe 0.

Polinomio di Newton

Pertanto $p_n(x)$ è un polinomio di grado al più n. Inoltre è un polinomio di interpolazione in $x_0 \dots x_n$ poiché

$$f(x_i) = p_n(x_i) + f[x_i, x_0, \dots, x_n] \underbrace{\omega(x_i)}_{=0} = p_n(x_i) \qquad \forall i = 0, \dots, n$$

Poiché il polinomio di interpolazione di grado al più n è unico, il **polinomio di Newton** è il polinomio di interpolazione di $f(x)$ in $x_0, \dots, x_n$.

I punti $x_0, x_1, \dots, x_n$ possono essere ordinati in modo arbitrario.

Esempio

Calcoliamo il polinomio di interpolazione nella forma di Newton per un insieme di dati:

x	$f(x)$	Tavola delle differenze divise			
1.0	0.7651977				
1.3	0.6200860	−0.4837057			
1.6	0.4554022	−0.5489460	−0.1087339		
1.9	0.2818186	−0.5786120	−0.0494433	0.0658784	
2.2	0.1103623	−0.5715210	0.0118183	0.0680685	0.0018251

$$\begin{aligned} p_4(x) &= 0.7651977 \\ &\quad + (x-1.0)(-0.4837057) \\ &\quad + (x-1.0)(x-1.3)(-0.1087339) \\ &\quad + (x-1.0)(x-1.3)(x-1.6)(0.0658784) \\ &\quad + (x-1.0)(x-1.3)(x-1.6)(x-1.9)(0.0018251) \\ p_4(1.5) &= 0.5118200 \\ R(1.5) &= \omega(1.5) f[1.5, 1.0, 1.3, 1.6, 1.9, 2.2] \end{aligned}$$

Tuttavia $f[1.5, 1.0, \dots, 2.2]$ non è calcolabile.

Implementare in Matlab la formula di Newton

La formula di Newton può essere scritta come

$$p_n(x) = d_0 + (x - x_0)\Big(d_1 + (x - x_1)\Big(d_2 + \dots (d_{n-1} + (x - x_{n-1})d_n) \dots\Big)\Big)$$

con $d_i = f[x_0, \dots, x_i]$, $i = 0, \dots, n$.
Note le differenze divise di $f(x)$ di ordine k relative a $k+1$ argomenti, si può valutare $p_n(\xi)$ usando lo **schema di Horner generalizzato**.
Complessità: n prodotti, $2n$ somme.

Poiché la tabella delle differenze divise richiede $\dfrac{n(n+1)}{2}$ prodotti e $n(n+1)$ somme, la complessità è la metà di quella necessaria a calcolare il polinomio di Lagrange.

```
function [z, d] = polyNewton(x, y, xx)
% POLYNEWTON - Valuta in xx il polinomio di Newton su (x, y)
  n = length(x);
  d = tabDiff(x,y);
  z = d(n)*ones(size(xx));
  for i = n-1 : -1 : 1
    z = z .* (xx - x(i)) + d(i);
  end
end
```


Aggiungere osservazioni

Se si deve aggiungere un'osservazione $\big(x_{n+1}, f(x_{n+1})\big)$ e calcolare $p_{n+1}(\xi)$ partendo da $p_n(\xi)$, basta aggiungere un nuovo punto alla tabella delle differenze divise, calcolare la nuova riga di valori

$$x_{n+1} \quad f(x_{n+1}) \quad f[x_n, x_{n+1}] \quad f[x_{n-1}, x_n, x_{n+1}] \quad \dots \quad f[x_0, \dots, x_{n+1}]$$

e poi

$$p_{n+1}(\xi) = p_n(\xi) + f[x_0, \dots, x_{n+1}]\,\underbrace{(\xi - x_0)\cdots(\xi - x_n)}_{\omega(\xi)}$$

Si osservi che **l'errore di interpolazione** è

$$R(x) = f(x) - p_n(x) = f[x, x_0, \dots, x_n]\omega(x)$$

La formula è generale e fornisce una caratterizzazione di $R(x)$ anche per funzioni $f(x) \notin C^{n+1}([a,b])$. **Tuttavia**, poiché $R(x)$ è univocamente determinato, se $f(x) \in C^{n+1}([a,b])$, dati $x_0, x_1, \dots, x_n$ distinti, segue che esiste ξ dipendente da x e da $x_0, x_1, \dots, x_n$ tale che

$$\boxed{\frac{f^{(n+1)}(\xi)}{(n+1)!} = f[x, x_0, x_1, \dots, x_n]}$$

Il concetto di differenza divisa è una generalizzazione della definizione di derivata.

Formula di Taylor

La differenza divisa di ordine 1 di $f(x)$ è indeterminata se i due argomenti coincidono. Tuttavia, ricordando che, se $f \in C^1([a,b])$, la differenza divisa di ordine 1 di $f(x)$ relativa a x_0, x_1 è tale che

$$f[x_0, x_1] = f'(\xi) \qquad x_0 < \xi < x_1\,,$$

segue che ha senso definire la **differenza divisa di ordine 1 di $f(x)$ con argomenti coincidenti** come

$$f[x_0, x_0] = f'(x_0)$$

Infatti $f'(x_0) = \lim_{x \to x_0} \dfrac{f(x) - f(x_0)}{x - x_0} = \lim_{x \to x_0} f[x, x_0] = f[x_0, x_0]$.
In modo analogo, se $f(x) \in C^n$ in un intorno di x_0, si definisce la **differenza divisa di ordine n di $f(x)$ relativa a $n+1$ argomenti coincidenti** come

$$f[\underbrace{x_0, x_0, \ldots, x_0}_{(n+1)\text{ volte}}] = \frac{f^{(n)}(x_0)}{n!}.$$

Formula di Taylor

Con tale assunzione, la formula di Newton relativa a $n+1$ argomenti coincidenti x_0 di una funzione f (tale che $f(x) \in C^n$ in un intorno $\mathcal{I}$ di x_0 ed esista la derivata $(n+1)$-esima in $\mathcal{I}$) diventa la **formula di Taylor**:

$$\begin{aligned} f(x) &= f(x_0) + f[x_0, x_0](x - x_0) + f[x_0, x_0, x_0](x - x_0)^2 + \ldots + \\ &\quad + f[\underbrace{x_0, x_0, \ldots, x_0}_{n+1 \text{ volte}}](x - x_0)^n + f[x, x_0, \ldots, x_0](x - x_0)^{n+1} \\ &= f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \ldots + \\ &\quad + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n + \frac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)^{n+1} \end{aligned}$$

con ξ appartenente al più piccolo intervallo di estremi x e x_0. Allora

$$\boxed{f(x) = p_n(x) + R_n(x)}$$

Ogni funzione $f(x) \in C^n$ in un intorno di x_0 e per cui esista la derivata $(n+1)$-esima in tale intorno si può scrivere come la somma di un polinomio di grado n in $(x - x_0)$, detto **polinomio di Taylor**, e di un termine resto.

Polinomio di Taylor ed errore di approssimazione

Teorema

Sia $f(x) \in C^n([x_0, x])$ ed esista la derivata $(n+1)$-esima di $f(x)$ in un intorno di x. Allora, per ogni x appartenente a un intorno di x_0 esiste ξ contenuto nel più piccolo intervallo che contiene $\mathcal{I}(x, x_0)$ (aperto) tale che

$$f(x) = f(x_0) + \sum_{i=1}^{n} \left(\frac{f^{(i)}(x_0)}{i!} (x - x_0)^i \right) + \frac{f^{(n+1)}(\xi)}{(n+1)!} (x - x_0)^{n+1}$$

Si può ottenere una maggiorazione dell'errore. Se $\left|f^{(n+1)}(x)\right| \leqslant M \; \forall x \in \mathcal{I}(x_0)$

$$|R(x)| \leqslant \frac{M|x - x_0|^{n+1}}{(n+1)!}$$

Pertanto preso $\epsilon \geqslant \dfrac{Mh^{n+1}}{(n+1)!}$, il polinomio di Taylor $p_n(x)$ approssima la funzione $f(x)$ entro la tolleranza ϵ per $x \in [x_0 - h, \, x_0 + h]$, ossia per $|x - x_0| \leqslant h$.
Se $f(x)$ è un polinomio di grado n, $f(x) \equiv p_n(x)$.

Polinomio di Taylor ed errore di approssimazione

Si osservi che **la funzione $f(x)$ e il polinomio di Taylor coincidono in x_0, non solo per quanto riguarda il valore della funzione, ma anche per i valori delle derivate fino all'ordine n**

$$f^{(k)}(x_0) = p_n^{(k)}(x_0) \qquad \forall k = 0, \ldots, n.$$

Tuttavia l'approssimazione di Taylor è solo locale, esatta solo nel punto x_0 e tanto peggiore quanto più ci si allontana da x_0.
Infatti **tutte le informazioni usate nell'approssimazione sono concentrate nel punto x_0**.

Pertanto si limita l'uso del polinomio di Taylor alle situazioni in cui occorre approssimare una funzione nelle vicinanze ad x_0.
Inoltre si può usare solo per funzioni estremamente regolari e il calcolo delle derivate può essere estremamente costoso.

Se $x_0 = 0$, il polinomio di Taylor viene designato come formula di Mac Laurin.

Esempio

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots + \frac{x^n}{n!} + \frac{x^{n+1}}{(n+1)!}e^{\theta x}, \qquad 0 < \theta < 1$$

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots + (-1)^n \frac{x^{2n+1}}{(2n+1)!} +$$

$$+ \ (-1)^{n+1} \frac{x^{2n+3}}{(2n+3)!} \cos(\theta' x), \quad 0 < \theta' < 1$$

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} + \ldots + (-1)^n \frac{x^{2n}}{(2n)!} +$$

$$+ \ (-1)^{n+1} \frac{x^{2n+2}}{(2n+2)!} \cos(\theta'' x), \qquad 0 < \theta'' < 1$$

Se si vuole approssimare $\sin(x)$ entro una tolleranza ϵ in un intervallo, l'ampiezza h di tale intervallo deve essere tale che

$$|R_{2n+1}(x)| \leqslant \frac{|x^{2n+3}|}{(2n+3)!} \leqslant \epsilon \qquad \forall x \text{ tale che } |x| \leqslant h$$

Esempio

Sia $\epsilon = 10^{-5}$ e $p_5(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!}$, $(n = 2)$, allora $p_5(x)$ approssima $\sin(x)$ entro $|x| \leqslant h$ con h tale che

$$\left|\frac{x^7}{7!}\right| \leqslant 10^{-5} \qquad \Rightarrow \qquad x \leqslant \sqrt[7]{10^{-5} 7!} \approx 0.6525$$

Se dunque si sceglie $h = 0.65$ si ha $p_5(0.5) = 0.4794270$ con $\sin(0.5) = 0.4794255$. Dunque l'errore è $0.15 \cdot 10^{-5}$.

Esempio

Si calcoli il polinomio di Mac Laurin in $f(x) = \sqrt{1+x}$ di grado 3 e si trovi una approssimazione di $f(0.1)$ e una stima dell'errore.

$$f(x) = \sqrt{1+x} \qquad f'(x) = \frac{1}{2\sqrt{1+x}}$$

$$f''(x) = -\frac{1}{4(1+x)^{3/2}} \qquad f'''(x) = \frac{3}{8(1+x)^{5/2}}$$

$$f^{(4)}(x) = -\frac{15}{16(1+x)^{7/2}} \qquad p_3(x) = 1 + \frac{1}{2}x - \frac{1}{8}x^2 + \frac{1}{16}x^3$$

$$p_3(0.1) \approx f(0.1) = \sqrt{1.1} \approx 1 + \frac{1}{2}0.1 - \frac{1}{8}(0.1)^2 + \frac{1}{16}(0.1)^3 \approx 1.0488125$$

$$R_3 = \frac{x^4}{4!}\left(-\frac{15}{16(1+\xi)^{7/2}}\right)$$

$$|R_3| \leqslant \frac{(0.1)^4 15}{24 \cdot 16} \max_{[0,0.1]} \frac{1}{(1+\xi)^{7/2}} = \frac{0.0005}{128} 1 \approx 3.9 \cdot 10^{-6}$$

$$\sqrt{1.1} - p_3(0.1) = 3.7 \cdot 10^{-6}$$

Esempio

Si usa il polinomio $p_3(x)$ per calcolare $\int_0^{0.1} \sqrt{1+x}\,dx$:

$$\int_0^{0.1} \left(1 + \frac{1}{2}x - \frac{1}{8}x^2 + \frac{1}{16}x^3\right) dx = 0.102459858$$

$$\text{errore: } E = -\int_0^{0.1} \frac{x^4}{4!}\frac{15}{16}\frac{1}{(1+\xi(x))^{7/2}} dx, \ \text{con } |E| \leqslant \int_0^{0.1} \frac{x^4 5}{128} dx \approx 7.82 \cdot 10^{-8}$$

$$E < 0 \quad \Rightarrow \quad 0.1024598958 - 7.8 \cdot 10^{-8} \leqslant \int_0^{0.1} \sqrt{1+x}\,dx \leqslant 0.1024598958$$

$$\text{errore effettivo} = 7.4 \cdot 10^{-8} \text{ poiché } \int_0^{0.1} \sqrt{1+x}\,dx \leqslant 0.1024598958$$

Tavola di $p_3(x)$: l'errore cresce al crescere della distanza di x da 0.

x	0.1	0.5	1	2	10
$p_3(x)$	1.048813	1.2266	1.438	2.00	56.00
errore	$4.0 \cdot 10^{-6}$	$1.9 \cdot 10^{-3}$	$2.4 \cdot 10^{-2}$	$2.7 \cdot 10^{-1}$	2.68

Interpolazione di Birkoff-Hermite

L'interpolazione di Birkoff-Hermite è una generalizzazione dei polinomi di Taylor e di Lagrange (o Newton).

Si impongono condizioni sui valori che deve assumere un polinomio in punti prefissati e condizioni sui valori delle derivate.

È essenziale che, se si impone il valore di una derivata del polinomio in un punto, siano assegnati anche i valori di tutte le derivate di ordine inferiore in quel punto e il valore della funzione in quel punto.

Solo se vale tale condizione l'interpolazione di Birkoff-Hermite ha una e una sola soluzione.

Esempio

Siano assegnate le condizioni $f(x_0) = p(x_0)$, $f'(x_1) = p'(x_1)$, $f(x_2) = p(x_2)$ e $x_0 < x_2$ e $x_1 = (x_0 + x_2)/2$. Si vogliono trovare i coefficienti del polinomio interpolante di grado 2, $p_2(x) = a_0 + a_1 x + a_2 x^2$ (derivata: $p_2'(x) = a_1 + 2a_2 x$), imponendo le condizioni di interpolazione:

$$\begin{cases} a_0 + a_1 x_0 + a_2 x_0^2 = f(x_0) \\ a_1 + 2a_2 \dfrac{x_0 + x_2}{2} = f'(x_1) \\ a_0 + a_1 x_2 + a_2 x_2^2 = f(x_2) \end{cases}$$

La matrice del sistema è singolare:

$$\left| \begin{pmatrix} 1 & x_0 & x_0^2 \\ 0 & 1 & x_0 + x_2 \\ 1 & x_2 & x_2^2 \end{pmatrix} \right| = 0$$

Il sistema non ha soluzione unica. In casi come questi non si sa nè se esiste la soluzione e, se esiste, essa non è unica. Nel caso si imponga $f(x_0) = p(x_0)$, $f(x_1) = p(x_1)$, $f'(x_1) = p'(x_1)$, la soluzione esiste ed è unica:

$$\begin{cases} a_0 + a_1 x_0 + a_2 x_0^2 = f(x_0) \\ a_0 + a_1 x_1 + a_2 x_1^2 = f(x_1) \\ a_1 + 2a_2 x_1 = f'(x_1) \end{cases} \qquad \det(A) = (x_1 - x_0)^2 \neq 0 \Leftrightarrow x_1 \neq x_0$$

Polinomio di Birkoff-Hermite

Si assume che la funzione f da approssimare sia sufficientemente regolare in $[a, b]$, ossia sia di classe $C^m([a, b])$ con m pari al massimo ordine delle derivate per cui si impongono condizioni.

Teorema

Dati $n+1$ punti distinti $x_0, x_1, \ldots, x_n \in [a, b]$ e $n+1$ interi non negativi $m_0, m_1, \ldots, m_n \in \mathbb{N}$, assegnati $n+1+\sum_{i=0}^n m_i$ valori $\{f_i^{(k)}\}$, $i = 0, \ldots, n$, $k = 0, 1, \ldots, m_i$, esiste uno e un solo polinomio di grado $M = n + \sum_{i=0}^n m_i$ tale che

$$p_M^{(k)}(x_i) = f_i^{(k)}, \quad i = 0, \ldots, n, \quad k = 0, \ldots, m_i$$

Tale polinomio si dice **polinomio di Birkoff-Hermite**.

nodi	x_0	x_1	$\ldots$	x_n
funzione	f_0	f_1	$\ldots$	f_n
derivata 1ª	f_0'	f_1'	$\ldots$	f_n'
derivata 2ª	f_0''	f_1''	$\ldots$	f_n''
$\vdots$	$\vdots$	$\vdots$		$\vdots$
n. dati	$1+m_0$	$1+m_1$	$\ldots$	$1+m_n$

Se $n = 0$ e $m_0 = M$, il polinomio di Birkoff-Hermite è il polinomio di Taylor di grado M in x_0.
Se $m_i = 0$ per $i = 0, \ldots, n$, il polinomio di Birkoff-Hermite è il polinomio di Lagrange di grado $n \equiv M$ nei punti $x_0, \ldots, x_n$.

Polinomio di Hermite

Particolarmente interessante è il caso in cui $m_i = 1$ ($i = 0, \ldots, n$). In tal caso $p_M(x)$ e $f(x)$ hanno lo stesso **"profilo"**, poiché, nei nodi x_i, $p_M(x)$ e $f(x)$ coincidono e coincidono anche le rette tangenti ai grafici delle due funzioni in $\big(x_i, f(x_i)\big)$ $i = 0, \ldots, n$.
Il polinomio di interpolazione si dice **polinomio di Hermite** ed è di grado $M = 2n+1$ in quanto soddisfa le $M+1$ condizioni:

$$p_{2n+1}(x_i) = f(x_i) = f_i \qquad p_{2n+1}'(x_i) = f'(x_i) = f_i'$$

$$p_{2n+1}(x) = \sum_{i=0}^n f_i H_i(x) + \sum_{i=0}^n f_i' K_i(x) \qquad \text{polinomio di Hermite}$$

$$H_i(x) = L_i^2(x)\big(1 - 2(x - x_i)L_i'(x_i)\big) \qquad K_i(x) = L_i^2(x)(x - x_i)$$

con $L_i(x)$ l'i-esimo polinomio di Lagrange sui nodi x_i. Infatti si osserva che

$$p_{2n+1}'(x) = \sum_{i=0}^n f_i H_i'(x) + \sum_{i=0}^n f_i' K_i'(x)$$

$$H_i'(x) = 2L_i(x)L_i'(x)\big(1 - 2(x - x_i)L_i'(x_i)\big) - L_i(x)^2 2L_i'(x_i)$$

$$K_i'(x) = L_i(x)\Big(L_i(x) + 2(x - x_i)L_i'(x)\Big)$$

Polinomio di Hermite

Dunque vale che

$$H_i(x_j) = \begin{cases} 1 & \text{se } i = j \\ 0 & \text{se } i \neq j \end{cases} \qquad K_i(x_j) = 0$$

$$H_i'(x_j) = 0 \qquad\qquad K_i'(x_j) = \begin{cases} 1 & \text{se } i = j \\ 0 & \text{se } i \neq j \end{cases}$$

Dunque, per $j = 0, \dots, n$

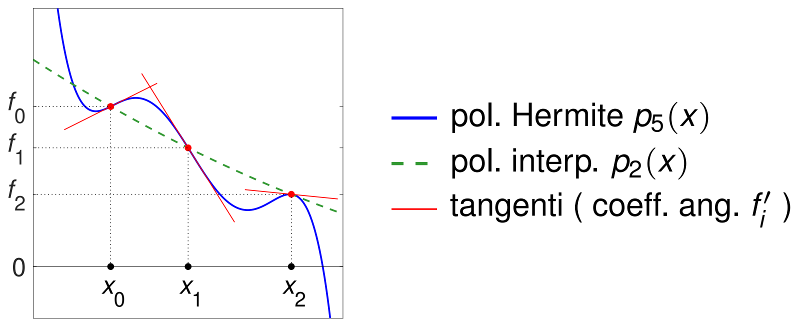
$$\begin{aligned} p_{2n+1}(x_j) &= f_j \\ p'_{2n+1}(x_j) &= f'_j \end{aligned}$$

Tale polinomio è unico.
Infatti, assunto che esista un altro polinomio $q(x)$ di grado $2n+1$ tale che $q(x_i) = f_i$, $q'(x_i) = f'_i \ \forall i = 0, \dots, n$; il polinomio di grado al più $2n+1$ dato da $p_{2n+1}(x) - q(x)$ ha x_i come zeri di molteplicità almeno 2 e pertanto si annulla almeno $2n+2$ volte. Per il Teorema fondamentale dell'algebra, allora $p_{2n+1}(x) - q(x) \equiv 0$. Segue che $p_{2n+1} \equiv q$.

Altra formulazione del polinomio di Hermite

È complesso ricavare il polinomio di Hermite dalla sua formula. Se si ricorda che $f[x_i, x_i] = f'_i$, si può ricavarlo come un polinomio di Newton di grado $2n+1$ relativo agli argomenti $x_0,\ x_0,\ x_1,\ x_1,\ \dots,\ x_n,\ x_n$.
Per calcolare $p_{2n+1}(x)$ si costruisce la seguente tabella delle differenze divise:

$$\begin{array}{llllll}
x_0 & f(x_0) & & & & \\
x_0 & f(x_0) & f[x_0,x_0] = f'_0 & & & \\
x_1 & f(x_1) \to & f[x_0,x_1] \to & f[x_0,x_0,x_1] & & \\
x_1 & f(x_1) & f[x_1,x_1] = f'_1 \to & f[x_0,x_1,x_1] \to & f[x_0,x_0,x_1,x_1] & \\
x_2 & f(x_2) \to & f[x_2,x_1] \to & f[x_1,x_1,x_2] \to & f[x_0,x_1,x_1,x_2] \to & f[x_0,x_0,x_1,x_1,x_2] \\
x_2 & f(x_2) & f[x_2,x_2] = f'_2 \to & f[x_1,x_2,x_2] \to & f[x_1,x_1,x_2,x_2] \to & f[x_0,x_1,x_1,x_2,x_2] \to f[x_0,x_0 x_1,x_1,x_2,x_2] \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \quad \vdots
\end{array}$$



Altra formulazione del polinomio di Hermite

Analogamente a quanto visto per il polinomio di Newton, si può dimostrare che la funzione si scrive nel seguente modo:

$$\begin{aligned} f(x) = f(x_0) &+ f[x_0, x_0](x - x_0) \\ &+ f[x_0, x_0, x_1](x - x_0)^2 \\ &+ f[x_0, x_0, x_1, x_1](x - x_0)^2(x - x_1) \\ &+ \ldots + f[x_0, x_0, \ldots, x_n, x_n](x - x_0)^2 \cdots (x - x_{n-1})^2(x - x_n) \\ &+ f[x, x_0, x_0, \ldots, x_n, x_n](x - x_0)^2 \cdots (x - x_n)^2 \end{aligned}$$

$$\begin{aligned} f(x) &= p_{2n+1}(x) + f[x, x_0, x_0, \ldots, x_n, x_n](x - x_0)^2 \cdots (x - x_n)^2 \\ &= p_{2n+1}(x) + f[x, x_0, x_0, \ldots, x_n, x_n]\omega(x)^2 \end{aligned}$$

$$\begin{aligned} f'(x) = p'_{2n+1}(x) &+ \frac{d}{dx}\Big(f[x, x_0, x_0, \ldots, x_n, x_n]\Big)\omega(x)^2 \\ &+ f[x, x_0, x_0, \ldots, x_n, x_n]2\omega(x)\omega'(x) \end{aligned}$$

Poiché $f(x_i) = p_{2n+1}(x_i)$, $f'(x_i) = p'_{2n+1}(x_i)$ e il polinomio di Hermite è l'unico che soddisfa tali condizioni, segue che $p_{2n+1}(x)$ è tale polinomio.

Errore di interpolazione di Hermite

L'errore di interpolazione vale

$$f[x, x_0, x_0, \ldots, x_n, x_n](x - x_0)^2 \cdots (x - x_n)^2$$

Per funzioni regolari, si ottiene il seguente risultato.

Teorema

Sia $f \in C^{2n+2}([a, b])$ e sia $p_{2n+1}(x)$ il polinomio di Hermite relativo a f nei nodi distinti $x_0, x_1, \ldots, x_n \in [a, b]$.
Allora esiste ξ dipendente da $x_0, x_1, \ldots, x_n$ tale che

$$f(x) - p_{2n+1}(x) = \frac{f^{(2n+2)}(\xi)}{(2n+2)!}(x - x_0)^2 \cdots (x - x_n)^2 = \frac{f^{(2n+2)}(\xi)}{(2n+2)!}\omega(x)^2$$

Anche qui $\xi = \xi(x, x_0, \ldots, x_n)$, cioè il punto non noto ξ dipende da x e da tutti i nodi di interpolazione.

Algoritmo

In pseudocodice, un modo di costruire la tabella delle differenze divise per il polinomio di Hermite è il seguente:

for $i \leftarrow 0$ **to** n **do**
  $z_{2i} \leftarrow x_i$
  $z_{2i+1} \leftarrow x_i$
  $a_{2i,0} \leftarrow f_i$
  $a_{2i+1,0} \leftarrow f_i$
  $a_{2i+1,1} \leftarrow f_i'$
end for
for $i \leftarrow 1$ **to** n **do**
  $a_{2i,1} = \dfrac{a_{2i,0} - a_{2i-1,0}}{z_{2i} - z_{2i-1}}$
end for
for $j \leftarrow 2$ **to** $2n+1$ **do**
  for $i \leftarrow j$ **to** $2n+1$ **do**
    $a_{i,j} = \dfrac{a_{i-1,j-1} - a_{i,j-1}}{z_{i-j} - z_i}$
  end for
end for

```
x = x(:);    % colonna dei nodi di interp.
f = f(:);    % colonna dei valori f(x)
f1 = f1(:); % colonna delle derivate f'(x)
n = numel(x);
z = [x x]'; z = z(:); A = zeros(2*n);
a0 = [f f]'; A(:,1) = a0(:);
A(2:2:end, 2) = f1;
A(3:2:end, 2) = ...
    ( A(3:2:end,1) - A(1:2:(end-2),1) ) ...
    ./ ( z(3:2:end) - z(1:2:(end-2)) );
for k = 3 : 2*n
   A(k:end, k) = ...
    (A(k:end, k-1) - A(k-1:(end-1), k-1))...
    ./ ( z(k:end) - z(1:(end-(k-1))) );
end
```

Nota: in realtà, come nel caso della tabella di Newton, dato che servono solo gli elementi diagonali della tabella, si eseguono i calcoli mantenendo solo un vettore `d`, con i soli elementi diagonali, non l'intera matrice `A`. Ricordiamo che gli indici in Matlab partono da 1.

Complessità computazionale: $\mathcal{O}(2n^2)$ quozienti e $\mathcal{O}(4n^2)$ differenze.

Calcolo del polinomio di Hermite con lo schema di Horner

Avendo la tabella delle differenze divise del polinomio interpolante di Hermite, il valore del polinomio in $x = \eta$ si può calcolare con lo schema di Horner:

Algoritmo:

 $y \leftarrow a_{2n+1,2n+1}$
 for $i \leftarrow 2n$ **to** 0 **do**
  $y \leftarrow y \cdot (\eta - x_{\lfloor \frac{i}{2} \rfloor}) + a_{i,i}$
 end for
 Stampa y

```
% Matrice A: tabella delle diff. divise
xx = xx(:);  % colonna dei punti di valut.
y = A(end, end) * ones(size(xx));
for i = 2*n-1 : -1 : 1
   y = y .* (xx - x( floor((i+1)/2) )) ...
       + A(i, i);
end
```

Nota: ricordiamo che in Matlab gli indici degli array partono da 1, non da zero. Anche qui, nella realtà, si usa solo il vettore `d` degli elementi diagonali della tabella.

Bastano $2n+1$ prodotti e $4n+2$ somme.

La funzione *floor* (pavimento) è definita per ogni $t \in \mathbb{R}$ da:

$$\lfloor t \rfloor = \max\{\ell \in \mathbb{Z} \mid \ell \leqslant t\}$$

Nota: questa funzione coincide con la parte intera $[t]$ di t quando $t \geqslant 0$, ma non coincide con $[t]$ se $t < 0$:

$$\lfloor t \rfloor = \begin{cases} [t] & \text{se } t \geqslant 0 \\ [t] - 1 & \text{se } t < 0 \end{cases}$$

Esempio

$x_0 = 1.3,\ x_1 = 1.6,\ x_2 = 1.9$

$$\begin{array}{llllll} f_0 = & 0.6200860 & f_1 = & 0.4554022 & f_2 = & 0.2818186 \\ f'_0 = & -0.5220232 & f'_1 = & -0.5698959 & f'_2 = & -0.5811571 \end{array}$$

Polinomio di Hermite di grado 5:

1.3	0.6200860					
1.3	0.6200860	−0.5220232				
1.6	0.4554022	−0.5489460	−0.0897427			
1.6	0.4554022	−0.5698959	−0.0698330	0.0663657		
1.9	0.2818186	−0.5786120	−0.0290537	0.0679655	0.0026663	
1.9	0.2818186	−0.5811571	−0.0084837	0.0685667	0.0010020	−0.0027738

$$\begin{aligned} p_5(1.5) = & \ 0.6200860 \\ & - 0.5220232(1.5-1.3) \\ & - 0.0897427(1.5-1.3)^2 \\ & + 0.0663657(1.5-1.3)^2(1.5-1.6) \\ & + 0.0026663(1.5-1.3)^2(1.5-1.6)^2 \\ & - 0.0027738(1.5-1.3)^2(1.5-1.6)^2(1.5-1.9) \\ = & \ 0.5118277 \end{aligned}$$

Condizionamento del problema dell'interpolazione polinomiale

Siano dati $x_0, x_1, \ldots, x_n$ punti distinti di $[a, b]$ e $y_0, y_1, \ldots, y_n$ valori assunti da $f(x)$ in $x = x_i$.
Sia $p_n(x)$ il polinomio di Lagrange di grado n per $f(x)$ relativo ai nodi $x_0, x_1, \ldots, x_n$ e $\tilde{p}_n(x)$ quello relativo agli stessi nodi e a valori $\tilde{y}_0, \ldots, \tilde{y}_n$.

$$\tilde{p}_n(x) - p_n(x) = \sum_{i=0}^{n} L_i(x)(\tilde{y}_i - y_i) \qquad \epsilon_i = \tilde{y}_i - y_i$$

$$\left|\tilde{p}_n(x) - p_n(x)\right| \leqslant \sum_{i=0}^{n} |L_i(x)|\,|\tilde{y}_i - y_i| \leqslant \max_{i=0,\ldots,n} |\tilde{y}_i - y_i| \sum_{i=0}^{n} |L_i(x)|$$

$$= \|\tilde{y} - y\|_\infty \sum_{i=0}^{n} |L_i(x)|$$

$\sum_{i=0}^{n} |L_i(x)|$ è la funzione di Lebesgue. Sia $\Lambda_n = \max_{x\in[a,b]} \left\{\sum_{i=0}^{n} |L_i(x)|\right\} = \|L\|_\infty$.
Facendo il massimo di ambo i membri per gli $x \in [a, b]$, si trova:

$$\max_{x\in[a,b]} |\tilde{p}_n(x) - p_n(x)| = \left\|\tilde{p}_n(x) - p_n(x)\right\|_\infty \leqslant \Lambda_n \left\|\tilde{y} - y\right\|_\infty$$

La **costante di Lebesgue Λ_n** è un **indicatore** del condizionamento del problema.

Condizionamento del problema dell'interpolazione polinomiale

Si ha:

$$\|p_n(x)\|_\infty = \max_{[a,b]} |p_n(x)| \geqslant \max_{x_i, i=0,\ldots,n} |p_n(x_i)| = \max_{i=0,\ldots,n} |y_i| = \|y\|_\infty$$

$$\Rightarrow \quad \frac{\|\tilde{p}_n(x) - p_n(x)\|_\infty}{\|p_n(x)\|_\infty} \leqslant \Lambda_n \frac{\|\tilde{y} - y\|_\infty}{\|y\|_\infty}$$

- Se $a \leqslant x_0 < x_1 < \ldots < x_n \leqslant b$, allora $\Lambda_n \leqslant 2^n (h/h^*)^n$, con $h = \max_{i=0,\ldots,n-1}(x_{i+1} - x_i)$, $h^* = \min_{i=0,\ldots,n-1}(x_{i+1} - x_i)$ e dunque $h/h^* > 1$.
- Se $x_i = x_0 + i(b-a)/n$, per $n \to \infty$ si ha $\Lambda_n \approx 2^n / (n \ln(n))$: dunque, con nodi equidistanti, per n grande l'interpolazione polinomiale è un problema mal condizionato.
- Se come nodi sono scelti gli zeri dei polinomi di Chebyshev di grado crescente, ossia

$$x_k = \cos\left(\frac{2k+1}{2(n+1)}\pi\right)\frac{(b-a)}{2} + \frac{a+b}{2} \qquad k = 0, \ldots, n$$

  allora per $n \gg$ si ha $\Lambda_n \approx (2/\pi)\ln(n)$.

Ricordiamo che, per i nodi di Chebyshev, si ha anche che la quantità

$\max_{x\in[a,b]} |\omega(x)| = \|\omega(x)\|_\infty = \omega^*$ è minima possibile e vale $2((b-a)/4)^{n+1}$.

In definitiva, con i nodi di Chebyshev il mal condizionamento è più contenuto.

Matrice di interpolazione

La matrice di interpolazione è data da:

$$\begin{pmatrix} x_0^{(0)} & & & & \\ x_0^{(1)} & x_1^{(1)} & & & \\ x_0^{(2)} & x_1^{(2)} & x_2^{(2)} & & \\ \vdots & \vdots & \vdots & \ddots & \\ x_0^{(n)} & x_1^{(n)} & x_2^{(n)} & \ldots & x_n^{(n)} \end{pmatrix}$$

Data $f(x)$ con $x \in [a,b]$ e data la successione dei polinomi di interpolazione $\{p_n(x)\}$ costruiti a partire da una riga $\{x_i^{(n)}\}_{i=0,\ldots,n}$ della matrice, vale il seguente teorema:

Teorema di Faber

Per ogni matrice di interpolazione esiste una funzione $f \in C^0([a,b])$ per cui la successione $\{p_n(x)\}$ non converge uniformemente a $f(x)$.

Il fenomeno di Runge

Esempio (fenomeno di Runge).

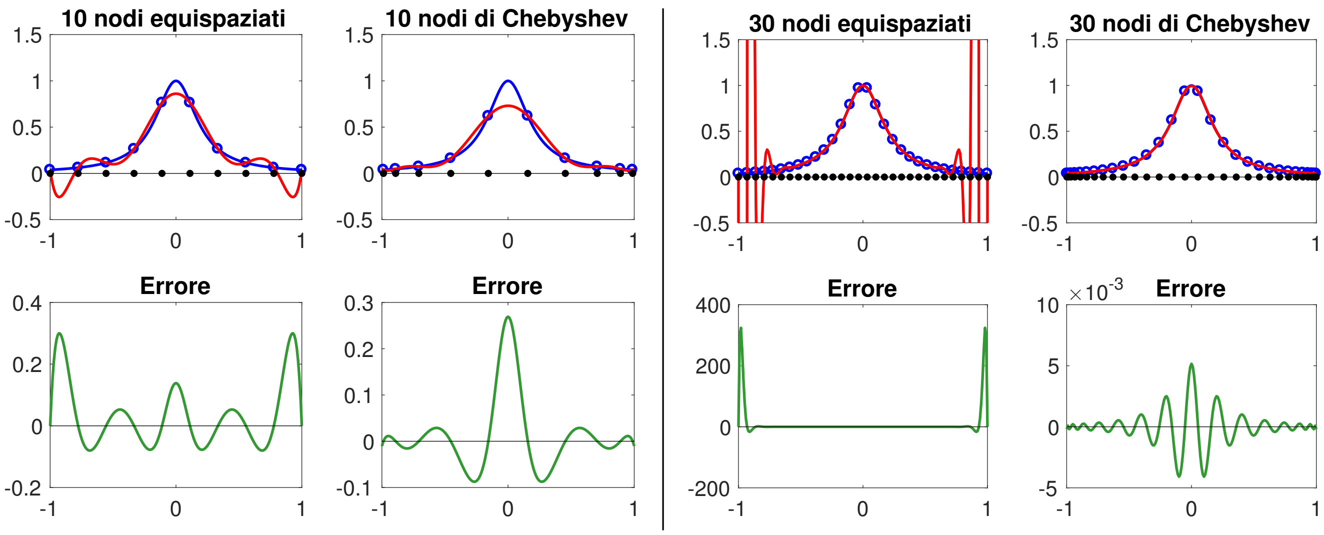
Siano $[a, b] = [-1, 1]$ e

$$f(x) = \frac{1}{1 + 25x^2}$$

Se si considera $x_i^{(n)} = -1 + i\dfrac{b-a}{n}$, la successione $\{p_n(x)\}$ non converge a $f(x)$.
Già per $n = 20$ negli estremi ci sono forti oscillazioni. Solo al centro dell'intervallo l'approssimazione è buona.

Se $x_i^{(n)} = \cos\left(\dfrac{2i+1}{2(n+1)}\pi\right)$, allora $\{p_n(x)\}$ converge **uniformemente** a $f(x)$.

Il fenomeno di Runge



Legenda: funzione di Runge, polinomio interpolante, funzione errore, ○ punti di interpolazione, • nodi di interpolazione

	nodi di interpol.	
nodi di tabulazione: 200	10	30
$\|$ errore $\|_\infty$ con nodi equidistanti	0.300	324.238
$\|$ errore $\|_\infty$ con nodi di Chebyshev	0.269	0.005

Convergenza uniforme

Teorema di Bernstein

Sia $f \in C^1([a,b])$. Se $\{p_n(x)\}$ è la successione di polinomi di interpolazione a partire dagli zeri di una successione di polinomi di Chebyshev, allora

$$\|f - p_n\|_\infty \xrightarrow[n \to +\infty]{} 0 \quad \text{uniformemente in } [a,b]$$

ossia $\{p_n(x)\}$ converge uniformemente a $f(x)$.
Se $f \in C^2([a,b])$, allora $\|f - p_n\|_\infty = \mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$.

Teorema di Hermite-Féjèr

Sia $f \in C^0([a,b])$. Se $\{p_n(x)\}$ è la successione dei polinomi di Hermite costruiti su nodi di Chebyshev e tali che $p_{2n+1}(x_i) = f(x_i)$, $p'_{2n+1}(x_i) = 0$, $i = 0, 1, \dots, n$, allora

$$\|f - p_n\|_\infty \xrightarrow[n \to +\infty]{} 0 \text{ uniformemente in } [a,b]$$

Funzioni spline

Dato l'intervallo $[a,b]$, si consideri una successione finita di numeri reali (nodi) appartenenti all'intervallo, tali che $a \leqslant x_0 < x_1 < x_2 < \ldots < x_m < x_{m+1} \leqslant b$ (a e b possono assumere i valori $-\infty$ e $+\infty$). Si individua in tal modo una partizione dell'intervallo $[a,b]$ in $m+1$ sottointervalli $\mathcal{I}_i = [x_i, x_{i+1}[$, $\mathcal{I}_m = [x_m, x_{m+1}]$.

Definizione

Si dice funzione spline di grado n, o di ordine $n+1$, relativa alla partizione $\{x_i\}_{i=0,\dots,m+1}$ di $[a,b]$, una funzione $s(x)$ che soddisfa le seguenti due proprietà:

1. $s(x)$ è un polinomio $s_i(x)$ di grado non superiore a n in ciascun sottointervallo $\mathcal{I}_i$ della partizione, $i = 0, \dots, m$;
2. $s(x) \in C^{n-1}([a,b])$, ossia la funzione e le sue derivate fino all'ordine $n-1$ sono continue sull'intervallo $[a,b]$; ciò significa che per ogni nodo interno alla partizione valgono le seguenti mn condizioni:

$$s_i^{(k)}(x_{i+1}) = s_{i+1}^{(k)}(x_{i+1}), \qquad i = 0, \dots, m-1; k = 0, 1, \dots, n-1$$

Funzioni spline

In altre parole, una spline $s(x)$ entro ciascun intervallo $\mathcal{I}_i$, $i = 0, \ldots, m$, è un polinomio di grado al più n che in ogni punto interno all'intervallo è C^∞ e agli estremi coincide con il polinomio relativo all'intervallo precedente (se esiste) e con quello dell'intervallo successivo (se esiste) fino alla derivata $(n-1)$-esima.
L'insieme delle spline di grado n relative alla partizione $\{x_i\}_{i=0,\ldots,m+1}$ si denota con $\mathcal{S}_n\{x_1, \ldots, x_m\}$. Tale insieme contiene l'insieme dei polinomi di grado non superiore ad n.

Si osserva che la somma di funzioni spline è ancora una funzione spline e che il prodotto di uno scalare reale per una funzione spline è ancora una funzione spline. Pertanto **l'insieme delle spline $\mathcal{S}_n\{x_1, \ldots, x_m\}$ è uno spazio funzionale lineare**.

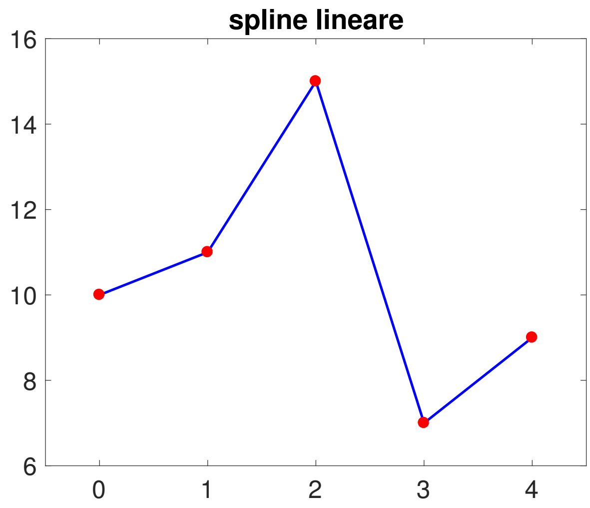
Inoltre, la derivata di una spline di grado n è una spline di grado $n-1$ relativa alla medesima partizione e l'integrale di una spline di grado n è una spline di grado $n+1$ relativa alla medesima partizione.

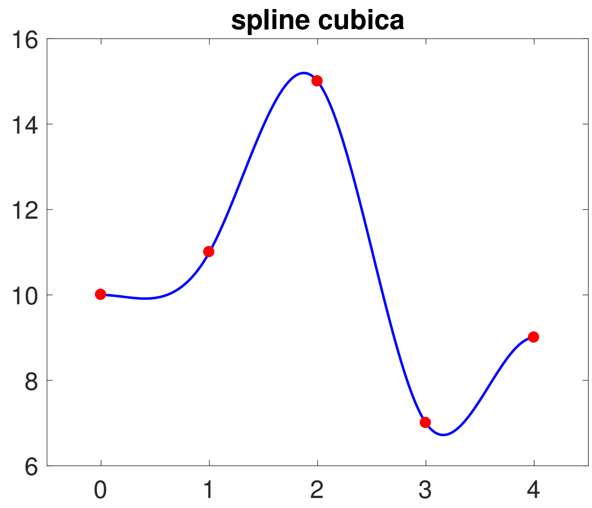
Ogni spline dipende da $(n+1)(m+1)$ parametri, che devono soddisfare nm condizioni nei nodi interni (uguaglianza dei valori della funzione e delle derivate fino all'ordine $n-1$). Pertanto ogni spline dipende da $m+n+1$ parametri.

Esempi

Spline di grado 1 (o lineare): $s_i(x)$ è il segmento che unisce (x_i, y_i) con (x_{i+1}, y_{i+1}) e vale che $s_i(x_{i+1}) = s_{i+1}(x_{i+1})$. E' una funzione di classe C^0 e dipende da $2(m+1) - m = m+2$ parametri.

Spline di grado 3 o cubica: $s_i(x)$ è un polinomio di grado al più 3 in $[x_i, x_{i+1}]$ e vale che $s_i(x_{i+1}) = s_{i+1}(x_{i+1})$, $s_i'(x_{i+1}) = s_{i+1}'(x_{i+1})$, $s_i''(x_{i+1}) = s_{i+1}''(x_{i+1})$. Dipende da $4(m+1) - 3m = m+4$ parametri.





Proprietà delle spline

Teorema

Lo spazio delle funzioni spline $\mathcal{S}_n\{x_1, \dots, x_m\}$ è uno spazio lineare di dimensione $m + n + 1$.

Definizione

Una funzione spline di grado n relativa ai nodi $x_1, \dots, x_{m+1}$ si dice **periodica** di periodo $x_{m+1} - x_1$ se è una spline che soddisfa le ulteriori n condizioni:

$$s^{(k)}(x_1) = s^{(k)}(x_{m+1}), \qquad k = 0, \dots, n-1.$$

Lo spazio delle funzioni spline periodiche è uno spazio lineare di dimensione m (perché $(n+1)m - mn = m$).

Spline naturali

Definizione

Una funzione spline di grado dispari $n = 2k - 1$, $k \in \mathbb{N}$, relativa ai nodi $x_0, x_1, \dots, x_{m+1}$, si dice **naturale** se è una spline che negli intervalli $[x_0, x_1]$ e $[x_m, x_{m+1}]$ diventa un polinomio di grado $k - 1$.

Di conseguenza, $s^{(j)}(x_0) = s^{(j)}(x_{m+1}) = 0\ \forall j = k, k+1, \dots, 2k-2$.

Per esempio, una spline cubica è naturale se nel primo e nell'ultimo intervallo è un segmento: in tal caso infatti $s''(x_0) = s''(x_{m+1}) = 0$.

In questo caso i parametri da cui dipende la spline diventano:

$$m + n + 1 - (2k - 2) = m + 2k - 1 + 1 - 2k + 2 = m + 2$$

ossia il numero dei parametri coincide con il numero totale di nodi della suddivisione.

Interpolazione con Spline Lineari

Teorema

Dati $m+2$ nodi distinti in $[a,b]$, denotati con $x_0, x_1, \ldots, x_{m+1}$, tali che $x_0 \geqslant a$, $x_{m+1} \leqslant b$ e $x_i < x_{i+1}$ $\forall i = 0, \ldots, m$, e assegnati $y_0, \ldots, y_{m+1}$, esiste una e una sola spline lineare $s(x)$ tale che $s(x_i) = y_i$ $\forall i = 0, \ldots, m+1$. Essa è

$$s(x) = \sum_{i=0}^{m+1} y_i \ell_i(x)$$

con

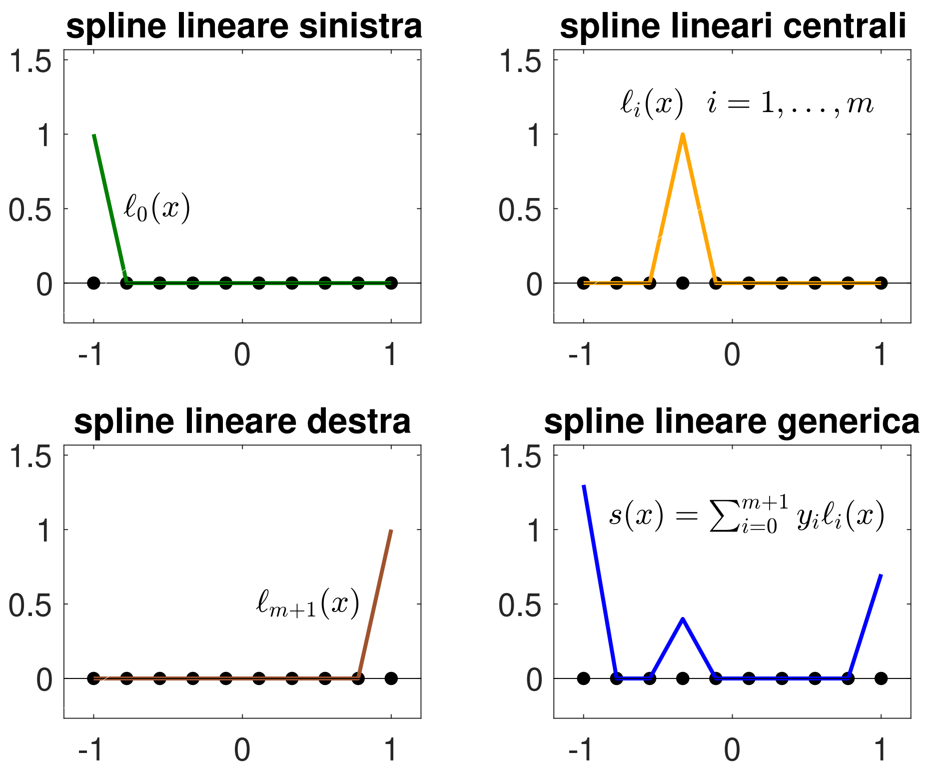
$$\ell_i(x) = \begin{cases} \dfrac{x - x_{i-1}}{x_i - x_{i-1}} & \text{per } x \in [x_{i-1}, x_i] \\ \dfrac{x - x_{i+1}}{x_i - x_{i+1}} & \text{per } x \in [x_i, x_{i+1}] \\ 0 & \text{altrove} \end{cases} \qquad \forall i = 1, \ldots, m$$

per $i = 1, \ldots, m$ e

$$\ell_0(x) = \begin{cases} \dfrac{x - x_1}{x_0 - x_1} & \text{per } x \in [x_0, x_1] \\ 0 & \text{altrove} \end{cases} \qquad \ell_{m+1}(x) = \begin{cases} \dfrac{x - x_m}{x_{m+1} - x_m} & \text{per } x \in [x_m, x_{m+1}] \\ 0 & \text{altrove} \end{cases}$$

Qui m rappresenta il numero dei **nodi interni**.

Interpolazione con Spline Lineari



Esempio di generica spline lineare su nodi equidistanti, costruita con spline semplici.

Interpolazione con Spline Lineari

Dim. Detta $s_i(x)$ la spline in $[x_i, x_{i+1}]$, deve essere $s_i(x) = p^{(i)}x + q^{(i)}$, $i = 0, \ldots, m$.
Inoltre deve essere $s(x_i) = y_i$, $s(x_{i+1}) = y_{i+1}$, da cui

$$\begin{cases} p^{(i)}x_i + q^{(i)} = y_i \\ p^{(i)}x_{i+1} + q^{(i)} = y_{i+1} \end{cases}$$

e poiché $\begin{vmatrix} x_i & 1 \\ x_{i+1} & 1 \end{vmatrix} \neq 0$ il sistema ammette una e una sola soluzione

$$\begin{aligned} s_i(x) &= y_i \frac{x - x_{i+1}}{x_i - x_{i+1}} + y_{i+1} \frac{x - x_i}{x_{i+1} - x_i} = \frac{y_i(x - x_{i+1}) - y_{i+1}(x - x_i)}{x_i - x_{i+1}} \\ &= y_i \ell_i(x) + y_{i+1}\ell_{i+1}(x) \end{aligned}$$

per $x \in [x_i, x_{i+1}]$. In tale intervallo

$$y_0\ell_0(x) + \ldots + y_{i-1}\ell_{i-1}(x) + y_{i+2}\ell_{i+2}(x) + \ldots + y_{m+1}\ell_{m+1}(x) = 0$$

da cui $s(x) = \sum_{i=0}^{m+1} y_i \ell_i(x)$.

Analisi dell'errore

In $[x_i, x_{i+1}]$, l'errore commesso è pari a quello di interpolazione con un polinomio di grado 1.
Se $f \in C^2([a, b])$,

$$f(x) - s_i(x) = \frac{f''(\xi_i)}{2!}(x - x_i)(x - x_{i+1}),$$

per $x \in [x_i, x_{i+1}]$, $\quad x_i < \xi < x_{i+1}$.
Se $|f''(x)| \leqslant M$ per $x \in [a, b]$, segue

$$|f(x) - s_i(x)| \leqslant \frac{M}{2} \max |(x - x_i)(x - x_{i+1})| = \frac{M}{2} \frac{(x_{i+1} - x_i)^2}{4}$$

Posto $h = \max_{i=0,\ldots,m}(x_{i+1} - x_i)$ si ha

$$\max_{x \in [a,b]} |f(x) - s(x)| = \|f(x) - s(x)\|_\infty \leqslant \frac{M}{8} h^2$$

La complessità del metodo è di 3 prodotti e 4 addizioni.

Codice Matlab

Calcolo di una spline lineare interpolante relativa a $\boldsymbol{x}$ e $\boldsymbol{y}$ nei punti $\boldsymbol{xx}$.

```matlab
function [yy] = splineLineare(x, y, xx)
% SPLINELINEARE - Valuta in xx la spline lineare interpolante x e y
%  x  (double array) - vettore dei nodi (ordinati in modo crescente)
%  y  (double array) - vettore delle osservazioni
%  xx (double array) - vettore in cui valutare la spline lineare
%  yy (double array) - vettore dei valori della spline nei punti xx
%
  x = x(:); y = y(:); xx = xx(:);
  for i = 1 : length(xx)
    if ( xx(i) < x(1) || xx(i) > x(end) )
      error('punto non interno all''intervallo di interpolazione');
    end
    k = min( find( abs(x - xx(i)) < eps ) );
    if ( ~isempty(k) )
      yy(i) = y( k );
    else
      k     = min( find( x > xx(i) ) );
      yy(i) = ( y(k-1) * (xx(i)-x(k)) - y(k) * (xx(i)-x(k-1)) ) ...
              / ( x(k-1) - x(k) );
    end
  end
end
```


Interpolazione di Hermite tramite polinomi cubici a tratti

Assegnati i nodi $a \leqslant x_0 < x_1 < \ldots < x_m < x_{m+1} \leqslant b$, si vuol determinare una funzione $g(x)$ che in ogni intervallo $[x_i, x_{i+1}]$, $i = 0, \ldots, m$, è un polinomio cubico e tale che, se $\tilde{s}_i(x)$ rappresenta tale funzione in $[x_i, x_{i+1}]$, valga che

$$\begin{cases} \tilde{s}_i(x_i) = y_i, & \tilde{s}_i(x_{i+1}) = y_{i+1} \\ \tilde{s}_i'(x_i) = z_i, & \tilde{s}_i'(x_{i+1}) = z_{i+1} \end{cases} \qquad i = 0, \ldots, m$$

Si assume che $y_0, y_1, \ldots, y_{m+1}$ e $z_0, z_1, \ldots, z_{m+1}$ siano assegnati.
Dall'interpolazione di Hermite, si ha

x_i	y_i			
x_i	y_i	z_i		
x_{i+1}	y_{i+1}	$f[x_i, x_{i+1}] = \dfrac{y_{i+1} - y_i}{x_{i+1} - x_i}$	$\dfrac{f[x_i, x_{i+1}] - z_i}{x_{i+1} - x_i}$	
x_{i+1}	y_{i+1}	z_{i+1}	$\dfrac{z_{i+1} - f[x_i, x_{i+1}]}{x_{i+1} - x_i}$	$\dfrac{z_{i+1} + z_i - 2f[x_i, x_{i+1}]}{(x_{i+1} - x_i)^2}$

Interpolazione di Hermite tramite polinomi cubici a tratti

Assegnati i nodi $a \leqslant x_0 < x_1 < \ldots < x_m < x_{m+1} \leqslant b$, si vuol determinare una funzione $g(x)$ che in ogni intervallo $[x_i, x_{i+1}]$, $i = 0, \ldots, m$, è un polinomio cubico e tale che, se $\widetilde{s}_i(x)$ rappresenta tale funzione in $[x_i, x_{i+1}]$, valga che

$$\begin{cases} \widetilde{s}_i(x_i) = y_i, & \widetilde{s}_i(x_{i+1}) = y_{i+1} \\ \widetilde{s}_i'(x_i) = z_i, & \widetilde{s}_i'(x_{i+1}) = z_{i+1} \end{cases} \qquad i = 0, \ldots, m$$

Si assume che $y_0, y_1, \ldots, y_{m+1}$ e $z_0, z_1, \ldots, z_{m+1}$ siano assegnati.
Dall'interpolazione di Hermite, si ha

$$\begin{array}{c|c|c|c|c}
x_i & y_i & & & \\
x_i & y_i & z_i & & \\
x_{i+1} & y_{i+1} & f[x_i, x_{i+1}] = \dfrac{y_{i+1} - y_i}{x_{i+1} - x_i} & \dfrac{f[x_i, x_{i+1}] - z_i}{x_{i+1} - x_i} & \\
x_{i+1} & y_{i+1} & z_{i+1} & \dfrac{z_{i+1} - f[x_i, x_{i+1}]}{x_{i+1} - x_i} & \dfrac{z_{i+1} + z_i - 2f[x_i, x_{i+1}]}{(x_{i+1} - x_i)^2}
\end{array}$$

Posto $h_i = x_{i+1} - x_i$, $i = 0, 1, \ldots, m$, si ha

$$\widetilde{s}_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^2(x - x_{i+1})$$

$$a_i = y_i\,, \quad b_i = z_i\,, \quad c_i = \frac{f[x_i, x_{i+1}] - z_i}{h_i}\,, \quad d_i = \frac{z_{i+1} + z_i - 2f[x_i, x_{i+1}]}{h_i^2}$$

Interpolazione di Bessel cubica a tratti

Si approssimano i valori delle derivate prime nei nodi con la formula

$$z_i = \frac{h_{i-1} f[x_i, x_{i+1}] + h_i f[x_{i-1}, x_i]}{h_i + h_{i-1}}$$

dove si aggiungono due nodi

$$x_{-1} = x_0 - h_0$$
$$x_{m+2} = x_{m+1} + h_m$$

È un modo per approssimare le derivate prime di una funzione incognita nei nodi.

Il polinomio si valuta in ξ con lo schema di Horner. Se $\xi \in (x_i, x_{i+1})$ e a_i, b_i, c_i, d_i sono i coefficienti dell'interpolante in tale i-esimo intervallo, $i = 0, \ldots, m$, allora

$$p \leftarrow d_i \cdot (\xi - x_{i+1}) + c_i$$
$$p \leftarrow p \cdot (\xi - x_i) + b_i$$
$$p \leftarrow p \cdot (\xi - x_i) + a_i$$

Infatti

$$\widetilde{s}_i(x) = a_i + (x - x_i)\Big(b_i + (x - x_i)\big(c_i + (x - x_{i+1})d_i\big)\Big)$$

Osservazione

L'interpolazione mediante spline lineare determina una funzione di approssimazione non derivabile nei nodi x_i, $i = 1, \ldots, m$.

Nelle applicazioni, spesso si richiede che l'approssimazione sia derivabile. Dunque si preferisce interpolare in ogni intervallo $[x_i, x_{i+1}]$ con polinomi di Hermite di grado 3, essendo noti $f(x_i)$, $f(x_{i+1})$, $f'(x_i)$, $f'(x_{i+1})$, ottenendo una funzione continua e derivabile.

Tuttavia in questo caso è necessario conoscere $f'(x_i)$, $i = 0, \ldots, m+1$, il che non è sempre possibile.

Ci si pone il problema di trovare un'approssimante data da un polinomio di grado 3 a tratti, che sia almeno continuo e derivabile senza conoscere i valori delle derivate prime nei nodi x_i.

Il problema è risolto dall'interpolazione mediante spline cubiche interpolanti.

Interpolazione con spline cubiche

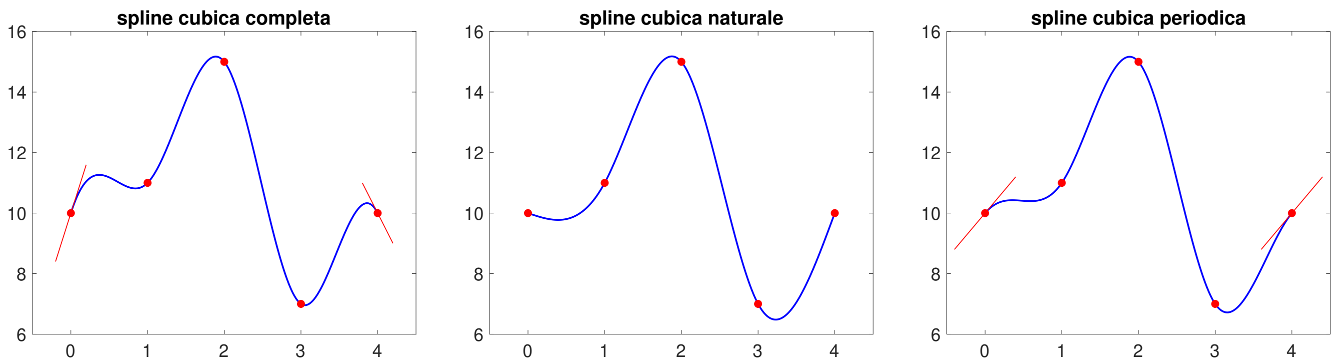
Teorema

Sia data $\{x_i\}_{i=0,\ldots,m+1}$ una partizione dell'intervallo chiuso e limitato $[a, b]$ con $a \leqslant x_0 < x_1 < \ldots < x_m < x_{m+1} \leqslant b$ e siano assegnati $y_0, y_1, \ldots, y_{m+1}$.
Esiste una ed una sola spline cubica interpolante $s(x) \in \mathcal{S}_3\{x_1, \ldots, x_m\}$ tale che $s(x_i) = y_i$, $i = 0, 1, \ldots, m+1$, e tale che valga una delle seguenti condizioni:

a) $s'(x_0) = z_0$, $s'(x_{m+1}) = z_{m+1}$, con z_0, z_{m+1} assegnati (**spline completa**)

b) $s''(x_0) = s''(x_{m+1}) = 0$ (**spline naturale**)

c) $y_0 = y_{m+1} = s(x_0) = s(x_{m+1}), s'(x_0) = s'(x_{m+1}), s''(x_0) = s''(x_{m+1})$ (**spline periodica** di periodo $x_{m+1} - x_0$)



Costruzione della spline cubica interpolante

Per vedere che quanto affermato dal teorema è vero, sia $s_i(x)$ il polinomio di grado 3 definito in $[x_i, x_{i+1}]$, per ogni $i = 0, 1, \ldots, m$, e sia $h_i = x_{i+1} - x_i$. Allora:

$$s_i(x) = \alpha_i + \beta_i(x - x_i) + \gamma_i(x - x_i)^2 + \delta_i(x - x_i)^3$$

Di conseguenza, si ha

$$\begin{aligned} s_i'(x) &= \beta_i + 2\gamma_i(x - x_i) + 3\delta_i(x - x_i)^2 \\ s_i''(x) &= 2\gamma_i + 6\delta_i(x - x_i) \end{aligned}$$

Affinché $s_i(x)$ e $s_{i+1}(x)$ siano parti di una spline cubica, in ogni nodo di raccordo x_{i+1} deve valere che

$$\left.\begin{aligned} s_i(x_{i+1}) &= s_{i+1}(x_{i+1}) \\ s_i'(x_{i+1}) &= s_{i+1}'(x_{i+1}) \\ s_i''(x_{i+1}) &= s_{i+1}''(x_{i+1}) \end{aligned}\right\} \; i = 0, \ldots, m-1$$

Se sono noti i valori nei nodi di y_i e di z_i, $\forall i = 0, \ldots, m+1$, allora $s_i(x)$ è completamente determinata, come si è già visto:

$$\left.\begin{aligned} s_i(x_i) &= \alpha_i = y_i \\ s_i'(x_i) &= \beta_i = z_i \end{aligned}\right\} \; i = 0, \ldots, m+1$$

Costruzione della spline cubica interpolante

Infatti vale che

$$\begin{aligned} s_i(x) &= \alpha_i + \beta_i(x - x_i) + \gamma_i(x - x_i)^2 + \delta_i(x - x_i)^3 \\ &= a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^2(x - x_{i+1}) \\ &= a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 - d_i h_i(x - x_i)^2 \end{aligned}$$

con

$$a_i = y_i\,, \quad b_i = z_i\,, \quad c_i = \frac{f[x_i, x_{i+1}] - z_i}{h_i}\,, \quad d_i = \frac{z_{i+1} + z_i - 2f[x_i, x_{i+1}]}{h_i^2}$$

Poiché $x - x_{i+1} = x - x_i + x_i - x_{i+1} = x - x_i - h_i$, vale che

$$\begin{aligned} \alpha_i &= a_i = y_i \\ \beta_i &= b_i = z_i \\ \gamma_i &= c_i - d_i h_i = \frac{f[x_i, x_{i+1}] - z_i}{h_i} - d_i h_i \\ \delta_i &= d_i = \frac{z_{i+1} + z_i - 2f[x_i, x_{i+1}]}{h_i^2} \end{aligned}$$

Allora per conoscere $s_i(x)$ basta determinare z_i, $i = 0, \ldots, m+1$.

Costruzione della spline cubica interpolante

Si ricorda che $s_i''(x) = 2\gamma_i + 6\delta_i(x - x_i)$ e che valgono le seguenti relazioni:

$$\left.\begin{aligned} s_i''(x_{i+1}) &= s_{i+1}''(x_{i+1}) \\ 2\gamma_i + 6\delta_i h_i &= 2\gamma_{i+1} \\ c_i - d_i h_i + 3 d_i h_i &= c_{i+1} - d_{i+1} h_{i+1} \end{aligned}\right\} \quad i = 0, 1, \ldots, m-1$$

Sostituendo nell'ultima relazione le espressioni di c_i e d_i si ottiene:

$$\frac{f[x_i, x_{i+1}] - z_i}{h_i} + 2\frac{z_i + z_{i+1} - 2f[x_i, x_{i+1}]}{h_i} = \frac{f[x_{i+1}, x_{i+2}] - z_{i+1}}{h_{i+1}} - \frac{z_{i+1} + z_{i+2} - 2f[x_{i+1}, x_{i+2}]}{h_{i+1}}$$

$$\frac{z_i}{h_i} + 2z_{i+1}\left(\frac{1}{h_i} + \frac{1}{h_{i+1}}\right) + \frac{z_{i+2}}{h_{i+1}} = \frac{3}{h_i} f[x_i, x_{i+1}] + \frac{3}{h_{i+1}} f[x_{i+1}, x_{i+2}]$$

Facendo il comune denominatore, si ottiene un **sistema di m equazioni nelle $m+2$ incognite** $z_0, z_1, \ldots, z_{m+1}$:

$$h_{i+1} z_i + 2(h_i + h_{i+1}) z_{i+1} + h_i z_{i+2} = 3\Big(h_{i+1} f[x_i, x_{i+1}] + h_i f[x_{i+1}, x_{i+2}]\Big) = r_{i+1}$$
$$i = 0, \ldots, m-1$$

Costruzione della spline cubica interpolante

Caso a: z_0, z_{m+1} assegnati (spline completa)

Le equazioni diventano

$$\begin{cases} 2(h_0 + h_1) z_1 + h_0 z_2 = r_1 - h_1 z_0 \\ h_2 z_1 + 2(h_1 + h_2) z_2 + h_1 z_3 = r_2 \\ \quad\vdots \\ h_{m-1} z_{m-2} + 2(h_{m-2} + h_{m-1}) z_{m-1} + h_{m-2} z_m = r_{m-1} \\ h_m z_{m-1} + 2(h_{m-1} + h_m) z_m = r_m - h_{m-1} z_{m+1} \end{cases}$$

$$\boxed{T\boldsymbol{z} = \boldsymbol{r}} \qquad \text{con} \quad \boldsymbol{z} = (z_1, \ldots, z_m)^T$$

$$T = \begin{pmatrix} 2(h_0 + h_1) & h_0 & & & \\ h_2 & 2(h_1 + h_2) & h_1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & h_{m-2} \\ & & & h_m & 2(h_{m-1} + h_m) \end{pmatrix}, \quad \boldsymbol{r} = \begin{pmatrix} -h_1 z_0 + r_1 \\ r_2 \\ \vdots \\ r_{m-1} \\ -h_{m-1} z_{m+1} + r_m \end{pmatrix}$$

T è di ordine m, strettamente diagonale dominante e quindi non singolare $\Rightarrow$ esiste una e una sola spline cubica interpolante.

Caso b: $s''(x_0) = s''(x_{m+1}) = 0$ (spline naturale)

$$s''(x_0) = 0 \;\Rightarrow\; 2\gamma_0 = 0 \qquad\qquad s''(x_{m+1}) = 0 \;\Rightarrow\; 2\gamma_m + 6\delta_m h_m = 0$$

$$\frac{f[x_0,x_1] - z_0}{h_0} - \frac{z_0 + z_1 - 2f[x_0,x_1]}{h_0} = 0$$

$$\frac{f[x_m,x_{m+1}] - z_m}{h_m} + 2\frac{z_m + z_{m+1} - 2f[x_m,x_{m+1}]}{h_m} = 0$$

Allora

$$2z_0 + z_1 = 3f[x_0,x_1] \qquad\qquad z_m + 2z_{m+1} = 3f[x_m,x_{m+1}]$$

$$\boxed{T\boldsymbol{z} = \boldsymbol{r}} \qquad \text{con} \quad \boldsymbol{z} = (z_0, z_1, \ldots, z_m, z_{m+1})^T$$

$$T = \begin{pmatrix} 2 & 1 & & & & \\ h_1 & 2(h_0+h_1) & h_0 & & & \\ & \ddots & \ddots & \ddots & & \\ & & h_m & 2(h_{m-1}+h_m) & h_{m-1} & \\ & & & 1 & 2 & \end{pmatrix}, \quad \boldsymbol{r} = \begin{pmatrix} 3f[x_0,x_1] \\ r_1 \\ \vdots \\ r_m \\ 3f[x_m,x_{m+1}] \end{pmatrix}$$

T è di ordine $m+2$, strettamente diagonale dominante e dunque non singolare
$\Rightarrow$ esiste una e una sola spline naturale.

Caso c: $s(x_0) = s(x_{m+1}), s'(x_0) = s'(x_{m+1}), s''(x_0) = s''(x_{m+1})$ (spline periodica)

$$y_0 = y_{m+1} \qquad\qquad z_0 = s'(x_0) = s'(x_{m+1}) = z_{m+1}$$

$$s''(x_0) = s''(x_{m+1}) \quad\Rightarrow\quad \gamma_0 = \gamma_m + 3\delta_m h_m$$

$$\frac{f[x_0,x_1] - z_0}{h_0} - \frac{z_0 + z_1 - 2f[x_0,x_1]}{h_0} = \frac{f[x_m,x_{m+1}] - z_m}{h_m} + 2\frac{z_m + z_{m+1} - 2f[x_m,x_{m+1}]}{h_m}$$

$$2(h_0 + h_m)z_0 + h_m z_1 + h_0 z_m = \underbrace{3h_0 f[x_m,x_{m+1}] + 3h_m f[x_0,x_1]}_{= r_0}$$

$$\boxed{T\boldsymbol{z} = \boldsymbol{r}} \qquad \text{con} \quad \boldsymbol{z} = (z_0, z_1, \ldots, z_m)^T$$

$$T = \begin{pmatrix} 2(h_m+h_0) & h_m & 0 & 0 & h_0 \\ h_1 & 2(h_0+h_1) & h_0 & & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & & \ddots & \ddots & h_{m-2} \\ h_{m-1} & 0 & 0 & h_m & 2(h_{m-1}+h_m) \end{pmatrix}, \quad \boldsymbol{r} = \begin{pmatrix} r_0 \\ r_1 \\ \vdots \\ r_{m-1} \\ r_m \end{pmatrix}$$

T è di ordine $m+1$ ed è strettamente diagonale dominante, dunque non singolare
$\Rightarrow$ esiste una sola spline periodica.

Codice Matlab per spline cubiche complete

```
function [C] = mySplineCompleta(x, y, z0, zf)
% MYSPLINECOMPLETA - Coeff. spline cubica completa interpolante
% INPUT
%  x   (double array) - vettore (ORDINATO) dei nodi di interpolazione
%  y   (double array) - vettore dei valori della funzione nei nodi
%  z0  (double)       - valore della derivata prima nel nodo iniziale
%  zf  (double)       - valore della derivata prima nel nodo finale
% OUPUT
%  C   (double array) - matrice con tante righe quanti sono i
%                       sottointervalli e 4 colonne, contenenti i
%                       coeff. dei polinomi di grado 3, da quello
%                       della potenza 3 a quello della potenza 0
%
%  C(i,1)*(x-x_i)^3 + C(i,2)*(x-x_i)^2 + C(i,3)*(x-x_i) + C(i,4)
%
%  Rappresentazione 'pp' (piecewise polynomial)
%
  x = x(:); y = y(:);
  m   = length(x); mm1 = m - 1; mm2 = m - 2;  % m = num. totale nodi
  h   = diff(x); % m-1 valori di h
  d0  = 2 * ( h(1:mm2) + h(2:mm1) ); % diagonale principale
  d1  = [0; h(1:(m-3))]; % prima sopradiagonale della matrice
  dm1 = [h(3:(mm1)); 0]; % prima sottodiagonale della matrice
```


Codice Matlab per spline cubiche complete

```
  T   = spdiags([dm1 d0 d1], [-1 0 1], mm2, mm2); % ordine m-2
  r   = ( ( y(2:mm1) - y(1:mm2) ) ./ h(1:mm2) ) .* h(2:mm1) + ...
        ( ( y(3:m)   - y(2:mm1) ) ./ h(2:mm1) ) .* h(1:mm2);
  r   = 3 * r;
  r(1)   = r(1)   - z0 * h(2);
  r(mm2) = r(mm2) - zf * h(mm2);
  z = T \ r; % risoluzione di un sistema tridiagonale
  z = [z0; z; zf];
  C = zeros(mm1, 4);
  C(:,4) = y(1:mm1); % coeff. del termine costante
  C(:,3) = z(1:mm1); % coeff. di x - x_i
  C(:,1) = ...       % coeff. di (x - x_i)^3
           ( z(2:m) + z(1:mm1) ) .* ( h(1:mm1).^2 ) ...
           - 2*( y(2:m) - y(1:mm1) ) .* h(1:mm1);
  C(:,2) = ...       % coeff. di (x - x_i)^2
           ( ( y(2:m) - y(1:mm1) ) ./ h(1:mm1) ...
             - z(1:mm1) ...
           ) ./ h(1:mm1) - C(:,1) .* h(1:mm1);
end
```

Codice Matlab per spline cubiche generali

```
function [yy] = valSpline(C, x, xx)
% VALSPLINE - Valutazione di una spline cubica interpolante
%  Calcola i valori yy in xx della spline cubica interpolante sui
%  nodi x, con coeff. nelle colonne di C, usando lo schema di Horner
% INPUT
%  C  (double array) - matrice dei coefficienti della spline cubica
%                      sulla partizione x, nella forma 'pp' (in
%                      colonna 1 il coeff. della potenza 3).
%  x  (double array) - vettore (ORDINATO) dei nodi di interpolazione
%  xx (double array) - vettore dei punti in cui calcolare la spline
% OUTPUT
%  yy (double array) - vettore dei valori della spline nei punti xx
%
  x = x(:); xx = xx(:); m = size(C,1); yy = zeros(size(xx));
  for i = 1 : length(xx)
    if ( xx(i) < x(1) || xx(i) > x(end) )
      error(sprintf('xx(%d) esterno all''intervallo',i));
    end
    if ( xx(i) == x(end) )
      k = m;
    else
      k = min( find( x > xx(i) ) ) - 1;
    end
```


Codice Matlab per spline cubiche generali

```
    yy(i) = ( ( C(k,1) * (xx(i) - x(k)) + C(k,2) ...
              ) * (xx(i) - x(k)) + C(k,3) ...
            ) * (xx(i) - x(k)) + C(k,4);  % oppure, yy(i) = ...
  end
end
```

In Matlab sono predefinite le seguenti funzioni:

spline: `pp = spline(x, y)` restituisce in `pp` la struttura (nella forma “pp”, ossia *piecewise polynomial*) della spline cubica interpolante i valori `y` nei nodi `x`.

`yy = spline(x, y, xx)` restituisce in `yy` i valori della spline cubica interpolante i valori `y` sui nodi `x`, valutata nei punti `xx`.

mkpp: `pp = mkpp(x, coeff)` restituisce in `pp` la struttura, in forma “pp”, del generico polinomio a tratti di grado $k - 1$ sulla partizione `x`, dove k è il numero di colonne della matrice `coeff` dei coefficienti su ciascun intervallo della partizione.

unmkpp: `[x, coeff, n, k, d] = unmkpp(pp)` estrae dal polinomio a tratti `pp` i nodi `x`, la matrice `coeff` dei coefficienti su ciascun intervallo, il numero `n` di intervalli della partizione, l'ordine massimo `k` (ordine = grado + 1) dei polinomi e la dimensione `d` dei coefficienti.

ppval: `yy = ppval(pp, xx)` restituisce i valori del polinomio a tratti rappresentato nella struttura `pp` (in forma “pp”) valutato nei punti `xx`.

Le funzioni **interp1**, **interp2**, **interp3** e **interpn** forniscono vari altri metodi di interpolazione mono-, bi-, tri- ed n-dimensionale, rispettivamente.

Teoremi sulle spline cubiche

Teorema

Tra tutte le funzioni $f \in C^2([a,b])$ tali che

$$f(x_i) = y_i \quad \forall i = 0, \ldots, m+1, \qquad x_0 \equiv a, \quad x_{m+1} \equiv b$$

e tali che valga una delle condizioni

a) $f'(x_0) = z_0, \qquad f'(x_{m+1}) = z_{m+1}$

b) $f''(x_0) = f''(x_{m+1}) = 0$

c) $f^{(k)}(x_0) = f^{(k)}(x_{m+1}) \qquad k = 0, 1, 2$

la spline cubica di interpolazione per cui vale a), oppure b), oppure c) è quella per cui vale la proprietà di ottimalià

$$\int_a^b \big(s''(x)\big)^2 dx \leqslant \int_a^b \big(f''(x)\big)^2 dx$$

l'uguaglianza valendo se e solo se $f \equiv s$.

Dim. Usando la relazione $(u-v)^2 = u^2 - v^2 - 2v(u-v)$, si ha:

$$\int_a^b \big(f''(x) - s''(x)\big)^2 dx = \int_a^b \big(f''(x)\big)^2 dx - \int_a^b \big(s''(x)\big)^2 dx - 2\int_a^b s''(x)\big(f''(x) - s''(x)\big) dx$$

Integrando per parti l'ultimo termine, si ha:

$$\int_a^b s''(x)\big(f''(x) - s''(x)\big) dx = \underbrace{s''(x)\big(f'(x) - s'(x)\big)\Big|_a^b}_{=0 \text{ per tutti i casi a), b), c)}} - \int_a^b s'''(x)\big(f'(x) - s'(x)\big) dx$$

$$= -\sum_{i=0}^{m} \left(s'''(x) \underbrace{\big(f(x) - s(x)\big)}_{=0 \text{ in } x_i \text{ e } x_{i+1}} \Bigg|_{x_i}^{x_{i+1}} \right) + \int_a^b \underbrace{s^{(4)}(x)}_{=0} \big(f(x) - s(x)\big) dx = 0$$

$$\Rightarrow \int_a^b \big(s''(x)\big)^2 dx \leqslant \int_a^b \big(s''(x)\big)^2 dx + \int_a^b \big(f''(x) - s''(x)\big)^2 dx = \int_a^b (f''(x))^2 dx$$

Inoltre, se $f \equiv s$ allora vale l'uguaglianza.

Viceversa, se vale l'uguaglianza $\int_a^b \big(f''(x)\big)^2 dx = \int_a^b \big(s''(x)\big)^2 dx$, allora $\int_a^b \big(f''(x) - s''(x)\big)^2 dx = 0$ e dunque $f(x) - s(x) = px + q$.

Poiché $f(x_0) = s(x_0)$ e $f(x_{m+1}) = s(x_{m+1})$, si ha $\begin{cases} px_0 + q = 0 \\ px_{m+1} + q = 0 \end{cases} \Rightarrow p = q = 0.$

Teoremi sulle spline cubiche

Teorema

Sia $f \in C^2([a,b])$ e sia $s(x)$ la spline cubica interpolante f. Allora, posto $h = \max\limits_{i=0,\dots,m}(x_{i+1} - x_i)$, per ogni $x \in [a,b]$ si ha

$$\left|f(x) - s(x)\right| \leqslant h^{3/2} \left(\int_a^b \left(f''(x)\right)^2 \mathrm{d}x \right)^{1/2}$$

$$\left|f'(x) - s'(x)\right| \leqslant h^{1/2} \left(\int_a^b \left(f''(x)\right)^2 \mathrm{d}x \right)^{1/2}$$

Poiché $\left|f''(x)\right| \leqslant M \; \forall x \in [a,b]$ e $\forall f \in C^2([a,b])$, segue che

$$\left|f(x) - s(x)\right| \leqslant h^{3/2} M \sqrt{b-a}$$

Pertanto, fissato $\epsilon \geqslant h^{3/2} M \sqrt{b-a}$, $s(x)$ è approssimazione di $f(x)$ in $[a,b]$ entro la tolleranza ϵ.

Uso della funzione `spline` di Matlab

```
yy = spline(x, y, xx)
```

- La funzione spline con tre parametri di ingresso valuta nel vettore di punti `xx` la spline cubica relativa ai nodi `x` e ai dati `y`. La spline che viene calcolata è quella per cui il secondo nodo e il penultimo nodo della suddivisione su cui essa è basata godono di una condizione "*not-a-knot*", ossia in tali nodi le derivate terze da destra e da sinistra coincidono e quindi la spline nei due intervalli interessati è rappresentata dallo stesso polinomio di grado 3.
- Se la funzione `spline` viene usata con due parametri di ingresso, cioè `pp = spline(x, y)`, il parametro di ritorno `pp` è una struttura che permette di descrivere la spline:

```
>> x = 0 : 5;
>> y = x .* exp( -(x-1).^2 );
>> pp = spline(x, y)
pp =
  form: 'pp'
  breaks: [0 1 2 3 4 5]
  coefs: [5x4 double]
  pieces: 5
  order: 4
  dim: 1
```

Uso della funzione `spline` di Matlab

`breaks` è il vettore dei nodi; `coefs` è una matrice con tante righe quanti sono i sottointervalli e un numero di colonne pari al numero dei coefficienti del polinomio che rappresenta la spline in ogni sottointervallo, nella forma:

$$\mathtt{coefs}_{i,1}(x-\mathtt{breaks}_i)^3+\mathtt{coefs}_{i,2}(x-\mathtt{breaks}_i)^2+\mathtt{coefs}_{i,3}(x-\mathtt{breaks}_i)+\mathtt{coefs}_{i,4}$$

ordinati dal coefficiente della potenza di grado più alto a quello del termine costante; `pieces` è il numero di sottointervalli; `dim` è il numero di variabili indipendenti; `order` è l'ordine della spline (che è il grado del polinomio più 1).

- Per estrarre le componenti della rappresentazione `pp` si usa la funzione

  ```
  [x, C, m, k] = unmkpp( pp )
  ```

  dove `x` è il vettore dei nodi della suddivisione, `C` è la matrice `m` $\times$ `k` dei coefficienti, `k` è l'ordine della spline e `m` è il numero dei sottointervalli.
  Viceversa, conoscendo `x` e `C`, si costruisce la rappresentazione `pp` della spline corrispondente con:

  ```
  pp = mkpp(x, C)
  ```

- Per valutare in un vettore `xx` una spline di cui si conosce la rappresentazione `pp` si usa:

  ```
  yy = ppval(pp, xx)
  ```