APPROSSIMAZIONE DI FUNZIONI E DI DATI

Letizia SCUDERI

Dipartimento di Scienze Matematiche, Politecnico di Torino

letizia.scuderi@polito.it

A.A. 2017/2018

Approssimazione di funzioni e di dati

L'approssimazione riveste un ruolo fondamentale nel Calcolo Numerico.

Approssimare una funzione f, nota in forma analitica, significa sostituirla con una funzione $\tilde{f}$ che le sia "vicina" in qualche senso e abbia una forma più semplice (per esempio, polinomiale) su cui si possa facilmente operare.

L'approssimazione di una funzione è utilizzata per esempio per la valutazione della funzione stessa su un calcolatore, oppure per rendere possibili operazioni quali la derivazione o l'integrazione.

Approssimare un insieme di dati (x_i, y_i) (ove, per esempio, $y_i = f(x_i)$ e f non è nota esplicitamente) significa determinare una funzione $\tilde{f}$ che abbia un andamento analogo a quello dei dati (o della funzione che ha generato i suddetti dati).

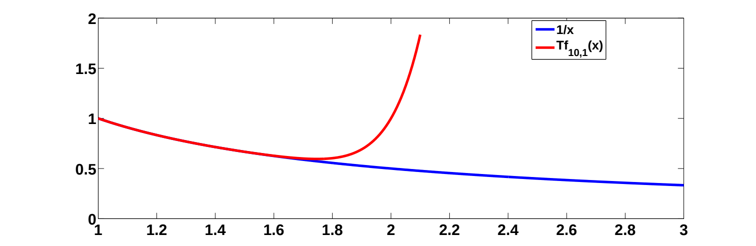
Un'applicazione di tale operazione si ha, per esempio, nella statistica, nella finanza, nella fisica e ogni qual volta si vuole costruire un modello che descriva il fenomeno in questione e ci permetta di fare previsioni attendibili in punti x diversi da x_i.

Osservazione

Com'è noto una funzione sufficientemente regolare f può essere approssimata nell'intervallo $(x_0 - \delta, x_0 + \delta)$ con $\delta > 0$ dal suo polinomio di Taylor Tf_{n,x_0}, di grado opportuno n e centrato in x_0.
Tale tipo di approssimazione, tuttavia, può essere insoddisfacente nelle applicazioni come dimostrano i seguenti esempi.

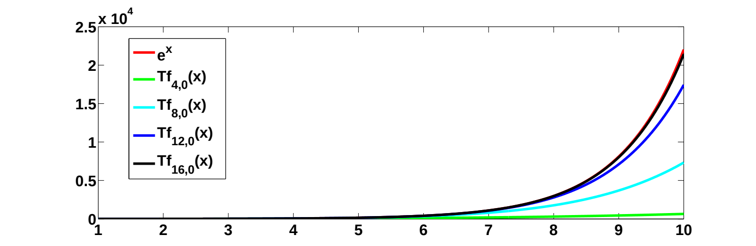
Esempio 1

Sia $f(x) = 1/x$ e $Tf_{10,1}(x)$ il polinomio di Taylor della funzione f, di grado 10 e di centro $x_0 = 1$. In figura si può notare come il polinomio di Taylor $Tf_{10,1}$ si discosti sempre di più da f, man mano che ci si allontana da x_0. Pertanto, il polinomio di Taylor Tf_{n,x_0} fornisce una buona approssimazione della funzione solo in un intorno del centro x_0.



Esempio 2

Sia $f(x) = e^x$ e $Tf_{n,0}(x)$ il polinomio di Taylor della funzione f, di grado n e di centro $x_0 = 0$. In figura si può notare come la funzione esponenziale possa essere bene approssimata mediante il suo polinomio di Taylor di centro $x_0 = 0$ in un intervallo più ampio di quello precedentemente considerato, purché il grado n sia sufficientemente grande. In tal caso, il calcolo di Tf_{n,x_0} richiede la conoscenza di f e delle sue derivate fino all'ordine n, che può talvolta essere elevato.



Servono quindi, in generale, dei metodi di approssimazione alternativi. Siano assegnati $n+1$ dati (x_i, y_i), $i = 1, \dots, n+1$, ove y_i è il valore assunto da una certa funzione $f(x)$ in x_i oppure la misura di un dato sperimentale in corrispondenza di x_i.

Definizione

Il **criterio dell'interpolazione** consiste nello scegliere, come approssimante dei dati (x_i, y_i), $i = 1, \dots, n+1$, una funzione $\tilde{f}(x)$ soddisfacente le seguenti condizioni:

$$\tilde{f}(x_i) = y_i \quad i = 1, \dots, n+1,$$

che impongono il passaggio di $\tilde{f}$ per i punti (x_i, y_i).

Le suddette equazioni sono dette **condizioni di interpolazione** e i punti x_i sono detti **nodi di interpolazione**; inoltre, si dice che $\tilde{f}$ **interpola i dati** (x_i, y_i) oppure, se $y_i = f(x_i)$, **interpola la funzione** f nei punti x_i.

In seguito verranno considerate:

- funzioni interpolanti $\tilde{f}(x)$ **polinomiali**, generalmente utilizzate per lo sviluppo di algoritmi nella differenziazione, nell'integrazione e nella soluzione di problemi differenziali;
- funzioni interpolanti $\tilde{f}(x)$ **polinomiali a tratti**, largamente utilizzate nella grafica e nella soluzione di problemi differenziali.

Interpolazione polinomiale

Teorema

Assegnati $n+1$ punti (x_i, y_i), con le ascisse distinte tra loro ($x_i \neq x_j$ per $i \neq j$), esiste uno e un sol polinomio $p_n(x)$ di grado minore o uguale a n interpolante i dati assegnati, ovvero soddisfacente le condizioni di interpolazione:

$$p_n(x_i) = y_i, \quad i = 1, \ldots, n+1$$

Dimostrazione

Per l'esistenza del polinomio interpolante verrà in seguito data una dimostrazione costruttiva.

Per dimostrare l'unicità, invece, si procede per assurdo.

... continua dimostrazione

Se esistessero due polinomi distinti $p_n(x)$ e $q_n(x)$ di grado n, entrambi soddisfacenti le condizioni di interpolazione $p_n(x_i) = y_i$ e $q_n(x_i) = y_i$, $i = 1, \ldots, n+1$, la loro differenza $p_n(x) - q_n(x)$ sarebbe un polinomio non identicamente nullo e di grado n, che si annullerebbe negli $n+1$ nodi distinti x_i,

$$p_n(x_i) - q_n(x_i) = 0 \quad i = 1, \ldots, n+1$$

e ciò sarebbe in contraddizione con il Principio di identità dei polinomi[1]. Pertanto, q_n deve coincidere con p_n.

[1] Un polinomio di grado n (a coefficienti reali o complessi) che ammette $n+1$ radici distinte è necessariamente il polinomio identicamente nullo.

Rappresentazione monomiale del polinomio di interpolazione

Definizione

Si definisce **rappresentazione monomiale** del polinomio interpolante $p_n(x)$ l'espressione:

$$p_n(x) = c_1 x^n + c_2 x^{n-1} + \ldots + c_n x + c_{n+1},$$

ove i coefficienti c_i, $i = 1, \ldots, n+1$, sono determinati imponendo che $p_n(x)$ soddisfi le condizioni di interpolazione.

I seguenti comandi Matlab consentono di ottenere i coefficienti della rappresentazione monomiale e di valutare il polinomio interpolante in qualsiasi punto o vettore di punti assegnato.

Comandi Matlab

- `c = polyfit(x,y,n)` calcola e memorizza in `c`, i coefficienti della rappresentazione monomiale del polinomio di grado `n`

$$p_n(x) = c(1)x^n + c(2)x^{n-1} + \ldots + c(n)x + c(n+1),$$

  interpolante i dati (x_i, y_i), $i = 1, \ldots, n+1$, le cui ascisse x_i e ordinate y_i sono memorizzate nei vettori `x` e `y`, rispettivamente.

- `p = polyval(c,z)` calcola e memorizza nel vettore `p`, i valori che un **qualsiasi** polinomio

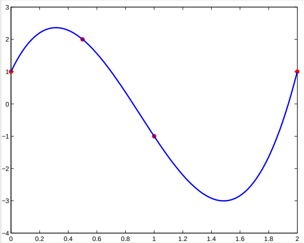
$$p_n(x) = c(1)x^n + c(2)x^{n-1} + \ldots + c(n)x + c(n+1),$$

  i cui coefficienti sono memorizzati in `c`, assume nelle componenti del vettore `z`.

Esempio 1

Le seguenti istruzioni Matlab consentono di calcolare e rappresentare graficamente il polinomio interpolante i dati $(0,1)$, $(1,-1)$, $(2,1)$, $(1/2,2)$.

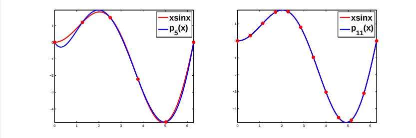
```
>> x = [0 1 2 1/2];
>> y = [1 -1 1 2];
>> c = polyfit(x,y,length(x)-1);
>> z = linspace(min(x),max(x));
>> p = polyval(c,z);
>> plot(x,y,'ro',z,p,'b')
```



Esempio 2

Le seguenti istruzioni Matlab consentono di calcolare i coefficienti dei polinomi di grado 5 e 11 interpolanti la funzione $f(x) = x \sin x$ in punti equispaziati nell'intervallo $[0, 2\pi]$.

```
f = @(x) x.*sin(x);
z = linspace(0,2*pi);
for n = [5 11]
   x = linspace(0,2*pi,n+1);
   y = f(x);
   c = polyfit(x,y,n);
   p = polyval(c,z);
   plot(x,y,'ro',z,f(z),'r',z,p,'b')
   pause
end
Warning: Polynomial is badly
conditioned.....
```



Osserviamo che Matlab ha inviato una segnalazione di cattivo condizionamento. Tuttavia, dal punto di vista grafico, il mal condizionamento del problema non sembra avere conseguenze evidenti.

L'esempio precedente mostra che la determinazione del polinomio interpolante mediante la sua rappresentazione monomiale comporta la risoluzione di un problema mal condizionato.

Tale problema è tanto più mal condizionato, quanto più è grande n. Pertanto, quando n è grande, è necessario calcolare il polinomio interpolante attraverso rappresentazioni alternative.

Tuttavia, fintantoché il grado del polinomio non è eccessivamente grande, è comunque possibile calcolare i coefficienti del polinomio interpolante tramite `polyfit` in quanto, come mostra l'esempio precedente, le conseguenze del mal condizionamento non sono ancora evidenti.

Osservazione

È doveroso osservare che il comando `c=polyfit(x,y,m)` di Matlab è stato concepito per il calcolo dei coefficienti del polinomio di grado m che soddisfa un criterio di approssimazione diverso da quello dell'interpolazione, noto con il nome di criterio dei minimi quadrati.

Nel caso particolare in cui $m = n$, ove n è il numero dei dati assegnati (x_i, y_i), $i = 1 \ldots, n+1$, diminuito di un'unità, il polinomio approssimante che si ottiene con il comando `c=polyfit(x,y,n)` soddisfa anche il criterio dell'interpolazione.

Pertanto, per calcolare i coefficienti di un polinomio interpolante, è possibile utilizzare il comando `polyfit`.

Tuttavia, come precedentemente osservato, per $m = n$ si può pervenire a un problema mal condizionato e, pertanto, in linea di principio non è conveniente utilizzare `polyfit` per calcolare il polinomio interpolante.

Rappresentazione di Lagrange del polinomio di interpolazione

Per l'effettiva costruzione di un polinomio interpolante è dunque necessario utilizzare rappresentazioni alternative a quella monomiale.

La rappresentazione, che ora si introduce, fornisce anche una dimostrazione costruttiva dell'esistenza del polinomio interpolante.

Siano x_i, $i = 1, \ldots, n+1$, $n+1$ **nodi distinti** e si considerino i seguenti polinomi di grado n:

$$\ell_j(x) = \frac{(x - x_1)\cdots(x - x_{j-1})(x - x_{j+1})\cdots(x - x_{n+1})}{(x_j - x_1)\cdots(x_j - x_{j-1})(x_j - x_{j+1})\cdots(x_j - x_{n+1})}$$

per $j = 1, \ldots, n+1$.

Poiché

$$\ell_j(x_i) = \begin{cases} 1 \text{ se } i = j \\ 0 \text{ se } i \neq j \end{cases},$$

$\ell_j(x)$ interpola i dati:

$$(x_1, 0),\ \ldots\ (x_{j-1}, 0),\ (x_j, 1),\ (x_{j+1}, 0),\ \ldots, (x_{n+1}, 0).$$

Assegnati i dati (x_i, y_i), $i = 1, \ldots, n$, è immediato verificare che il polinomio di grado n, così definito

$$p_n(x) = \sum_{j=1}^{n+1} y_j \ell_j(x)$$

soddisfa le condizioni di interpolazione. Infatti, risulta

$$p_n(x_i) = \sum_{j=1}^{n+1} y_j \ell_j(x_i) = y_i \ell_i(x_i) = y_i, \quad i = 1, \ldots, n+1$$

Dall'unicità del polinomio di grado n interpolante $n+1$ punti segue che l'espressione sopra del polinomio p_n è dunque una seconda rappresentazione del polinomio interpolante (precedentemente definito nella forma monomiale).

Definizione

Si definisce **rappresentazione di Lagrange** del polinomio p_n interpolante i dati (x_i, y_i), $i = 1, \ldots, n+1$ con $x_i \neq x_j$ per $i \neq j$, l'espressione

$$p_n(x) = \sum_{j=1}^{n+1} y_j \ell_j(x)$$

I polinomi $\ell_j(x)$ di grado n sono detti **polinomi fondamentali di Lagrange** associati ai nodi x_i.

Esempio

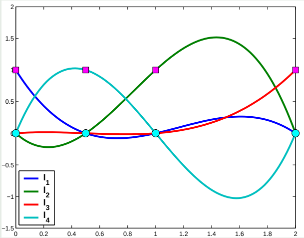
Scriviamo la rappresentazione di Lagrange del polinomio $p_3(x)$ interpolante i dati $(0, 1)$, $(1, -1)$, $(2, 1)$, $(1/2, 2)$, precedentemente considerati. Per i polinomi $\ell_j(x)$, $j = 1, 2, 3, 4$, si ha:

$$\ell_1(x) = \frac{(x-1)(x-2)(x-1/2)}{(0-1)(0-2)(0-1/2)}$$

$$\ell_2(x) = \frac{(x-0)(x-2)(x-1/2)}{(1-0)(1-2)(1-1/2)}$$

$$\ell_3(x) = \frac{(x-0)(x-1)(x-1/2)}{(2-0)(2-1)(2-1/2)}$$

$$\ell_4(x) = \frac{(x-0)(x-1)(x-2)}{(1/2-0)(1/2-1)(1/2-2)}$$



Quindi la rappresentazione di Lagrange del polinomio $p_3(x)$ interpolante è

$$p_3(x) = 1 \cdot \ell_1(x) - 1 \cdot \ell_2(x) + 1 \cdot \ell_3(x) + 2 \cdot \ell_4(x)$$

Osservazione 1

La rappresentazione di Lagrange del polinomio di interpolazione è interessante per diversi motivi:

- fornisce una dimostrazione costruttiva dell'esistenza del polinomio interpolante;
- rappresenta la base per la costruzione di formule di integrazione numerica e di metodi per l'approssimazione della soluzione di problemi differenziali.

Osservazione 2

Si osservi che ciascun polinomio $\ell_j(x)$ dipende da tutti i nodi $x_1, \ldots, x_{n+1}$ e, di conseguenza, se si desidera aggiungere un dato di interpolazione (x_{n+2}, y_{n+2}) all'insieme dei dati $\{(x_i, y_i)\}_{i=1,\ldots,n+1}$, occorre ricalcolare tutti i polinomi $\ell_j(x)$!

Pertanto, dal punto di vista del calcolo numerico del polinomio, la rappresentazione di Lagrange del polinomio interpolante è di interesse pratico solo per un numero piccolo di dati di interpolazione oppure quando i nodi x_i sono fissati e si calcolano i polinomi di interpolazione corrispondenti a diversi insiemi di valori $\{y_i\}$.

Nella pratica, per il calcolo numerico del polinomio di interpolazione, si utilizza una rappresentazione alternativa a quella monomiale e di Lagrange, nota come rappresentazione di Newton.

Convergenza del polinomio di interpolazione

Definizione

Denotato con $p_n(x)$ il polinomio interpolante la funzione $f(x)$ nei punti $x_1, \ldots, x_{n+1}$, si definisce **errore di interpolazione** la funzione

$$E_n(x) = f(x) - p_n(x)$$

Osservazione

- $E_n(x) = 0$ per $x = x_i$;
  infatti si ha $E_n(x_i) = f(x_i) - p_n(x_i) = 0$.
- $E_n(x) \equiv 0$ per ogni $x \in [a, b]$ e per ogni funzione f polinomiale di grado minore o uguale a n;
  infatti, posto $f = p$ e indicato con p_n il polinomio interpolante f, per l'unicità del polinomio interpolante, deve essere $p \equiv p_n$.

Definizione

Data una funzione $g(x)$ continua in $[a, b]$, si definisce **norma uniforme** o **norma infinito** della funzione g la quantità

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

Supponendo che f sia continua in $[a, b]$ e che i punti x_i siano tutti in $[a, b]$, una misura naturale dell'errore è $||E_n||_\infty$.

Definizione

Si dice che una successione di polinomi $\{p_n\}_{n\in\mathbb{N}}$ **converge uniformemente** a f se, e solo se,

$$\lim_{n\to\infty} ||f - p_n||_\infty = 0$$

Ci si pone ora la seguente domanda:
data una **qualunque successione di nodi** distinti,

$$\begin{array}{cccc}
x_1^{(0)} & & & \\
x_1^{(1)} & x_2^{(1)} & & \\
x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \\
\dots & \dots & \dots & \dots \\
x_1^{(n)} & x_2^{(n)} & \dots & x_{n+1}^{(n)} \\
\dots & \dots & \dots & \dots
\end{array}$$

tutti situati in $[a, b]$ e data una **qualunque funzione** f **continua** in $[a, b]$, denotato con $p_n(x)$ il polinomio di grado n interpolante $f(x)$ in $x_1^{(n)}, x_2^{(n)}, \dots, x_{n+1}^{(n)}$, è sempre vero che

$$\lim_{n\to\infty} ||f - p_n||_\infty = 0?$$

La risposta a questa domanda è **negativa**!

Infatti, si può dimostrare il seguente teorema.

Teorema

Data una **qualunque successione di nodi** distinti, tutti situati in $[a, b]$, esiste sempre una **funzione continua** $f(x)$ in $[a, b]$ che, interpolata su quei nodi, genera una successione di polinomi di interpolazione $\{p_n(x)\}$ **non uniformemente convergente** a $f(x)$ in $[a, b]$.

Se si considerano i **nodi equispaziati** in $[a, b]$,

$$x_i = a + (i-1)h, \quad i = 1, \ldots, n+1, \quad h = \frac{b-a}{n}$$

come nodi di interpolazione, l'esempio che segue mostra che la successione dei polinomi interpolanti la funzione

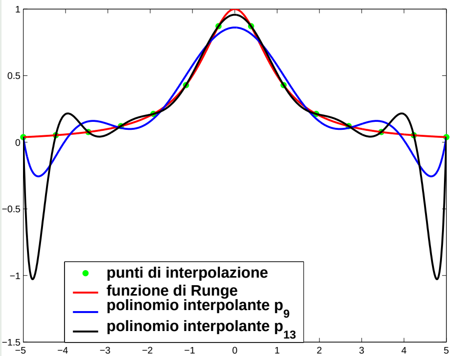
$$f(x) = \frac{1}{1+x^2}$$

infinitamente derivabile, detta **funzione di Runge**, non converge uniformemente a $f(x)$ in $[a, b]$, per certi intervalli $[a, b]$; per $[a, b] = [-5, 5]$, al tendere del numero dei nodi all'infinito, il limite della norma uniforme dell'errore di interpolazione è addirittura infinito!

Esempio

Denotato con p_n il polinomio interpolante $f(x) = \frac{1}{1+x^2}$ nei nodi x_i, $i = 1, \ldots, n+1$, equispaziati in $[-5, 5]$, si ha

$$\lim_{n\to\infty} ||E_n||_\infty = \infty$$

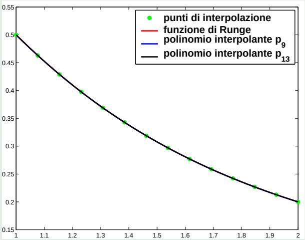


Se si considerano, invece, i nodi di interpolazione equispaziati in $[1, 2]$ anziché in $[-5, 5]$, l'errore di interpolazione converge uniformemente a zero, al tendere del numero dei nodi all'infinito.

Esempio

Denotato con p_n il polinomio interpolante $f(x) = \frac{1}{1+x^2}$ nei nodi x_i, $i = 1, \dots, n+1$, equispaziati in $[1, 2]$, si ha

$$\lim_{n\to\infty} ||E_n||_\infty = 0$$



Un'alternativa ai nodi equispaziati è rappresentata, per esempio, da una delle due seguenti scelte.

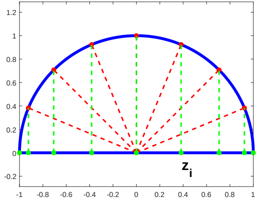
Definizione

Si definiscono **nodi di Chebyshev-Lobatto** i seguenti punti

$$z_i = -\cos\left(\frac{(i-1)\pi}{n}\right) \in [-1, 1], \quad i = 1, \ldots, n+1$$

In figura, si riporta la distribuzione dei nodi di Chebyshev-Lobatto per $n = 8$.

Figura: Nodi di Chebyshev-Lobatto z_i, $i = 1, ..., 9$, $n = 8$.



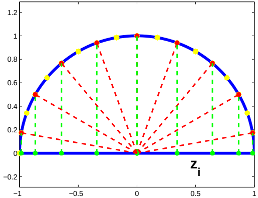
Definizione

Si definiscono **nodi di Chebyshev** i seguenti punti

$$z_i = -\cos\left(\frac{(2i-1)\pi}{2(n+1)}\right) \in (-1,1), \quad i = 1,\ldots,n+1$$

In figura, si riporta la distribuzione dei nodi di Chebyshev per $n = 8$.

Figura: Nodi di Chebyshev z_i, $i = 1, ..., 9$, $n = 8$.



Scegliendo, come nodi di interpolazione, i punti

$$x_i = \frac{b-a}{2} z_i + \frac{b+a}{2}, \quad i = 1, \ldots, n+1,$$

che rappresentano i nodi di Chebyshev-Lobatto oppure di Chebyshev nell'intervallo $[a, b]$, si può dimostrare che la successione dei polinomi $\{p_n(x)\}$, interpolanti una funzione $f \in C^1([a, b])$ nei nodi x_i, converge uniformemente a f, al tendere del numero dei nodi all'infinito.

Più precisamente, si può dimostrare il seguente teorema.

Teorema

Sia $\{p_n(x)\}$ la successione dei polinomi interpolanti $f(x)$ nei nodi $x_i \in [a, b]$ di Chebyshev-Lobatto oppure di Chebyshev. Se $f \in C^k([a, b])$, $k \geq 1$, allora

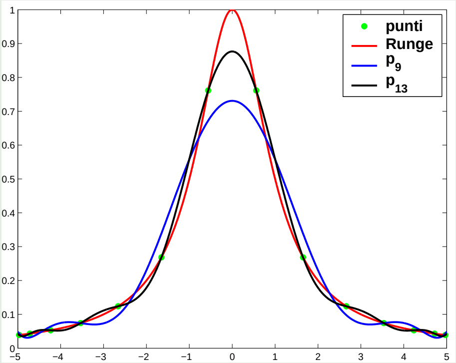
$$||f - p_n||_\infty = O\left(\frac{\log n}{n^k}\right), \quad n \to \infty$$

Pertanto, la convergenza è tanto più rapida, quanto più è regolare la funzione f.

Esempio

Denotato con p_n il polinomio interpolante $f(x) = \frac{1}{1+x^2}$ nei nodi di Chebyshev x_i, $i = 1, \ldots, n+1$, $n = 9, 13$ e relativi all'intervallo $[-5, 5]$, si ha

$$\lim_{n \to \infty} ||f - p_n||_\infty = 0$$



Gli esempi precedenti mostrano dunque che:

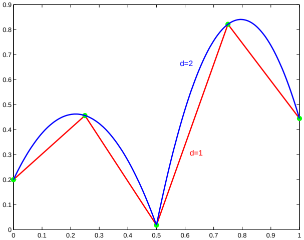
- a un aumento del grado n del polinomio interpolante, non corrisponde necessariamente un miglioramento nell'approssimazione di una funzione f;
- la convergenza uniforme del polinomio interpolante alla funzione interpolata dipende dalla scelta dei nodi e dalla regolarità della funzione f;
- la scelta dei nodi equispaziati non garantisce la convergenza uniforme, neppure per funzioni dotate di derivata continua di qualunque ordine;
- se f è continua con la sua derivata prima, la scelta dei nodi di Chebyshev oppure di Chebyshev-Lobatto, invece, garantisce la convergenza uniforme.

Funzioni polinomiali a tratti: spline

Definizione

Si definisce **funzione polinomiale a tratti** di grado d associata a una partizione dell'intervallo $[a, b]$, una funzione continua in $[a, b]$ e definita da un'unione di tratti contigui di polinomi algebrici diversi, ciascuno di grado d.

In generale il grado d viene scelto piccolo ($d = 1, 2, 3$).



Si osservi che le funzioni polinomiali a tratti sono continue ma, generalmente, non sono derivabili nei punti di raccordo.

Le funzioni **spline** sono particolari funzioni polinomiali a tratti che soddisfano opportune proprietà di regolarità nei punti di raccordo.

Definizione

Sia $a \equiv x_1 < x_2 < ... < x_{n+1} \equiv b$, una partizione dell'intervallo $[a, b]$ e $y_i = f(x_i)$. Si definisce **spline di ordine d interpolante** f nei nodi assegnati x_i, una funzione $S_d(x)$ soddisfacente le seguenti condizioni:

i) $S_d(x)$ è un polinomio di grado (al più) d per $x \in [x_i, x_{i+1}]$, $i = 1, \ldots, n$;

ii) la derivata $S_d^{(k)}(x)$ di ordine k di $S_d(x)$ è una funzione continua in $[a, b]$ per $k = 0, 1, \ldots, d-1$;

iii) $S_d(x_i) = y_i$, $i = 1, \ldots, n+1$.

La spline $S_1(x)$ di ordine 1 e interpolante f nei punti x_i, $i = 1, \ldots, n+1$ ha la seguente espressione:

$$S_1(x) = f(x_i) + \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}(x - x_i), \quad x \in [x_i, x_{i+1}]$$

Per calcolare e valutare la spline $S_1(x)$ si può utilizzare il seguente comando Matlab.

Comando Matlab

`s = interp1(x,y,z)` calcola e memorizza in `s` i valori che la spline lineare interpolante i dati (x_i, y_i), $i = 1, \ldots, n+1$, assume in `z`; i valori x_i e y_i sono memorizzati in `x` e in `y`, rispettivamente.

Ci si chiede ora se, aumentando il numero dei nodi di interpolazione in un intervallo $[a, b]$, per una data funzione continua f, la successione delle spline S_1 di ordine 1 e interpolanti f converga uniformemente a f.

A tal proposito si può dimostrare il seguente teorema.

Teorema

Data una partizione $a \equiv x_1 < x_2 < \ldots < x_{n+1} \equiv b$, sia S_1 la spline interpolante una funzione $f \in C^2([a, b])$ nei nodi della partizione x_i, $i = 1, \ldots, n+1$. Denotato con $h = \max_{1 \leq i \leq n}(x_{i+1} - x_i)$, allora

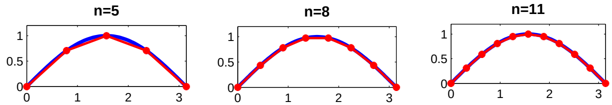
$$||f - S_1||_\infty = O(h^2), \quad h \to 0$$

La convergenza uniforme è comunque garantita qualunque sia la funzione continua f e qualunque sia la scelta dei nodi x_i.

Osservazione

Il comando `plot(x,y)` di Matlab con $y(i) = f(x(i))$ in generale non rappresenta graficamente la funzione $f(x)$ ma una sua approssimante, ovvero la spline $S_1(x)$ lineare e interpolante $f(x)$ nei nodi $x(i)$, $i = 1, ..., n + 1$.

Figura: Spline S_1 interpolante $f(x) = \sin(x)$ in $n = 5, 8, 11$ nodi equispaziati in $[0, \pi]$.



... continua osservazione

Poiché $O(h^2)$ è il massimo ordine di convergenza uniforme che si raggiunge con $S_1(x)$, anche quando $f \in C^k([a,b])$ con $k > 2$, per avere mediante il comando `plot` una rappresentazione grafica di f "corretta" nonché "liscia", occorre valutare la funzione in un numero opportunamente elevato di punti.

Le spline cubiche $S_3(x)$ sono funzioni polinomiali a tratti di grado locale 3 molto utilizzate nelle applicazioni.
Per determinare $S_3(x)$ è necessario assegnare 4 condizioni per ciascun sottointervallo, tante quanti sono i coefficienti che definiscono la spline in ogni sottointervallo:

$$S_3(x) = a_i + b_i x + c_i x^2 + d_i x^3, \quad x \in [x_i, x_{i+1}],\ i = 1, \dots, n$$

e, conseguentemente, $4n$ condizioni in tutto.

Imponendo le condizioni che definiscono una spline, ossia

- $n - 1$ condizioni di continuità di $S_3^{(k)}$, per $k = 0, 1, 2$, nei nodi interni $x_2, ..., x_n$
- $n + 1$ condizioni di interpolazione nei nodi $x_1, ..., x_{n+1}$

si hanno complessivamente $3(n - 1) + (n + 1) = 4n - 2$ condizioni.

Pertanto, per determinare in modo univoco i $4n$ coefficienti che individuano la spline, occorre imporre due ulteriori condizioni.

Fra tutte le possibili scelte, **le seguenti condizioni aggiuntive garantiscono l'esistenza e l'unicità delle corrispondenti spline cubiche**.

Definizione

Si definiscono **naturali** le spline cubiche soddisfacenti le condizioni i)-iii) e le condizioni aggiuntive

$$1) \qquad S_3^{(2)}(x_1) = 0, \; S_3^{(2)}(x_{n+1}) = 0$$

Si definiscono **"not-a-knot"** le spline cubiche soddisfacenti le condizioni i)-iii) e le condizioni aggiuntive

$$2) \qquad S_3^{(3)} \text{ è continua in } x_2 \text{ e in } x_n$$

Si definiscono **vincolate** le spline cubiche soddisfacenti le condizioni i)-iii) e le condizioni aggiuntive

$$3) \qquad S_3^{(1)}(x_1) = f'(x_1), \; S_3^{(1)}(x_{n+1}) = f'(x_{n+1})$$

Per quanto riguarda la convergenza uniforme delle spline cubiche interpolanti alla funzione interpolata, si può dimostrare il seguente teorema.

Teorema

Sia $S_3(x)$ la spline cubica interpolante i dati $(x_i, f(x_i))$ con $a \leq x_i \leq b$ e soddisfacente le condizioni aggiuntive 1), oppure 2), oppure 3).

Denotato con $h_i = x_{i+1} - x_i$ e $h = max_{1 \leq i \leq n} h_i$, se $f \in C^2([a, b])$ allora

$$||f - S_3||_\infty = O(h^2), \quad h \to 0$$

Inoltre, nel caso delle condizioni aggiuntive 2) oppure 3), se $f \in C^4([a, b])$ e se $h/h_i \leq \gamma < \infty$ per $h \to 0$, si ha

$$||f^{(p)} - S_3^{(p)}||_\infty = O(h^{4-p}), \quad h \to 0, \; p = 0, 1, 2, 3$$

Osservazioni

- la condizione $h/h_i \leq \gamma < \infty$ per $h \to 0$ impone che la partizione sia uniforme o quasi uniforme;
- $O(h^4)$ è il massimo ordine di convergenza che si ottiene con le suddette spline cubiche, cioè $O(h^4)$ vale anche quando $f \in C^k([a,b])$ con $k > 4$ (*saturazione*);
- le derivate di S_3, fino a quelle di ordine 3, convergono uniformemente alle corrispondenti derivate di f (*simultanea approssimazione*).

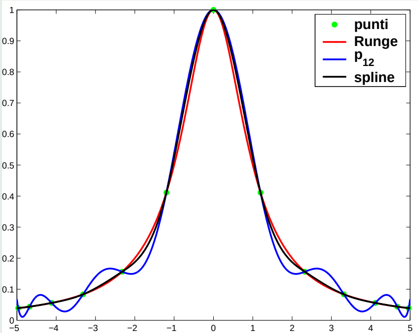
Per calcolare e valutare le spline "not-a-knot" o le spline vincolate si possono utilizzare i seguenti comandi.

Comandi Matlab

- `s = spline(x,y,z)` calcola e memorizza in `s` i valori che la spline cubica interpolante i dati (x_i, y_i), $i = 1, \ldots, n+1$ e soddisfacente la condizione 2) "not-a-knot", assume in `z`; i valori x_i e y_i sono memorizzati in `x` e in `y`, rispettivamente.
- `s = spline(x,[yd1 y ydn],z)` calcola e memorizza in `s` i valori che la spline cubica interpolante i dati (x_i, y_i), $i = 1, \ldots, n+1$ e soddisfacente la condizione 3) (`yd1`$= f'(x_1)$, `ydn`$= f'(x_{n+1})$), assume in `z`; i valori x_i e y_i sono memorizzati in `x` e in `y`, rispettivamente.

Esempio

In figura sono rappresentati graficamente il polinomio e la spline cubica "not-a-knot" interpolanti la funzione di Runge in 13 nodi di Chebyshev definiti nell'intervallo $[-5, 5]$.



Osservazioni

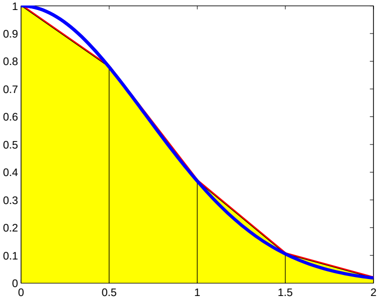
- A differenza dei polinomi interpolanti, le funzioni spline interpolanti una funzione continua convergono uniformemente alla funzione interpolata al crescere del numero dei nodi di interpolazione, qualunque sia la scelta dei nodi.
- In talune situazioni, per esempio nelle applicazioni grafiche, si utilizzano le spline interpolanti in luogo dei polinomi interpolanti, in quanto il comportamento eccessivamente oscillatorio del polinomio di interpolazione intorno alla funzione interpolata, al crescere del grado n, non è accettabile.

Un'importante applicazione dell'approssimazione di funzione è il calcolo di integrali.
È noto che non sempre si riesce a calcolare analiticamente un integrale. È questo, per esempio, il caso dell'integrale che definisce la **funzione degli errori**

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt$$

che interviene spesso nel calcolo delle probabilità, nella statistica e nelle equazioni differenziali alle derivate parziali.

In questi casi, si può procedere approssimando l'integrale di $f(x)$ mediante l'integrale della spline $S_1(x)$ lineare e interpolante f in nodi equispaziati, per esempio, dell'intervallo di integrazione; il vantaggio è che quest'ultimo valore è analiticamente calcolabile.



L'espressione che definisce il suddetto valore approssimato dell'integrale è detta **formula dei trapezi**, per via della sua interpretazione geometrica, ed è implementata nel seguente comando Matlab.

Comando Matlab

`t = trapz(x,y)` memorizza in `t` il valore approssimato dell'integrale mediante la formula dei trapezi, ottenuto cioè calcolando (analiticamente) l'integrale della spline lineare e interpolante i dati (x_i, y_i), $i = 1, \dots, n$; i valori x_i e y_i sono memorizzati nei vettori `x` e `y`, rispettivamente.

Esempio 1

Calcoliamo mediante la formula dei trapezi un'approssimazione t_n del valore $\mathrm{erf}(2) \approx 0.9953222650189527$ e l'errore assoluto a esso associato. Il valore di riferimento $\mathrm{erf}(2)$ è stato calcolato mediante il comando erf di Matlab.

```
f = @(t) exp(-t.^2);
x = 2;
j = 1;
for i = 2:9
  nodi(j) = 2.^i+1;
  xi = linspace(0,x,nodi(j));
  yi = f(xi);
  I(j) = 2/sqrt(pi)*trapz(xi,yi);
  err(j) = abs(erf(x)-I(j));
  j = j+1;
end
[nodi' I' err']
```

n	t_n	$\lvert\mathrm{erf}(2) - t_n\rvert$
5	0.9936717209022358	1.65e-03
9	0.9948961896883085	4.26e-04
17	0.9952149048226961	1.07e-04
33	0.9952953724020329	2.69e-05
65	0.9953155385796442	6.73e-06
129	0.9953205832038151	1.68e-06
257	0.9953218445523365	4.20e-07
513	0.9953221599014968	1.05e-07

Si osservi che, al crescere del numero dei nodi di interpolazione, l'accuratezza del valore approssimato dell'integrale aumenta.

Esempio 2. Raffreddamento di un liquido.

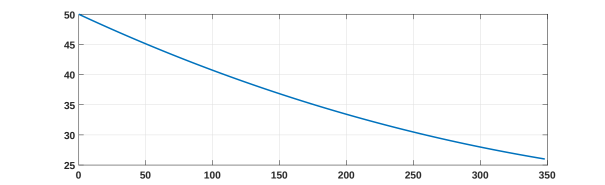
È stata misurata la temperatura del caffè che si raffredda in una tazza di porcellana a temperatura ambiente ($16^o C$). I dati ottenuti sono riportati nella tabella che segue:

Tempo (sec)	Temperatura ($^o C$)
0	50
62	44
226	32
348	26

Se vogliamo stimare il tempo in cui la temperatura del caffè ha raggiunto, per esempio $39^o C$, possiamo interpolare i dati assegnati.

... continua esempio

```
>> x = [0 62 223 348];
>> y = [50 44 32 26];
>> c = polyfit(x,y,length(x)-1);
>> z = linspace(min(x),max(x));
>> p = polyval(c,z);
>> plot(z,p)
>> grid on
```



Dal grafico si deduce che la temperatura di $39^o C$ viene raggiunta all'incirca dopo 110-120 secondi.

... continua esempio

Per calcolare il tempo con maggiore precisione, si può risolvere l'equazione algebrica $c(1)t^3 + c(2)t^2 + c(3)t + c(4) = 39$, mediante il comando Matlab r=roots(c), che fornisce in `r` le radici dell'equazione algebrica i cui coefficienti sono tutti memorizzati in `c`, a partire dal termine di grado massimo fino a quello di grado zero.

```
>> c(4) = c(4)-39;
>> r = roots(c)
r =
     1.0e+03 *
     4.0267
     1.1278
     0.1212
```

Pertanto, la temperatura di $39^o C$ viene raggiunta all'incirca dopo 121 secondi. Escludiamo i valori delle prime due radici in quanto maggiori di 348, tempo in cui la temperatura è già di $26^o C$.

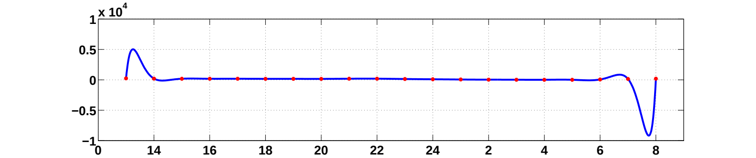
Esempio 3. Inquinamento dell'aria.

Secondo la banca dati della qualità dell'aria della regione Piemonte, nel giorno 13/11/2015 la centralina posta presso il Lingotto ha rilevato le seguenti concentrazioni di ossidi di azoto nell'aria (microgrammi/metro cubo) a intervalli di un'ora:

ora	ossidi di azoto	ora	ossidi di azoto
13:00	243	23:00	138
14:00	209	24:00	95
15:00	181	1:00	56
16:00	179	2:00	32
17:00	180	3:00	21
18:00	166	4:00	12
19:00	163	5:00	11
20:00	157	6:00	61
21:00	187	7:00	146
22:00	192	8:00	186

... continua esempio

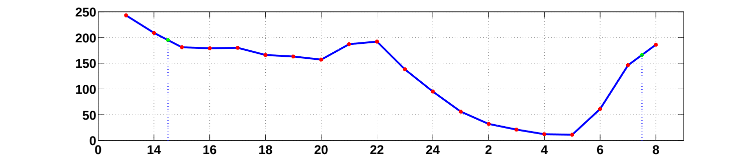
Vogliamo allora determinare una funzione che, sulla base dei dati disponibili, fornisca un'approssimazione dei livelli di concentrazione di ossidi di azoto in orari diversi da quelli in cui è stata effettuata la misurazione; per esempio, alle ore 14:30 e alle ore 7:30.
Se consideriamo il polinomio di grado 19 interpolante i dati assegnati, abbiamo:



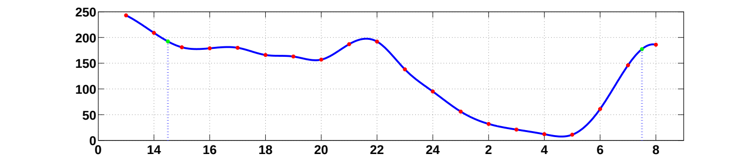
Le oscillazioni della funzione approssimante sono assolutamente inaccettabili; al mattino abbiamo, addirittura, valori negativi dei livelli di concentrazione!

... continua esempio

Se, invece, consideriamo la spline lineare interpolante i dati assegnati,



oppure la spline cubica not-a-knot interpolante,



otteniamo grafici, che appaiono più affidabili di quello generato dall'interpolatore di Lagrange negli stessi nodi.