Laboratorio di Calcolo Numerico
Interpolazione polinomiale.

Ángeles Martínez Calomardo
http://www.math.unipd.it/~acalomar/DIDATTICA/
angeles.martinez@unipd.it

Laurea in Informatica
A.A. 2018–2019

Introduzione

In questa lezione desideriamo introdurre dei metodi per determinare l'interpolante polinomiale di grado n, nei nodi a due a due distinti x_0, ..., x_n, di una funzione continua $f : [a, b] \to \mathbb{R}$, ovvero $p_n \in \mathbb{P}_n$ tale che

$$p_n(x_i) = f(x_i), i = 0, \ldots, n.$$

Mostreremo come farlo mediante:

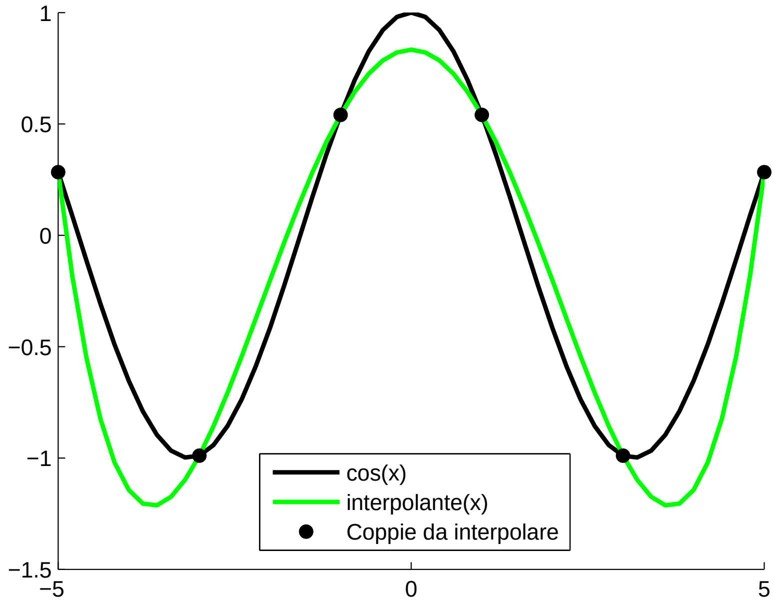
- i comandi Matlab `polyfit`, `polyval`.
- la formulazione di Newton di p_n che fa uso delle differenze divise e la valutazione di p_n in un punto (o vettore) x mediante l'algoritmo di Horner;

Interpolazione polinomiale

Sia $\mathbb{P}_n$ lo sp. vett. dei polinomi di grado n in $\mathbb{R}$. Date $n+1$ coppie (x_0,y_0), ..., (x_n,y_n) con $x_j \neq x_k$ se $j \neq k$, si calcoli $p_n \in \mathbb{P}_n$ t.c.

$$p_n(x_k) = y_k, \quad k = 0, \ldots, n$$

Tali polinomi p_n si dicono **interpolare** $\{(x_k, y_k)\}_{k=0,\ldots,n}$ oppure interpolare i valori y_k nei nodi x_k.



Alcuni fatti fondamentali

Alcune questioni risultano di importanza fondamentale:

- Esiste tale polinomio ed è unico? Sì.
- È possibile calcolarlo? Sì, $p_n(x) = \sum_{k=0}^{n} y_k L_k(x)$, dove L_k è il k-esimo polinomio di Lagrange relativo ai punti $\{x_k\}_{k=0,\dots,n}$.
- Per quanto riguarda l'errore:

Teorema

Sia $f \in C^{(n+1)}([a,b])$ e sia p_n il polinomio che interpola le coppie $\{(x_k, y_k)\}_{k=0,\dots,n}$ con $x_k \in [a,b]$, $k = 0, 1, \dots, n$, $x_k \neq x_s$ se $k \neq s$. Allora per ogni $x \in [a,b]$

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

dove $\xi \in \mathcal{I}$ con $\mathcal{I}$ il più piccolo intervallo contenente i nodi $x_0, \dots, x_n$ e x.

Interpolazione in Matlab/Octave

Supponiamo di dover interpolare le coppie $\{(x_k, y_k)\}_{k=0,\ldots,n}$ e supponiamo sia $\mathbf{x} = [x_0, \ldots, x_n]$, $\mathbf{y} = [y_0, \ldots, y_n]$. I coefficienti del polinomio interpolatore sono ottenibili con il comando `polyfit`.

Esempio:

```
>> x=[-2 1 3];
>> y=[-2 11 17];
>> a=polyfit(x,y,2)
a =
   -0.2667    4.0667    7.2000
```

In effetti, calcolando manualmente il polinomio interpolatore si ha, semplificando quanto ottenuto coi polinomi di Lagrange che è

$$p_2(x) = (-4/15) \cdot x^2 + (61/15) \cdot x + (36/5) \approx -0.2\bar{6}x^2 + 4.0\bar{6}x + 7.2.$$

Quindi, se $a = (a_k)_{k=1,\ldots,3}$, abbiamo $p_2(x) = a_1\, x^2 + a_2\, x + a_3$.
In generale, se p_n è il polinomio interpolatore di grado n, e $a = (a_k)$ è il vettore ottenuto utilizzando `polyfit`, allora

$$p_n(x) = a_1\, x^n + a_2\, x^{n-1} + \ldots + a_{n+1}.$$

Interpolazione in Matlab/Octave

Per valutare in un vettore di ascisse $\mathbf{X} = [X_k]_{k=1,\ldots,m}$ un polinomio

$$p_n(x) = a_1\, x^n + a_2\, x^{n-1} + \ldots + a_{n+1}$$

i cui coefficienti sono memorizzati nel vettore $\mathbf{P} = [a_k]_{k=1,\ldots,n+1}$ usiamo il comando `polyval`.
Dall'help:

```
>> help polyval
To get started, select "MATLAB Help" from the
Help menu.
POLYVAL Evaluate polynomial.
Y = POLYVAL(P,X), when P is a vector of length
N+1 whose elements are the coefficients of a
polynomial, is the value of the polynomial
evaluated at X.
...
Y=P(1)*X^N+P(2)*X^(N-1)+...+P(N)*X+P(N+1)
>>
```

Interpolazione in Matlab/Octave

Scriviamo una function MATLAB che:

- Date le n coppie di punti da interpolare, le cui ascisse e ordinate sono, rispettivamente, nei vettori $\mathbf{x} = [x_k]_{k=1,\ldots,n}$ e $\mathbf{y} = [y_k]_{k=1,\ldots,n}$,
- calcoli i coefficienti del polinomio di interpolazione di grado $n-1$ $p_{n-1}(x_k) = y_k$, per $k = 1, \ldots, n$ (vettore coeff)
- e lo valuti sui punti $\mathbf{s} = [s_k]_{k=1,\ldots,m}$, cioè, restituisca come parametro di output il vettore $\mathbf{t} = [t_k]_{k=1,\ldots,m}$ per cui $t_k = p_{n-1}(s_k)$, per ogni k.

```
function t=interpol(x,y,s)
grado=length(x)-1;
coeff=polyfit(x,y,grado);
t=polyval(coeff,s);
```

Formula di Newton alle differenze divise

Siano

- $f : [a, b] \to \mathbb{R}$ una funzione continua;
- x_0, ..., x_n nodi a due a due distinti.

Dati $x_0^*, \ldots, x_k^* \in \{x_0, \ldots, x_n\}$, a due a due distinti, le quantità

$$f[x_0^*, \ldots, x_k^*] = \begin{cases} \dfrac{f[x_1^*, \ldots, x_k^*] - f[x_0^*, \ldots, x_{k-1}^*]}{x_k^* - x_0^*} & \text{se } k > 0 \\ f[x_0^*] = f(x_0^*) & \text{altrimenti,} \end{cases}$$

si chiamano differenze divise.

Differenze divise

Il polinomio p_n, descritto nella formulazione di Newton

$$p_n(x) = \sum_{k=0}^{n} f[x_0, \ldots, x_k](x - x_0) \cdot \ldots \cdot (x - x_{k-1})$$

è tale che

$$p_n(x_k) = f(x_k), \; k = 0, \ldots, n.$$

Risulta quindi rilevante calcolare

$$f[x_0, \ldots, x_k]$$

per $k = 0, \ldots, n$.

Calcolo delle Differenze divise

Formula generale $f[x_0, \dots, x_n] = \dfrac{f[x_1, \dots, x_n] - f[x_0, \dots, x_{n-1}]}{x_n - x_0}$.

Si procede secondo il seguente schema

$$
\begin{array}{c|ccccccc}
x_0 & f[x_0] = f(x_0) & & & & & & \\
 & & \searrow & & & & & \\
 & & & f[x_0, x_1] & & & & \\
 & & \nearrow & & \searrow & & & \\
x_1 & f[x_1] = f(x_1) & & & & f[x_0, x_1, x_2] & & \\
 & & \searrow & & \nearrow & & \searrow & \\
 & & & f[x_1, x_2] & & & & \ddots \\
 & & \nearrow & & & & & \\
x_2 & f[x_2] = f(x_2) & & & & & & f[x_0, \dots, x_n] \\
\vdots & \vdots & & & & \vdots & & \\
 & & & & & & & \iddots \\
\vdots & \vdots & & \vdots & & & \nearrow & \\
 & & & & & f[x_{n-2}, x_{n-1}, x_n] & & \\
\vdots & \vdots & & & \nearrow & & & \\
 & & & f[x_{n-1}, x_n] & & & & \\
 & & \nearrow & & & & & \\
x_n & f[x_n] = f(x_n) & & & & & &
\end{array}
$$

Calcolo delle differenze divise

Esercizio

Si scriva una funzione di nome `polnewton` *che abbia la seguente intestazione e che calcoli i coefficienti della base di Newton con le differenze divise:*

```
function c = polnewton (x,y)
% POLNEWTON Calcola i coefficienti del polinomio interpolatore
%           utilizzando la forma di Newton con le differenze divise
%
% Uso:
% c = polnewton (x,y)
%
% Dati di ingresso:
% x      vettore dei nodi
% y      vettore dei valori della funzione da interpolare nei nodi
%
% Dati di uscita:
% c      vettore colonna dei coefficienti ordinati per indici
%        crescenti (c_0, c_1, ... )
```

Differenze divise e polinomio interpolatore

La function `polnewton.m` deve implementare il seguente algoritmo:

$n1 =$ lunghezza vettore x
if $n1 \neq$ lunghezza vettore y
  Errore. La tabulazione non contiene lo stesso numero di componenti. Stop.
end if
for $i = 1, \dots, n1$
  $ctab(i,1) = y(i)$
end for i
for $j = 2, \dots, n1$
  for $i = 1, \dots, n1 - j + 1$
    $ctab(i,j) = (ctab(i+1, j-1) - ctab(i, j-1))/(x(i+j-1) - x(i))$
  end for i
end for j
c = vettore colonna contenente la prima riga di ctab

NOTA BENE: Per tradurre in Matlab l'ultima istruzione si usi il comando `c = ctab(1,:)` che estrae la riga 1 di una matrice.

Valutazione del polinomio interpolatore

Per valutare il polinomio interpolatore

$$p_n(x^*) = \sum_{k=0}^{n} c_k (x^* - x_0) \cdots (x^* - x_{k-1})$$

in un generico punto x^* si usa l'**algoritmo di Horner** (che richiede $2n$ addizioni e n moltiplicazioni).
Chiamando $d_j = (x^* - x_j)$ si ha che

$$\begin{aligned} u &= p_n(x^*) = c_0 + c_1 d_0 + c_2 d_0 d_1 + \ldots + c_n d_0 d_1 \ldots d_{n-1} = \\ &= (\ldots(((c_n) d_{n-1} + c_{n-1}) d_{n-2} + c_{n-2}) d_{n-3} + \ldots + c_1) d_0 + c_0. \end{aligned}$$

Se $c = (c_0, \ldots, c_n)$ è il vettore di dimensione $n+1$ contenente i coefficienti del polinomio, possiamo valutarlo calcolando dapprima $u = c_n$, poi $u = c_{n-1} + (x^* - x_{n-1}) * u$, poi $u = c_{n-2} + (x^* - x_{n-2}) * u$ e così via:

```
u = c(n)
for j = n - 1, ... , 0 step -1
   u = u * (x - x(j)) + c(j)
end for j
```

Algoritmo di Horner

In Matlab occorre adattare gli indici delle componenti dei vettori dovuto al fatto che non si può partire da zero. Lo pseudocodice già modificato è il seguente:

$n1 =$ lunghezza del vettore x

if $n1 \neq$ lunghezza del vettore **c**

  Errore. Inconsistente numero di componenti. Stop.

$u = c(n1)$

for $j = n1 - 1 \ldots 1, \mathbf{step} - 1$

$u = u * (xstar - x(j)) + c(j)$

end for j

Si scriva la function `hornerN` che implementa l'algoritmo precedente. La sintassi deve essere:

```
[u]= hornerN (x, c, xstar)
```

NB: Si usi il comando `error` per scrivere a video il messaggio di errore e **si renda l'implementazione vettoriale**, in modo da valutare il polinomio simultaneamente su tutte le componenti di xstar, in caso questa variabile sia un vettore.

Scrittura dello script `script_interp.m`

Si vuole determinare il polinomio interpolatore della funzione di Runge

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

calcolato su 11 nodi equispaziati nell'intervallo $I = [-5, 5]$.

Si memorizzino i nodi in un vettore `xn` ed i corrispondenti valori della funzione nel vettore `fxn`.

Si chiami la function `polnewton.m` per ottenere le differenze divise e la function `hornerN` per valutare il polinomio interpolante su un vettore con 201 ascisse equispaziate in I.

Lo script deve inoltre rappresentare graficamente l'interpolante trovato (linea tratteggiata rossa). Nello stesso grafico si rappresentino anche i punti della tabulazione (con un cerchietto rosso) e la funzione data (linea intera nera).

Infine si produca una seconda figura che rappresenti nello stesso intervallo l'asse x e la funzione errore definita come

$$E(x) = f(x) - P(x).$$

La struttura dello script richiesto è mostrata nel lucido successivo.

Scritture dello script `script_interp.m`

```
%==========================================================
% Script per interpolazione con la forma di Newton
% Necessita delle function polnewton e hornerN
%==========================================================
% Definisce la funzione
% ..................................
% Definisce l'intervallo [a,b]
% ..................................
% Definisce il numero di nodi
% ..................................
% Definisce la tabulazione dei punti
% (ascisse equispaziate e ordinate
% come valori della funzione nei nodi definiti)
% ..................................
% Calcola i coefficienti del polinomio con la function polnewton
% ..................................
% Definisce le ascisse per i grafici (201 punti equispaziati)
% ..................................
% Valuta la funzione nelle 201 ascisse
% ..................................
% Valuta il polinomio nelle 201 ascisse (usa hornerN)
% ..................................
% Creazione della figura che contiene i punti della tabulazione
%   la rappresentazione della funzione originale
%   la rappresentazione del polinomio interpolante
% ..................................
% Creazione della figura che contiene la funzione errore
% ..................................
```