Interpolazione ed Approssimazione Polinomiali

Laboratorio di Calcolo Numerico

Federico Piazzon

Email: fpiazzon@math.unipd.it

10 Maggio 2022

Outline

Interpolazione Polinomiale

Richiamo teorico

Esistenza ed unicità dell'interpolante di dati

Per ogni $n \in \mathbb{N}$ e per ogni $x_0, x_1, \ldots, x_n \in \mathbb{R}$ a due a due distinti (i.e., con $x_i \neq x_j$ per ogni $i \neq j$) e $y_0, y_1, \ldots, y_n \in \mathbb{R}$, esiste un unico polinomio $p \in \mathcal{P}_n$ (i.e., sp. vettoriale dei polinomi di grado al più n) interpolante le coppie $(x_i, y_i)_{i=0,1,\ldots,n}$, i.e.,

$$p(x_i) = y_i,\ \forall i = 0, 1, \ldots, n.$$

Vista l'unicità chiameremo p il polinomio interpolatore.

Interpolare funzioni

Ovviamente vale il medesimo risultato se i valori da interpolare y_i sono campionamenti di una funzione (limitata) nei nodi x_i, i.e.,

$$y(x_i) := f(x_i),\ \forall i = 0, 1, \ldots, n.$$

In questo caso diremo $\exists! p \in \mathcal{P}_n$ che interpola f nei nodi x_i.

Osservazioni

- Il grado del polinomio interpolante è al più n
- Fissati i *nodi* $\boldsymbol{X} := \{x_0, x_1, \dots, x_n\}$ (e una base di $\mathcal{P}_n$) l'operazione di interpolazione di dati $\boldsymbol{y} \mapsto p$ o di funzioni $f \mapsto p$ è lineare.
- Le condizioni di interpolazione sono un sistema lineare quadrato: fissata una base $(b_0, b_1, \dots, b_n)$ di $\mathcal{P}_n$, si scrivono come sistema di Vandermonde

$$\begin{bmatrix} b_0(x_0) & b_1(x_0) & \dots b_n(x_0) & \\ b_0(x_1) & b_1(x_1) & \dots b_n(x_1) & \\ \vdots & \vdots & \vdots & \vdots \\ b_0(x_1) & b_1(x_1) & \dots b_n(x_1) & \end{bmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}$$

$$p(x) = \sum_{j=0}^{n} c_j b_j(x)$$

- Esistenza ed unicità discendono dall'invertibilità della matrice di Vandermonde $V(\boldsymbol{b}, \boldsymbol{X}) := [b_j(x_i)]_{i,j=0,1,\dots,n}$.

Interpolazione di Lagrange

Scrivendo il sistema di Vandermonde nella base canonica (i.e., $b_j(x) = x^j$) ed usando la regola di Cramer si può dare una soluzione esplicita del sistema di Vandermonde.

Polinomi fondamentali di Lagrange

Dati $n+1$ nodi distinti $x_0, x_1, \ldots, x_n \in \mathbb{R}$ esistono e sono unici i polinomi $\ell_{0,n}(x), \ell_{1,n}(x), \ldots, \ell_{n,n}(x) \in \mathcal{P}_n$ che soddisfano

$$\ell_{i,n}(x_j) = \begin{cases} 1 & i \neq j \\ 0 & i = j \end{cases}.$$

= e != invertiti

Si ha

$$\ell_{i,n}(x) = \left(\prod_{j=0}^{i-1} \frac{x - x_j}{x_i - x_j} \right) \cdot \left(\prod_{j=i+1}^{n} \frac{x - x_j}{x_i - x_j} \right).$$

Interpolante di Lagrange

Il polinomio $p(x) := \sum_{j=0}^{n} y_j \ell_{j,n}(x)$ è il polinomio interpolante i dati (x_i, y_i).

Posto $x_0, x_1, \ldots, x_n \in [a,b]$, $x \in \mathbb{R}$, $I := conv(x_0, x_1, \ldots, x_n, x)$, $\omega_n(x) := \prod_{i=0}^{n}(x - x_i)$, $E_n(x, f, \boldsymbol{X}) := f(x) - p_n(x)$, dove p_n è il polinomio che interpola f su X, si ha

$$|E_n(x, f, \boldsymbol{X})| \leq \max_{\xi \in I} |f^{(n+1)}(\xi)| \frac{|\omega_n(x)|}{(n+1)!}, \quad \forall f \in \mathcal{C}^{n+1}.$$

errore puntuale su x

Ponendo

$$d_{[a,b]}(f, \mathcal{P}_n) := \min_{p \in \mathcal{P}_n} \max_{x \in [a,b]} |f(x) - p(x)|$$

distanza di f dai polinomi

Si ha

$$\max_{x \in [a,b]} |E_n(x, f, \boldsymbol{X})| \leq \left(1 + \max_{x \in [a,b]} \sum_{i=0}^{n} |\ell_{i,n}(x)|\right) d_{[a,b]}(f, \mathcal{P}_n)$$

fattore di amplificazione

F. e cost. di Lebesgue

lambda piccolo

La funzione $x \mapsto \Lambda_n(x) := \sum_{i=0}^{n} |\ell_{i,n}(x)|$ è detta funzione di Lebesgue, mentre il su massimo

$$\Lambda_n := \max_{x\in[a,b]} \sum_{i=0}^{n} |\ell_{i,n}(x)|$$

è la costante di Lebesgue. Tali quantità danno una stima della stabilità dell'interpolazione: se p interpola (x_i, y_i) e $\tilde{p}$ interpola $x_i, \tilde{y}_i)$ si ha

$$\max_{x\in[a,b]} |p(x) - \tilde{p}(x)| \leq \max_{x\in[a,b]} \sum_{i=0}^{n} |\ell_{i,n}(x)| \max_i |y_i - \tilde{y}_i|$$
$$\leq \Lambda_n \max_i |y_i - \tilde{y}_i|.$$

Interpolazione con Matlab

calcolare i coeff. con `polyfit`

Supponiamo `length(x)=length(y)=n+1`, allora

`c=polyfit(x,y,n)`

Consente di calcolare i coefficienti `c` del polinomio di grado `n` che interpola i valori `y` sui nodi `x` **rispetto ad una scalatura della base canonica ordinata rispetto al grado decrescente**

```
x=[−1,0,1];y=[0,−1,0];n=2;
c=polyfit(x,y,n)

c =

    1.0000   −0.0000   −1.0000
```

Infatti in questo esempio avremo

$$p(x) = \sum_{j=1}^{n+1} c_j x^{n-j+1} = x^2 + 0 - 1$$

valutare un polinomio con polyval

Sia $c \in \mathbb{R}^{n+1}$, allora il comando

```
p=polyval(c,x)
```

valuta il polinomio di grado al più n avente i coefficienti c sulla base canonica ordinata secondo il grado decrescente:

$$p(x) = \sum_{j=1}^{n+1} c_j x^{n-j+1}$$

```
>> p=polyval(c,[2,3])

p =

     3     8
```

L'algoritmo utilizzato è lo schema di Horner (che si basa sull'uso della base canonica!)

`polifit polyval`: vantaggi e limiti

- I comandi `polifit polyval` sono disegnati per interpolare (o più in generale fittare) dati tramite modelli polinomiali di grado basso e su dati di input relativamente piccoli.
- L'uso della base monomiale consente una valutazione estremamente efficiente dei polinomi ma,
- `polyfit` soffre (a causa della scelta della base canonica) di forte instabilità per problemi mal condizionati ($n >> 1$)
- `polyfit` calcola l'interpolante risolvendo il sistema di Vandermonde nella base canonica ($b_j = x^{n-j+1}$), oltre al problema di malcondizionamento la soluzione "automatizzata" di questo ("difficile") problema è una black-box su cui non si ha controllo: matlab potrebbe scegliere di non interpolare ma "fittare" i dati (i.e., minimizzare lo scarto).

Polinomi di Lagrange

Disponibile da scaricare la function LagrangePoly.m, vediamola:

Editor - /home/federico/LagrangePoly.m

LagrangePoly.m | testlagrange.m

```
function L=LagrangePoly(xinterp,xeval)
%-----------------------------------------------------------------------
%
%                                   Ver. 03-05-2021
% L=LagrangePoly(xinterp,xeval) valuta i polinomi di Lagrange
%
% INPUT------------------------------------------------------------------
% xinterp          double [1 x n+1] or [n+1 x 1] nodi di interpolazione
% xeval            double [1 x m] or [m x1] nodi di valutazione
%
% OUTPUT-----------------------------------------------------------------
% L                double [m x n+1] L{i,j} è il j-esimo pol di Lagrange va
%                  lutato sul xeval{i}
%-----------------------------------------------------------------------
xinterp=xinterp(:);xeval=xeval(:);
n=length(xinterp)-1;m=length(xeval);
L=zeros(m,n);
Xei=-(xeval-xinterp');
Xii=xinterp-xinterp';Xii=Xii-diag(diag(Xii))+eye(length(xinterp));
for i=1:n+1
    L(:,i)=prod(Xei(:,[1:i-1,i+1:n+1]),2);
end
L=L./prod(Xii);
```

uso di `LagrangePoly.m`

Visto che $L_{i,j} = \ell_{j,n}(x_i^{(eval)})$ si ha

$$p(x_k^{eval}) = \sum_{j=1}^{n+1} \ell_{j,n}(x_k^{(eval)}) y_j = L_{k,:} * y$$

dunque

```
L=LagrangePoly(xinterp,xeval);
peval=L*yinterp;
```

calcola la valutazione del polinomio interpolante nei punti `xeval`, mentre

```
L=LagrangePoly(xinterp,xeval);
Lambda=sum(abs(L),2);
```

calcola la valutazione della Funzione di Lebesgue.

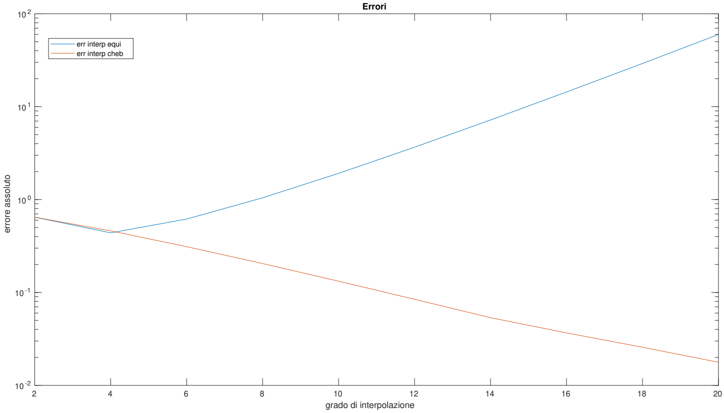
L'esempio di Runge

Interpoliamo a grado $n = 2, 4, \ldots, 20$ la funzione di Runge $f : \mathbb{R} \to \mathbb{R}$, $f(x) := \frac{1}{25x^2+1}$ nell'intervallo $[-1, 1]$ con punti equispaziati e nodi di Chebyshev scalati ottenendo $p_n^{(equi)}$ e $p_n^{(cheb)}$. Valutiamo l'errore con

$$\begin{aligned}
E_n^{(equi)} &:= \max_{k=1,\ldots,m} |p_n^{(equi)}(x_k^{(eval)}) - f(x_k^{(eval)})| \\
&\approx \max_{x\in[-5,5]} |p_n^{(equi)}(x) - f(x)|, \\
E_n^{(cheb)} &:= \max_{k=1,\ldots,m} |p_n^{(cheb)}(x_k^{(eval)}) - f(x_k^{(eval)})| \\
&\approx \max_{x\in[-5,5]} |p_n^{(cheb)}(x) - f(x)|
\end{aligned}$$

Considerando 5001 punti di valutazione equispaziati $x_k^{(eval)}$.

```
Interpolazione della f. di Runge
-----------------------------------------------------
Parametri: a=-5, b=5, 5001 punti di valutazione
-----------------------------------------------------
Risultati:
|grado|  |err interp equi   |    |err interp cheb   |
-----------------------------------------------------
|    2|  |6.462292231266e-01|    |6.005977463736e-01|
|    4|  |4.383571218948e-01|    |4.020167419379e-01|
|    6|  |6.169471659454e-01|    |2.642273670813e-01|
|    8|  |1.045173911784e+00|    |1.708356260403e-01|
|   10|  |1.915647963301e+00|    |1.091534951882e-01|
|   12|  |3.663367283759e+00|    |6.921570780777e-02|
|   14|  |7.194786113357e+00|    |4.660214490813e-02|
|   16|  |1.439360413261e+01|    |3.261337756995e-02|
|   18|  |2.919043772698e+01|    |2.249213032652e-02|
|   20|  |5.982012654045e+01|    |1.533371682593e-02|
```


Errori
err interp equi
err interp cheb
errore assoluto
grado di interpolazione
2
4
6
8
10
12
14
16
18
20
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}

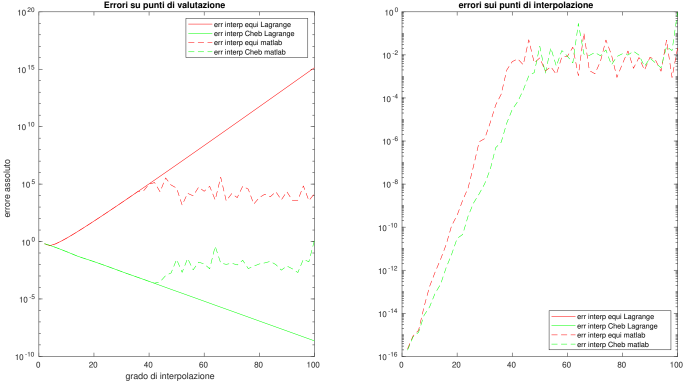
Esercizio proposto

Esercizio 1

Modificando `Runge1.m` ottenere uno script `Runge2.m` che per, $n = 2, 12, 22, \ldots, 102$,

1. non crei le figure dei polinomi interpolanti, nodi e funzione.
2. Crei anche i pol. interpolanti (sulle due famiglie di nodi) calcolate con `polyfit-polyval`
3. Per ogni n calcoli anche l'errore massimo sui nodi di interpolazione (teoricamente nullo!!) per le quattro famiglie di interpolanti salvandolo su 4 vettori
4. Produca anche un grafico dei massimi errori di interpolazione
5. Stampi a video anche una tabella dei massimi errori di interpolazione
6. Stampi su due file le tabelle prodotte.

Risultati: instabilità di `polyfit`



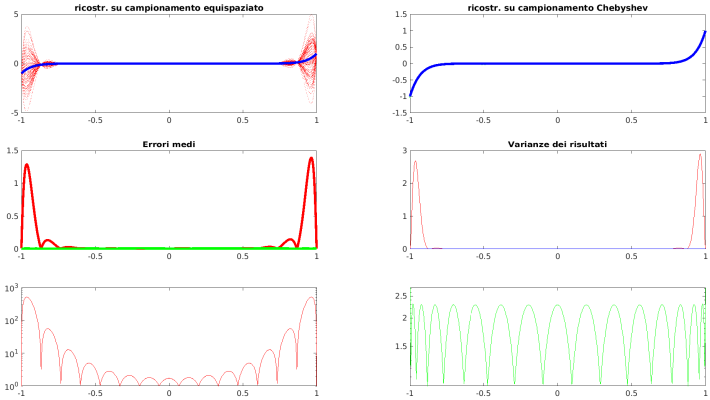
Supponiamo che, per n dato, $f(x) = x^n$, ma che il campionamento di f sia affetto da "rumore bianco" $\tilde{y}_i = f(x_i) + \theta_i$ con $\theta_i \sim \epsilon N(0,1)$ indipendenti. Interpoliamo il campionamento di f (esatto vs affetto da errore) su nodi equispaziati e di Chabyshev, dalla teoria sappiamo che

$$\max_i |\theta_i| \leq |p_n(x, \{y_i\}) - p_n(x, \{\tilde{y}_i\})| \leq \sum_{j=0}^{n} |\ell_{j,n}(x)| \max_i |\theta_i|$$

Il fattore $\Lambda_n(x) := \sum_{j=0}^{n} |\ell_{j,n}(x)|$ funge da moltiplicatore dell'errore aumentando la *media* e la *varianza* statistica.

Eseguiamo `stabinterp.m` con $n = 12$ ed $eps = 10^{-2}$ per vedere un esempio, facendo $M = 1000$ esperimenti per calcolare le statistiche.

Risultati


ricostr. su campionamento equispaziato
ricostr. su campionamento Chebyshev
Errori medi
Varianze dei risultati

Esercizio proposto

Esercizio 2

Si modifichi `stabinterp.m` per ottenere un ciclo su $n = 1 : 20$ ma su una funzione assegnata: $f(x) := exp(\sin(x))$. Invece che i grafici prodotti in `stabinterp.m` si calcolino la media statistica $\mu(n)$ e la varianza statistica $\sigma(n)$ di $E_n := |p_n(2, \{\tilde{y}_i\}) - f(2)|$ e si producano due grafici semilogaritmici di μ e σ.

- `mean(A,dim)` calcola la media di `A` lungo la dimensione `dim`
- `var(A)` calcola la varianza di ogni colonna di `A` (se `A` è una matrice)