lunedì 11 aprile 2016 08:30

INTERPOLAZIONE POLINOMIALE

Dati $n+1$ punti $P_i = (x_i, y_i)$, $i = 0, \dots, n$ associati ai **nodi** x_i che ASSUMIAMO distinti tra loro, il problema della interpolazione polinomiale richiede di trovare un polinomio

$$p_n(x) = a_0 + a_1 x + \dots + a_n x^n$$

di grado al più n che soddisfi le **condizioni di interpolazione**

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

Riscrivendo per esteso le $n+1$ equazioni (1) otteniamo il sistema lineare di $n+1$ equazioni nelle $n+1$ incognite $a_0, a_1, \dots, a_n$ dato da

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

o, in forma matriciale,

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix} \Longrightarrow V\underline{a} = \underline{y}$$

matrice di VANDERMONDE V — $\underline{a}$ — $\underline{y}$

Si dimostra che il determinante di V vale

$$|V| = \det(V) = \prod_{i=0}^{n-1} \prod_{j=i+1}^{n} (x_j - x_i)$$

per cui essendo i nodi distinti tra loro (ossia, $x_i \neq x_j$ per $i \neq j$) è sicuramente $|V| \neq 0$.

Quindi, il sistema lineare $V\underline{a} = \underline{y}$ ha sempre l'unica soluzione $\underline{a} = V^{-1}\underline{y}$. In altre parole, abbiamo provato il seguente notevole risultato.

TEOREMA Se i nodi x_i, $i = 0, \dots, n$ sono distinti ESISTE ed è UNICO il polinomio $p_n(x) = a_0 + a_1 x + \dots + a_n x^n$ di grado al più n che soddisfa le condizioni di interpolazione

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

qualunque siano i numeri y_i, $i = 0, \dots, n$.

Dal punto di vista della effettiva soluzione col calcolatore del sistema lineare $V\underline{a} = \underline{y}$, v'è da dire che si tratta di un problema FORTEMENTE MAL CONDIZIONATO anche per n piccoli. Praticamente, i risultati che si ottengono sono inaccurati già per $n \approx 7$.
Ad esempio preso

$$p(x) = (x-1)^8 = x^8 - 8x^7 + 28x^6 - 56x^5 + 70x^4 - 56x^3 + 28x^2 - 8x + 1$$

ed i nodi $x_i = -3:5$ la function polyfit di Matlab restituisce

```
>> a=polyfit(-3:5,[65536 6561 256 1 0 1 256 6561 65651],8)

a =

 Columns 1 through 7

   1.0029   -8.0114   27.9601  -55.8403   70.1398  -56.5590   27.8973

 Columns 8 through 9

   -7.5893    1.0000
```

ESEMPIO Scriviamo il sistema di Vandermonde per

x_i	x_0	x_1	x_2
	-1	0	1
y_i	2	1	3
	y_0	y_1	y_2

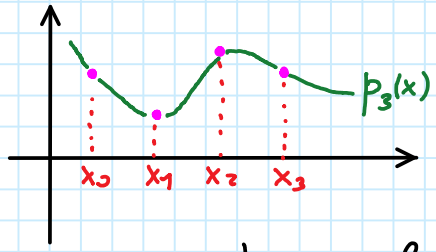
Dato che ci sono 3 nodi, è $n = 2$. Quindi il polinomio di interpolazione è $p_2(x) = a_0 + a_1 x + a_2 x^2$ con a_0, a_1, a_2 che soddisfano il sistema lineare

$$\begin{pmatrix} 1 & x_0 & x_0^2 \\ 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \end{pmatrix} \quad \text{ossia} \quad \begin{pmatrix} 1 & -1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix}$$

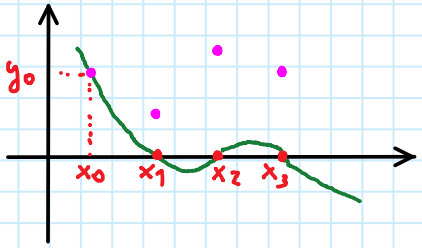
lunedì 11 aprile 2016 08:30

FORMULA DI LAGRANGE DEL POLINOMIO DI INTERPOLAZIONE

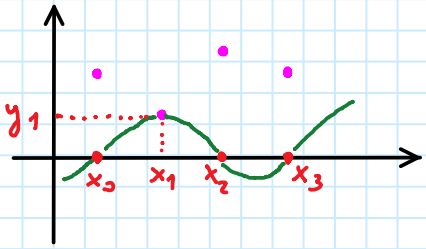
Il polinomio di grado $n = 3$ che interpola i seguenti dati



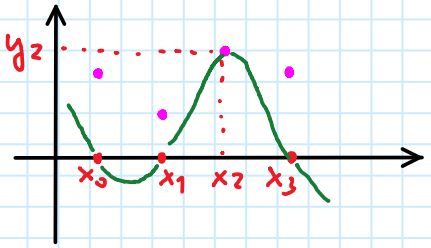
può essere pensato come la somma di 4 polinomi:



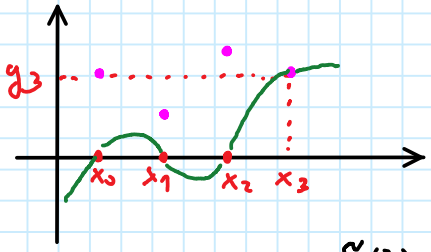
$$\tilde{L}_0^{(3)}(x) = y_0 \frac{\prod_{\substack{i=0 \\ i\neq 0}}^{3}(x-x_i)}{\prod_{\substack{i=0 \\ i\neq 0}}^{3}(x_0-x_i)}$$



$$\tilde{L}_1^{(3)}(x) = y_1 \frac{\prod_{\substack{i=0 \\ i\neq 1}}^{3}(x-x_i)}{\prod_{\substack{i=0 \\ i\neq 1}}^{3}(x_0-x_i)}$$



$$\tilde{L}_2^{(3)}(x) = y_2 \frac{\prod_{\substack{i=0 \\ i\neq 2}}^{3}(x-x_i)}{\prod_{\substack{i=0 \\ i\neq 2}}^{3}(x_0-x_i)}$$



$$\tilde{L}_3^{(3)}(x) = y_3 \frac{\prod_{\substack{i=0 \\ i\neq 3}}^{3}(x-x_i)}{\prod_{\substack{i=0 \\ i\neq 3}}^{3}(x_0-x_i)}$$

È chiaro che $p_3(x) = \tilde{L}_0^{(3)}(x) + \tilde{L}_1^{(3)}(x) + \tilde{L}_2^{(3)}(x) + \tilde{L}_3^{(3)}(x)$. In generale, per $n+1$ nodi risulta

$$p_n(x) = \sum_{i=0}^{n} y_i \cdot L_i^{(n)}(x)$$

FORMA DI LAGRANGE DEL POLINOMIO DI INTERPOLAZIONE

dove $L_i^{(n)}(x)$ è il polinomio di Lagrange associato al nodo i-esimo e vale

$$L_i^{(n)}(x) = \frac{\prod_{\substack{r=0 \\ r\neq i}}^{n}(x-x_r)}{\prod_{\substack{r=0 \\ r\neq i}}^{n}(x_i-x_r)}, \quad i = 0, \dots, n$$

Notare che gli $L_i^{(m)}(x)$ dipendono <u>solo</u> dei nodi x_i.
È immediato verificare che

$$\sum_{i=0}^{m} L_i^{(m)}(x) = 1 \quad \forall x \in \mathbb{R}$$

Infatti, presi i punti $P_i = (x_i, y_i) = (x_i, 1)$, $i = 0, \dots, m$, il polinomio di grado al più m che verifica le condizioni $p_m(x_i) = 1$ per $i = 0, \dots, m$ è chiaramente $p_m(x) = 1 \ \forall x \in \mathbb{R}$. Per l'unicità, questo è anche l'unico polinomio. Allora è

$$1 = p_m(x) = \sum_{i=0}^{m} y_i L_i^{(m)}(x) = \sum_{i=0}^{m} 1 \cdot L_i^{(m)}(x) = \sum_{i=0}^{m} L_i^{(m)}(x).$$

ESEMPIO Scriviamo i polinomi di Lagrange e la forma di Lagrange del polinomio di interpolazione per

	x_0	x_1	x_2
x_i	-1	0	1
y_i	2	1	3
	y_0	y_1	y_2

È $m = 2$;

$$L_0^{(2)}(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} = \frac{(x-0)(x-1)}{(-1-0)(-1-1)} = \frac{1}{2}(x^2 - x)$$

$$L_1^{(2)}(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} = \frac{(x-(-1))(x-1)}{(0-(-1))(0-1)} = -(x^2-1) = -x^2+1$$

$$L_2^{(2)}(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} = \frac{(x-(-1))(x-0)}{(1-(-1))(1-0)} = \frac{1}{2}(x^2+x)$$

$$p_2(x) = y_0 L_0^{(2)}(x) + y_1 L_1^{(2)}(x) + y_2 L_2^{(2)}(x) =$$

$$= 2 \cdot \frac{1}{2}(x^2 - x) + 1 \cdot (-x^2+1) + 3 \cdot \frac{1}{2}(x^2+x) = \frac{3}{2}x^2 + \frac{1}{2}x + 1$$

Come controllo, osserviamo che $L_0^{(2)}(x) + L_1^{(2)}(x) + L_2^{(2)}(x) = 1 \ \forall x \in \mathbb{R}$.

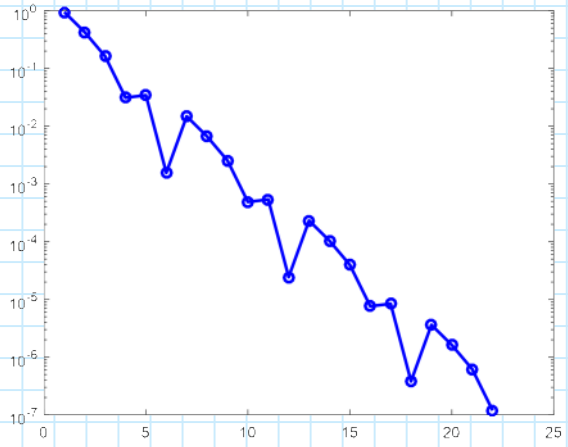
OSS. Con l'esempio precedente abbiamo risolto, indirettamente, il sistema lineare $V\underline{a} = \underline{y}$ visto due pagine fa.

lunedì 11 aprile 2016 08:30

LEZIONE DI LABORATORIO

```
function [a,b,x] = bisezione(f,a,b,toll,kmax)
%BISEZIONE metodo di bisezione
%   [X] = BISEZIONE(F,A,B,TOLL,KMAX) metodo di
%   bisezione per il calcolo dell'unica radice
%   di f presente nell'intervallo [a,b].
%   F e' il nome della funzione.
%   In uscita riporto i vettori a, b, x degli
%   intervalli e la corrispondente soluzione
%   creati dal metodo di bisezione:
%      a   b   x
%      a0  b0  x0        con a0=a, b0=b
%      a1  b1  x1
%      a2  b2  x2
%      ..........
%      ..........
%

%   11/04/2016

x  = 0.5*(a+b); % radice
ex = 0.5*(b-a); % errore

k = 1;
while( ex>=toll && k<=kmax )
        if f( x(k) )*f( a(k) )<0
            % radice nel primo intervallo
            a(k+1) = a(k);
            b(k+1) = x(k);
            %x(k+1) = 0.5*( a(k+1) + b(k+1) );
            %ex     = 0.5*( b(k+1) - a(k+1) );
        elseif f( x(k) )*f( b(k) )<0
            % radice nel secondo intervallo
            a(k+1) = x(k);
            b(k+1) = b(k);
            %x(k+1) = 0.5*( a(k+1) + b(k+1) );
            %ex     = 0.5*( b(k+1) - a(k+1) );
        else
            % trovato radice
            a(k+1) = x(k);
            b(k+1) = b(k);
            %x(k+1) = 0.5*( a(k+1) + b(k+1) );
            %ex     = 0.5*( b(k+1) - a(k+1) );
        end
        x(k+1) = 0.5*( a(k+1) + b(k+1) );
        ex     = 0.5*( b(k+1) - a(k+1) );
        k = k+1;
end

% trasformo i vettori in colonna
a = a(:);
b = b(:);
x = x(:);
```

```
% script di prova del metodo di bisezione
%

% 11/04/2016

% definisco la funzione f
f  = inline('x.^2-1');
xi = 1; % radice da approssimare

% definisco l'intervallo che contiene xi
a  = 0.9;
b  = 3;

% parametri di arresto del metodo
toll = 1E-6;
kmax = 100;

% chiamo la function bisezione
[a,b,x] = bisezione(f,a,b,toll,kmax);

% visualizzo i risultati
[a b x]

% grafico di log10(|errore|)
semilogy(abs(x-xi),'o-b','lineWidth',2)
```



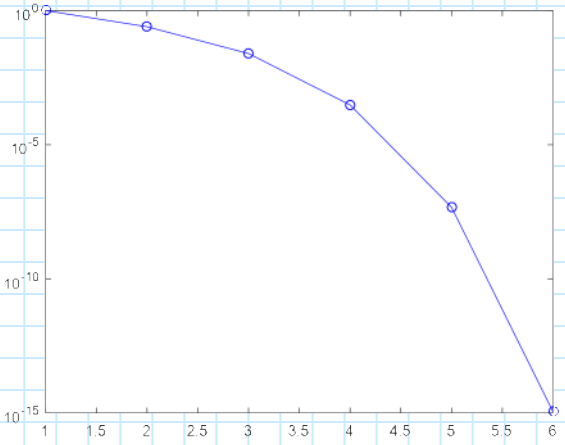
Notare che non è garantita una convergenza monotona dell'errore

```
function x = newton(f,df,x0,toll,kmax)
%NEWTON metodi di Newton per f(x)=0
%
%

% 11/04/2016

x  = x0;
k  = 1;
ex = 2*toll;
while( ex>=toll && k<=kmax)
   x(k+1) = x(k)-f( x(k) )/df( x(k) );
   ex     = abs( x(k+1)-x(k) );
   k      = k+1;
end

% trasformo x in colonna
x = x(:);
```

```
% script per Newton

% definisco la funzione e la sua derivata
f  = inline('x.^2-1'); % funzione
df = inline('2*x');    % derivata
xi = 1;

x0   = 2.0;
toll = 1E-12;
kmax = 100;
x    = newton(f,df,x0,toll,kmax);

% grafico errori
semilogy(abs(x-xi),'o-b')
```

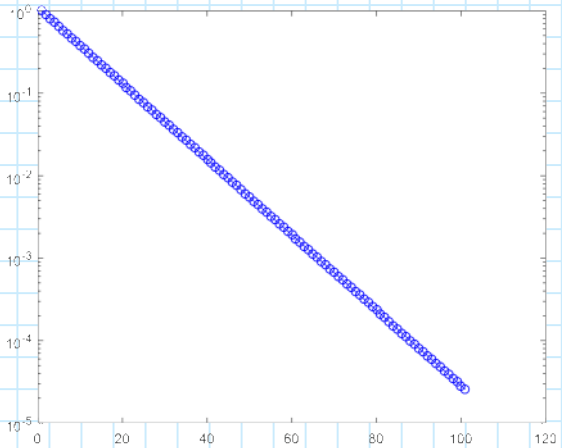
radice semplice => ordine $p = 2$

$f(x) = x^2 - 1$




```
% script per Newton

% definisco la funzione e la sua derivata
f  = inline('(x-1).^9.*log(x)'); % funzione
df = inline('9*(x-1).^8.*log(x)+(x-1).^9./x');    % derivata
xi = 1;

x0   = 2.0;
toll = 1E-12;
kmax = 100;
x    = newton(f,df,x0,toll,kmax);

% grafico errori
semilogy(abs(x-xi),'o-b')
```

radice multipla => $p = 1$



$f(x) = (x-1)^9 \cdot h_m(x)$ ha

$\xi = 1$ con molteplicità $m = 10$