POLINOMIO INTERPOLANTE DI LAGRANGE

Sfrutteremo una base per lo spazio dei polinomi diversa da quella canonica $B = \{1, x, x^2, \dots, x^n\}$

Useremo una base composta dai polinomi fondamentali di Lagrange

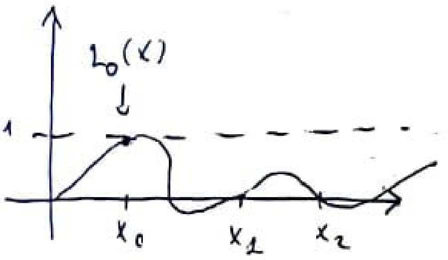
$B = \{L_0(x), L_1(x), \dots, L_n(x)\}$

Cos'è, o meglio come è definito, $L_k(x)$?

$$p_n(x) = c_0 L_0(x) + c_1 L_1(x) + \dots + c_n L_n(x)$$

$$L_k(x) := L_k(x_i) = \begin{cases} 0 & \text{per } k \neq i \quad (\delta_{ki}) \\ 1 & \text{per } k = 1 \quad (\delta_{ki}) \end{cases}$$

Simboli di Kronecker



L_0 vale 1 solo in x_0

L_1 vale 1 solo in x_1

⋮

L_n vale 1 solo in x_n

Come possiamo scrivere L_k?

$L_k(x)$ sicuramente ha k zeri.

$$L_k(x) = a_k (x - x_0)(x - x_1) \cdot \ldots \cdot (x - x_{k-1})(x - x_{k+1}) \cdot \ldots \cdot (x - x_n)$$

↳ impostiamo $L_k(x_k) = 1$

$$1 = L_k(x_k) = a_k (x_k - x_0) \cdot \ldots \cdot (x_k - x_{k-1})(x_k - x_{k+1}) \cdot \ldots \cdot (x_k - x_n)$$

$$\boxed{L_k(x) = a_k \prod_{\substack{j=0 \\ j \neq k}}^{n} \frac{(x - x_j)}{(x_k - x_j)}}$$

Polinomi fondamentali di Lagrange

Non dipendono dalle ordinate, ma solo dai nodi.

Ovvero per x_i, y_i, per $i = 0, \dots, n$ fissando x_i e lasciando libere y_i i polinomi non cambiano!

Ora re-imponiamo di nuovo il passaggio per i punti.

$$p_n(x_k) = y_k \rightarrow \underbrace{c_0 L_0(x_k)}_{0} + \dots + \underbrace{c_k}_{y_k} \underbrace{L_k(x_k)}_{1} + \dots + \underbrace{c_n L_n(x_k)}_{0} = c_k$$

Quindi:

$$\boxed{p_n(x) = \sum_{k=0}^{n} y_k \cdot L_k(x)} \leftarrow \text{Polinomio interpolante di Lagrange}$$

Osserviamo subito che il polinomio interpolante è di grado n. (possiamo subito dedurre che il p. inter. ha grado = # punti - 1 dati)

Sappiamo che:

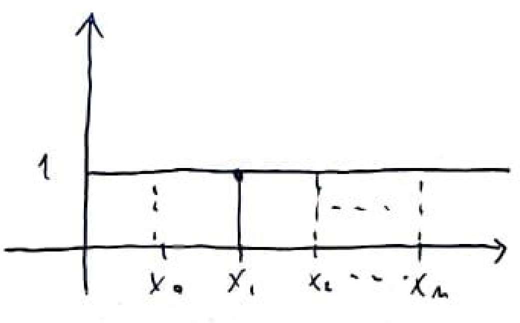
$$\sum_{k=0}^{n} L_k(x) = 1$$

Ma perché?

Consideriamo i seguenti dati

$(x_0, 1), (x_1, 1), \dots, (x_n, 1)$, $x_i \neq x_j$ per $i \neq j$

$$1 = \sum_{k=0}^{n} 1 \cdot L_k(x)$$



Esempio sulla costruzione di un polinomio di Lagrange

X	0	$\frac{1}{2}$	1
Y	1	-1	2

$$p_n(x) = y_0 \cdot L_0(x) + y_1 L_1(x) + y_2 L_2(x)$$

$$p_n(x) \quad 1 \cdot L_0(x) + (-1) L_1(x) + 2 L_2(x)$$

$$L_0(x) = \frac{(x-\frac{1}{2})(x-1)}{(0-\frac{1}{2})(0-1)} = 2(\cancel{2}x - \tfrac{1}{2})(x-1)$$

$$L_1(x) = \frac{(x)(x-1)}{\frac{1}{2}\cdot(-\frac{1}{2})} = -4(x)(x-1)$$

$$L_2(x) = \frac{(x)(x-\frac{1}{2})}{(1-0)(1-\frac{1}{2})} = 2x\cdot(x-\tfrac{1}{2})$$

Possiamo anche verificare i calcoli?

↳ La somma degli x_k deve essere 1.

$$2x^2 - 3x + 1 - 4x^2 + 4x + 2x^2 - x = 1$$

Il risultato è corretto

$$p_2(x) = 1 \cdot L_0(x) - 1\, L_1(x) + 2\, L_2(x)$$
$$= 1\cdot(2(x-\tfrac{1}{2})(x-1)) - 1(-4(x)(x-1)) + 2(2x\cdot(x-\tfrac{1}{2}))$$

Ora possiamo calcolare il polinomio in vari punti per poterlo graficare.

Purtroppo questo genere di calcoli è molto costoso, occorre quindi un modo per evitare ~~dio~~ questo genere di operazioni.

Consideriamo (sempre nel nostro caso):

$$(x-0)(x-\tfrac{1}{2})(x-1) = \omega_3(x)$$

Otteniamo che

- $L_0(x) = \dfrac{\omega_3(x)}{(x-x_0)} = \dfrac{\omega_3(x)}{x-0}$
- $L_1(x) = \dfrac{\omega_3(x)}{(x-x_1)} = \dfrac{\omega_3(x)}{(x-\frac{1}{2})}$
- $L_2(x) = \dfrac{\omega_3(x)}{(x-x_1)} = \dfrac{\omega_3(x)}{(x-1)}$

$\dfrac{\omega_3(x)}{(x-x_k)}$ numeratore di L_k

Come possiamo ottenere il denominatore?

$$\frac{d}{dx}\omega_3(x) = ?$$

$$\omega_3'(x) = (x-\tfrac{1}{2})(x-1) + (x-0)(x-1) + (x-0)(x-\tfrac{1}{2})$$

$$\omega_3'(x_0) = (0-\tfrac{1}{2})(0-1) + 0 + 0 = \tfrac{1}{2}$$ ovvero il den. di L_0

$$L_0(x) = \frac{\omega_3(x)}{(x-x_0)} \cdot \frac{1}{\omega_3'(x_0)}$$

$$L_1(x) = \frac{\omega_3(x)}{(x-x_1)} \cdot \frac{1}{\omega_3'(x_1)}$$

$$L_2(x) = \frac{\omega_3(x)}{(x-x_2)} \cdot \frac{1}{\omega_3'(x_2)}$$

In generale definiamo POLINOMIO NODALE, di grado $n+1$, il polinomio monico:

$$p_n(x) = (x-x_0)(x-x_1)\cdot\ldots\cdot(x-x_n)$$

$$L_k(x) = \frac{\omega_{n+1}(x)}{(x-x_k)\cdot\omega_{n+1}'(x_k)}$$

È il medesimo polinomio fond. solo scritto in modo diverso.

$$p_n(x) = \underbrace{\sum_{k=0}^{n} y_k L_k(x) = \omega_{n+1}(x) \cdot \sum_{k=0}^{n} \frac{y_k}{(x-x_k)\cdot\boxed{\omega_{n+1}'(x_k)}}}_{\text{Formula baricentrica}}$$

rimane costante ad ogni iter!

Quale è l'errore commesso?

Supponiamo che i dati vengano forniti da una funzione. L'errore generato nasce dal fatto che approssimiamo una funzione con un polinomio.

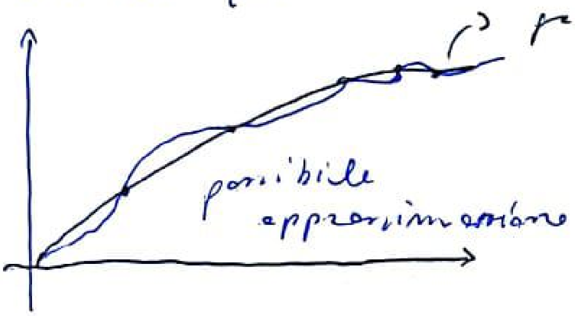
$x_0, \dots, x_n$

$x_i \neq x_j$, per $i \neq j$

$y_0, \dots, y_n$

$y_i = f(x_i)$

Con soli questi dati è difficile stabilire quale è l'errore sul f.



Supponiamo $f \in C^{n+1}_{[a,b]}$ $\rightarrow$ $\left(\begin{array}{l} \text{con 3 p.ti } C^3 \\ \text{con 2 p.ti } C^2 \\ \text{con 4 p.ti } C^4 \\ \vdots \\ \text{con } n \text{ p.ti } C^n \end{array} \right)$

difficile da calcolare

e inoltre non conosciamo ξ!

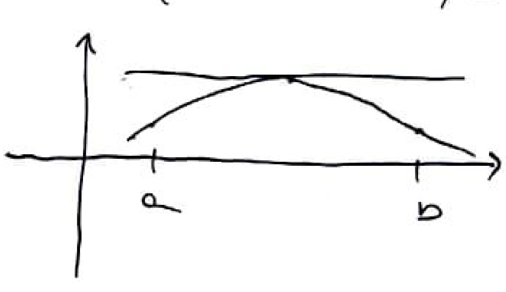
$$f(x) - p_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \cdot \omega_{n+1}(x)$$

Cerchiamo di maggiorare questa quantità.

$$M_{n+1} = \max_{x \in [a,b]} \left| f^{(n+1)}(x) \right|, \quad \left| f(x) - p_n(x) \right| \leq \frac{M_{n+1}}{(n+1)!} \cdot \max_{x \in [a,b]} |\omega_{n+1}(x)|$$

Ora applichiamo varie volte consecutivamente il Th. di Rolle.

f funzione derivabile e continua in $[a,b]$ e se $f(a) = f(b)$, $\exists\, \xi \in (a,b)$ t.c. $f'(\xi) = 0$



$\rightarrow$ Esiste un massimo o un minimo in $[a,b]$

$f(x) - p_n(x)$

$p_n(x_i) = f(x_i)$ per $i = 0, \dots, n$

Da cui

$$\underbrace{f(x_i) - p_n(x_i) = 0}_{\text{ha } n+1 \text{ zeri!}} \quad \text{per } i = 0, \dots, n$$

$f(x) - p_n(x) = R(x) \cdot \omega_{n+1}(x)$

Sfruttiamo una ~~[illegible]~~ funzione ausiliaria g.

$g(t) = f(t) - p_n(t) - R(x)\,\omega_{n+1}(t)$

$g(t) = 0$ in tutte le x_i e x

$g(t) = 0$ per $t = x_0, x_1, \dots, x_n, x$

→ Ci sono $n+2$ p.ti in cui $g(T)$ è zero.

Per il th. di Rolle:

$\exists\ n+1$ p.ti $\xi_0, \dots, \xi_n$ t.c. $g'(\xi_i) = 0$

$\exists\ g'$? nelle nostre ipotesi sicuramente sì!

→ Possiamo quindi riapplicare il th. di Rolle

Ripetiamo questo "processo" n volte

Infine

$\exists\ \xi : g^{(n+1)}(\xi) = 0$

$$g^{(n+1)}(t) = f^{(n+1)}(t) - \underbrace{p_n^{(n+1)}(t)}_{0} - \underbrace{R(x)}_{\text{c'è un coefficiente}} \cdot \underbrace{\omega^{(n+1)}(t)}_{\text{diviene } (n+1)!}$$

dato che p è un polinomio di grado n.

$$g^{(n+1)}(\xi) = 0 = f^{(n+1)} - R(x)\,(n+1)!$$

$$\# := R(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}$$

DISTRIBUZIONE DEI NODI

Si potrebbe pensare che la miglior distribuzione di nodi sia una equidistante, in realtà non si hanno particolari svantaggi o vantaggi, ma esistono "distribuzioni" migliori.

Esempio sui punti equispaziati:

$x_i = x_0 \cdot hi \quad$ con $i = 0, \dots, n$

$\omega_0(x) = (x - x_0)(x - x_1)$

Come possiamo maggiorare questa quantità?

Osserviamo che in $x = \frac{x_0 + x_1}{2}$ $|\omega_2(x)|$ è massimo

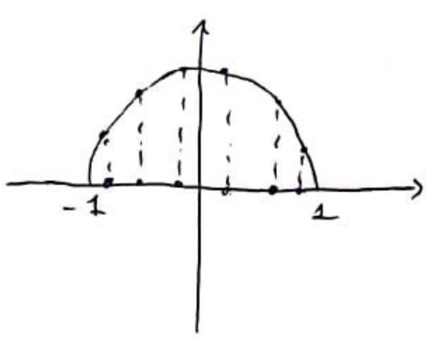
$|\omega_2(x)| = \frac{(x_1 - x_0)^2}{4} = \frac{h^2}{4}$

Se consideriamo n punti equidistanti:

$\underbrace{|\omega_{n+1}(x)| \leq n! \frac{h^{n+1}}{4}}$

(Si può dimostrare per induzione)

DISTRIBUZIONE DI CHEBYSHEV



Prendiamo dei punti equidistanti su una semicirconferenza e infine prendiamo come nodi le proiezioni di quest'ultimi sulla asse delle ascisse.

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

↳ Riduce l'errore sull'oscillazione dei nodi lagrange_multi.m | lagrange.m | test.m

home > antonio > elaborati_matlab > polinomio di Lagrange > lagrange.m > function [f] = lagrange(xdata, ydata, z)

```matlab
function [f] = lagrange(xdata, ydata, z)
    n = length(xdata);
    sm = 0;
    for i=1:n
        pr = 1;
        for j=1:n
            if i~=j
                pr = pr*(z-xdata(j))/(xdata(i) - xdata(j));
            end
        end
        sm = sm + ydata(i)*pr;
    end
    f = sm;

```

⊗ 0 ⚠ 0 Ln 1, Col 1 Spaces: 4 UTF-8 LF MATLAB lagrange.m | lagrange_multi.m | test.m

home > antonio > elaborati_matlab > polinomio di Lagrange > test.m > function [f] = test(xdata,ydata,z)

```
function [f] = test(xdata,ydata,z)
    m = length(z);
    f = zeros(m, 1);
    for i=1:m
        f(i) = lagrange(xdata, ydata, z(i));
    end
end
```

0 0 | Ln 1, Col 1 | Spaces: 4 | UTF-8 | LF | MATLAB

lagrange_multi.m × lagrange.m test.m

home > antonio > elaborati_matlab > polinomio di Lagrange > lagrange_multi.m > function [f] = lagrange_multi(xdata, ydata, z)

```
function [f] = lagrange_multi(xdata, ydata, z)
    n = length(xdata);
    m = length(z);
    f = zeros(m, 1);
    const = ydata;
    for i=1:n
        for j=1:n
            if i~=j
                const(i) = const(i)/(xdata(i) - xdata(j));
            end
        end
    end
    for i=1:m
        for j=1:n
            term = 1;
            for k=1:n
                if j~=k
                    term = term*(z(i)-xdata(k));
                end
            end
            f(i) = f(i) + term*const(j);
        end
    end
```

⊗ 0 ⚠ 0 Ln 1, Col 1 Spaces: 4 UTF-8 LF MATLAB chebyshev.m

home > antonio > elaborati_matlab > polinomio di Lagrange > chebyshev.m > function [distribution] = chebyshev(a, b, k)

```
function [distribution] = chebyshev(a, b, k)
    distribution = zeros(k, 1);
    for i=1:k
        distribution(i) = (a+b)/2 + (b-a)*cos((2*i*pi + pi)/(2*k + 2))/2;
    end
end
```

0 0 Ln 1, Col 1 Spaces: 4 UTF-8 LF MATLAB