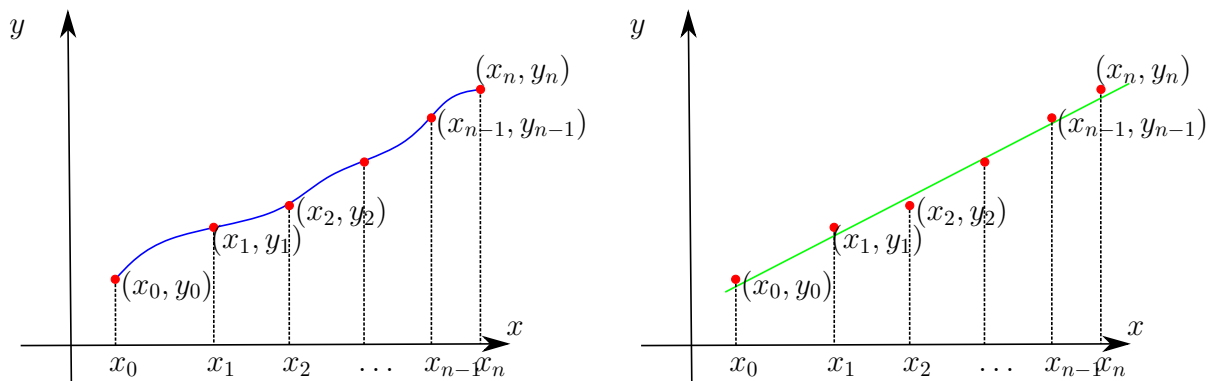


## ▷ Lezione V

# Interpolazione

Vogliamo affrontare il problema che consiste nel trovare, dato un set di punti  $(x_i, y_i)$ , una funzione interpolante ossia una funzione che passa per tutti i punti dati. Alternativamente, possiamo costruire una funzione più semplice, che passi solo sufficientemente vicino a tali punti. Un esempio è dato dai due grafici seguenti, a sinistra viene costruita la funzione interpolante in blu mentre a destra una retta approssimante in verde.



### Interpolazione polinomiale

Date  $n + 1$  coppie di punti  $(x_i, y_i)$  distinte, per  $i = 0, \dots, n$ , cerco un polinomio di grado  $n$  chiamato  $\pi_n \in \mathbb{P}^n$ , con  $\mathbb{P}^n$  lo spazio dei polinomi fino al grado  $n$ , che passi per gli  $n + 1$  punti, ovvero, tale che soddisfi

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

#### Teorema 5.1

Dati  $n + 1$  punti  $(x_i, y_i)$ , con  $i = 0, \dots, n$ , tale che  $x_i \neq x_k$  per  $i \neq k$  allora esiste uno e un solo polinomio  $\pi_n \in \mathbb{P}^n$  tale che

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

**Dimostrazione.** Dimostriamo che il polinomio esiste, e lo facciamo per costruzione. Ciascun elemento dello spazio dei polinomi di grado  $n$  può essere costruito come combinazione lineare di monomi. Scriviamo quindi che

$$\forall p_n \in \mathbb{P}^n : \quad p_n(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$$

Alternativamente possiamo scrivere che lo spazio  $\mathbb{P}^n$  è generato partendo dai monomi, in formula abbiamo

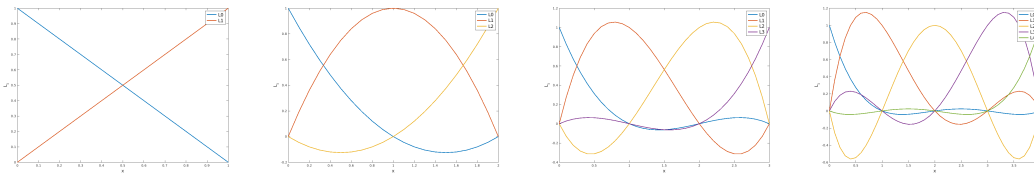
$$\mathbb{P}^n = \text{span}\{1, x, x^2, \dots, x^n\}.$$

Imporre che il polinomio  $p$  passi per gli  $n + 1$  punti genera un sistema di  $n + 1$  equazioni lineari in  $n + 1$  incognite: i coefficienti  $a_i$  (dove i coefficienti della matrice sono le potenze di  $x_i$ ). Se vogliamo evitare di risolvere questo sistema  $(n + 1) \times (n + 1)$ , un approccio alternativo consiste nel costruire una base per i polinomi.

Definisco quindi per ogni  $i = 0, \dots, n$  un polinomio  $\mathcal{L}_i$  di grado  $n$  con le seguenti caratteristiche:

$$\mathcal{L}_i(x_k) = \begin{cases} 1 & \text{se } k = i \\ 0 & \text{se } k \neq i \end{cases} \quad \rightarrow \quad \mathcal{L}_i(x) = \prod_{j=0, j \neq i}^n \frac{x - x_j}{x_i - x_j}$$

tali polinomi sono detti polinomi caratteristici di Lagrange, e hanno la seguente rappresentazione grafica per il grado 1, 2, 3 e 4.



I polinomi  $\mathcal{L}_i$  sono linearmente indipendenti, infatti voglio mostrare che la seguente espressione è valida se e solo se tutti i coefficienti  $\alpha_i = 0$

$$\sum_{i=0}^n \alpha_i \mathcal{L}_i(x) = 0 \quad \xrightarrow{\text{pongo } x=x_j} \quad \sum_{i=0}^n \alpha_i \mathcal{L}_i(x_j) = \alpha_j \mathcal{L}_j(x_j) = \alpha_j = 0 \quad \forall \quad j = 0, \dots, n$$

e quindi costituiscono una base di  $\mathbb{P}^n$ , ovvero possiamo scrivere che  $\mathbb{P}^n$  è costruito anche come

$$\mathbb{P}^n = \text{span}\{\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2, \dots, \mathcal{L}_n\}.$$

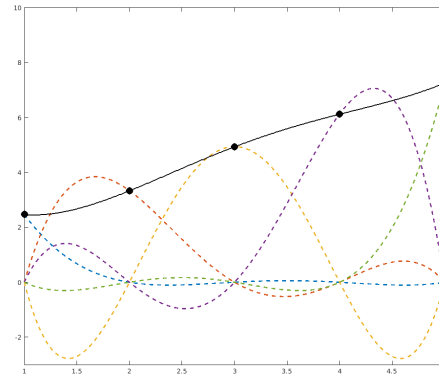
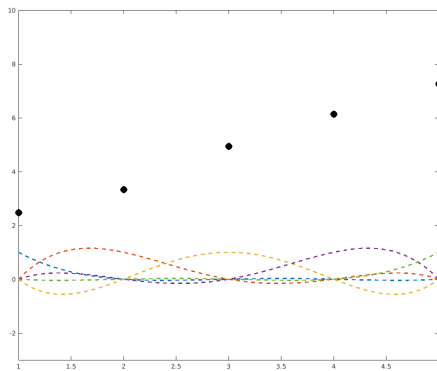
Significa che ogni polinomio  $p_n$  può essere scritto come

$$\forall p_n \in \mathbb{P}^n : \quad p_n(x) = b_0 \mathcal{L}_0(x) + b_1 \mathcal{L}_1(x) + b_2 \mathcal{L}_2(x) + \dots + b_n \mathcal{L}_n(x).$$

Date le proprietà dei polinomi di Lagrange, allora possiamo costruire l'interpolante polinomiale  $\pi_n(x)$  tale che  $\pi_n(x_i) = y_i$  nel seguente modo

$$\pi_n(x) = \sum_{i=0}^n y_i \mathcal{L}_i(x),$$

detto interpolante polinomiale Lagrangiano.



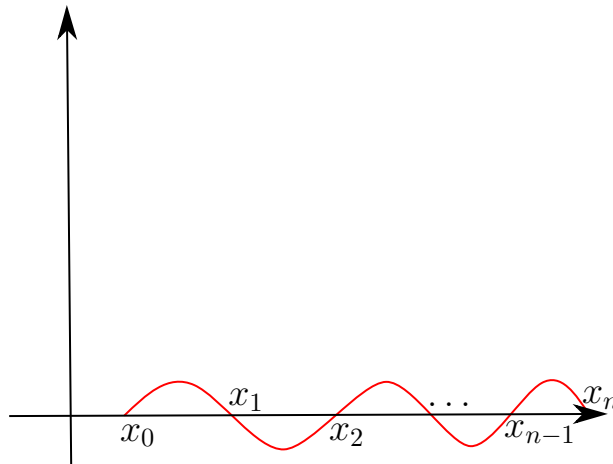
Dimostriamo ora l'unicità, e lo facciamo per assurdo, supponendo che esistano due polinomi distinti di  $\mathbb{P}^n$  che interpolano gli  $n+1$  punti  $(x_i, y_i)$ :

$$\pi_n \in \mathbb{P}^n \quad \text{e anche} \quad \tilde{\pi}_n \in \mathbb{P}^n$$

tali che soddisfano

$$\pi_n(x_i) = y_i \quad \text{e} \quad \tilde{\pi}_n(x_i) = y_i \quad \forall i = 0, \dots, n.$$

Definisco ora un terzo polinomio  $p_n = \pi_n - \tilde{\pi}_n$ , che per costruzione soddisfa:  $p_n \in \mathbb{P}^n$  e anche che  $p_n(x_i) = \pi_n(x_i) - \tilde{\pi}_n(x_i) = 0$ .



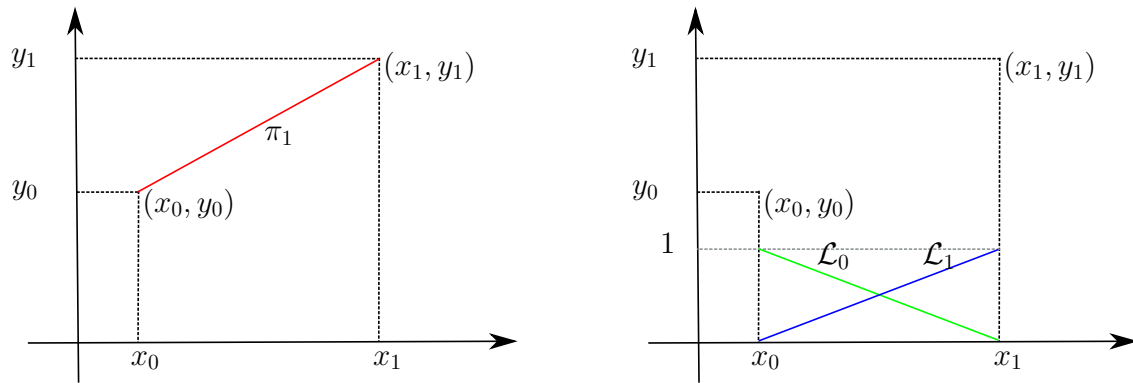
Tuttavia, se un polinomio di grado  $n$  si annulla in  $n+1$  punti deve necessariamente essere il polinomio identicamente nullo, ovvero

$$p_n(x) = 0 \quad \forall x,$$

andiamo quindi in contraddizione con l'ipotesi e necessariamente esiste solamente un polinomio interpolante.  $\square$

**Esempio 5.1**

Consideriamo il caso in cui  $n + 1 = 2$ , ovvero cerco un interpolante di grado 1.



Data l'espressione del polinomio interpolante di Lagrange, abbiamo che

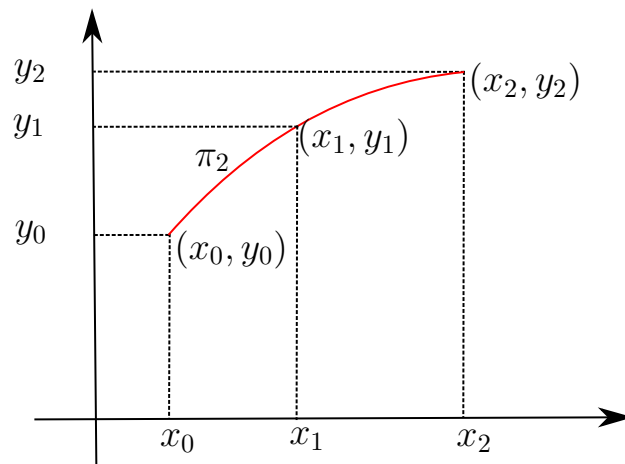
$$\pi_n(x) = \sum_{i=0}^1 y_i \mathcal{L}_i(x) = y_0 \mathcal{L}_0(x) + y_1 \mathcal{L}_1(x)$$

dove ciascun polinomio è dato dall'espressione

$$\mathcal{L}_0(x) = \frac{x - x_1}{x_0 - x_1} \quad \text{e} \quad \mathcal{L}_1(x) = \frac{x - x_0}{x_1 - x_0}.$$

**Esempio 5.2**

Consideriamo ora il caso in cui  $n + 1 = 3$ , ovvero cerco un interpolante di grado 2.



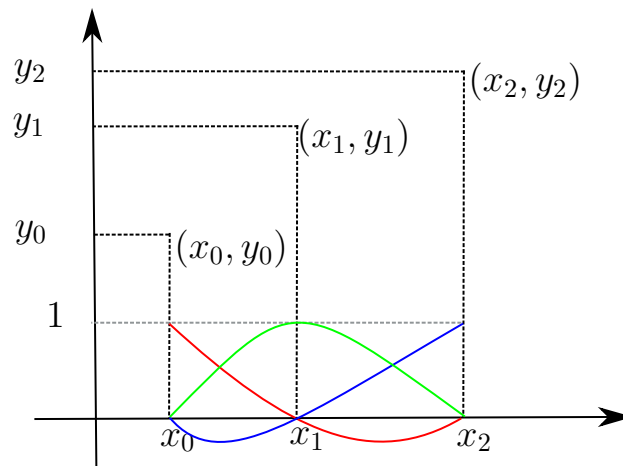
Data l'espressione del polinomio interpolante di Lagrange, abbiamo che

$$\pi_n(x) = \sum_{i=0}^2 y_i \mathcal{L}_i(x) = y_0 \mathcal{L}_0(x) + y_1 \mathcal{L}_1(x) + y_2 \mathcal{L}_2(x)$$

dove ciascun polinomio è dato dall'espressione

$$\mathcal{L}_0(x) = \frac{x - x_1}{x_0 - x_1} \cdot \frac{x - x_2}{x_0 - x_2} \quad \text{e} \quad \mathcal{L}_1(x) = \frac{x - x_0}{x_1 - x_0} \cdot \frac{x - x_2}{x_1 - x_2} \quad \text{e} \quad \mathcal{L}_2(x) = \frac{x - x_0}{x_2 - x_0} \cdot \frac{x - x_1}{x_2 - x_1}$$

Graficamente questi polinomi sono rappresentabili come in figura, dove notiamo che ogni polinomio è pari a uno nel nodo corrispondente, e si annulla negli altri due.



#### Teorema 5.2 - Stima dell'errore per l'interpolazione di Lagrange

Dato un intervallo limitato  $I$ , si considerano  $n + 1$  nodi di interpolazione distinti. Se  $f \in C^{n+1}(I)$  allora per ogni  $x \in I$ , esiste uno  $\xi$  tale che

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

Osserviamo che l'errore è nullo se valutato nei punti di interpolazione, ovvero che  $E_n f(x_i) = 0$ . Questo risultato può essere meglio precisato nel caso di nodi equispaziati, ossia  $x - x_i = h \forall i$ . Nel caso di nodi uniformi risulta evidente che il teorema non garantisce che si abbia convergenza, ovvero in generale

$$\lim_{n \rightarrow \infty} \max_{x \in I} |E_n f(x)| \neq 0.$$

## Stabilità dell'interpolazione

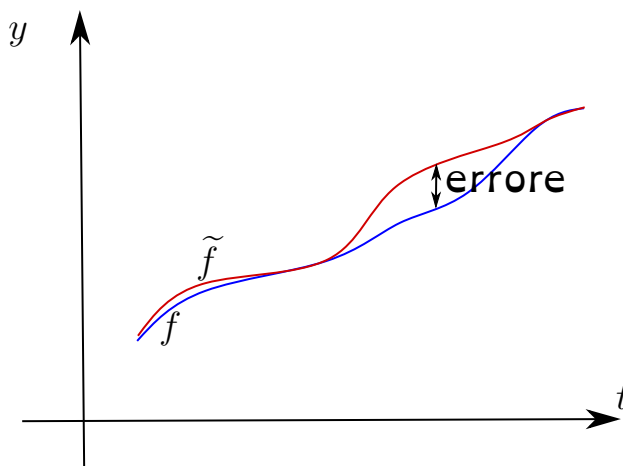
Data una funzione  $f$  definita su  $I$  e dati  $n + 1$  nodi  $x_i$ , considero quindi la sua interpolazione Lagrangiana data da

$$\Pi_n f(x) = \sum_{i=0}^n f(x_i) \mathcal{L}_i(x) = \sum_{i=0}^n y_i \mathcal{L}_i(x),$$

dove  $\Pi_n$  è l'operatore di interpolazione che data una funzione  $f$  restituisce il polinomio interpolatore  $\pi_n$  negli  $n + 1$  punti  $x_i$ . Nella formula precedente abbiamo anche che gli  $\mathcal{L}_i$  sono polinomi caratteristici di Lagrange. Considero ora una funzione  $\tilde{f}$  ottenuta perturbando  $f$ : il suo interpolato è dato dalla seguente espressione

$$\Pi_n \tilde{f}(x) = \sum_{i=0}^n \tilde{f}(x_i) \mathcal{L}_i(x).$$

Ci chiediamo, se la differenza  $f - \tilde{f}$  è piccola allora possiamo anche dire che la differenza  $\Pi_n f - \Pi_n \tilde{f}$  è piccola? In altre parole, l'interpolazione è stabile?



Svolgiamo il seguente calcolo

$$\begin{aligned} \left\| \Pi_n f - \Pi_n \tilde{f} \right\|_{\infty} &= \max_{x \in I} \left| \Pi_n f(x) - \Pi_n \tilde{f}(x) \right| = \max_{x \in I} \left| \sum_{i=0}^n f(x_i) \mathcal{L}_i(x) - \sum_{i=0}^n \tilde{f}(x_i) \mathcal{L}_i(x) \right| \leq \\ &\leq \max_{i=0, \dots, n} \left| f(x_i) - \tilde{f}(x_i) \right| \max_{x \in I} \sum_{i=0}^n |\mathcal{L}_i(x)| \end{aligned}$$

L'ultimo termine, che non dipende da  $f$  ma solo dai valori dei  $\mathcal{L}_i$  nei nodi  $x_i$ , è detto costante di Lebesgue e, per nodi equispaziati, è data da

$$\Lambda_n = \max_{x \in I} \sum_{i=0}^n |\mathcal{L}_i(x)| \approx \frac{2^{n+1}}{en(\log n + \gamma)}$$

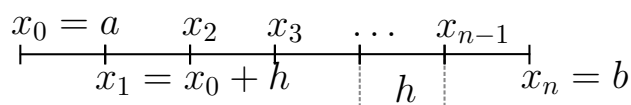
dove  $e \approx 2.71$  è il numero di Nepero e  $\gamma \approx 0.57721$ . Abbiamo, ad esempio che dato  $n = 20$  con nodi equi-spaziati allora la costante di Lebesgue risulta pari a  $\Lambda_{20} = 20000$  ottenendo quindi un'interpolazione instabile. Tuttavia, il valore di  $\Lambda_n$

dipende dal posizionamento dei nodi ed è possibile scegliere una spaziatura dei nodi di interpolazione particolarmente vantaggiosa. Consideriamo i nodi detti di *Chebyshev-Gauss-Lobatto*: la costante di Lebesgue in questo caso risulta  $\Lambda_{20}^c = 3$ , ottenendo quindi un'interpolazione stabile dato che

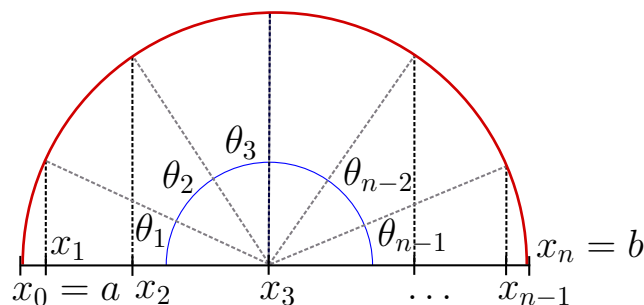
$$\Lambda_n < \frac{2}{\pi} \left( \log n + \gamma + \log \frac{8}{\pi} \right) + \frac{\pi}{72n^2} \approx \frac{2}{\pi} \log n$$

ha una crescita di tipo logaritmico in  $n$ .

Possiamo rappresentare graficamente i nodi equi-spaziati nel seguente modo



I nodi di Chebyshev-Gauss-Lobatto invece si calcolano partendo dalla seguente costruzione



dove abbiamo suddiviso l'intervallo  $[0, \pi]$  in  $n$  intervalli uguali, ottenendo angoli  $\theta_i = \frac{\pi i}{n}$ ; la posizione dei nodi  $\hat{x}_i \in [-1, 1]$  si ottiene quindi come

$$\hat{x}_i = -\cos \frac{\pi i}{n} \quad i = 0, \dots, n.$$

Per interpolare un generico intervallo  $[a, b]$  allora posso mappare gli  $\hat{x}_i$  con la formula

$$x_i = \frac{a+b}{2} + \frac{b-a}{2} \hat{x}_i \quad i = 0, \dots, n.$$

ottenendo quindi che  $x_i \in [a, b]$ . Per  $\hat{x}_0$  abbiamo  $x_0 = a$  ed analogamente  $\hat{x}_n = 1$  implica  $\hat{x}_n = b$ .

#### Lemma 5.1 - Convergenza dell'interpolazione

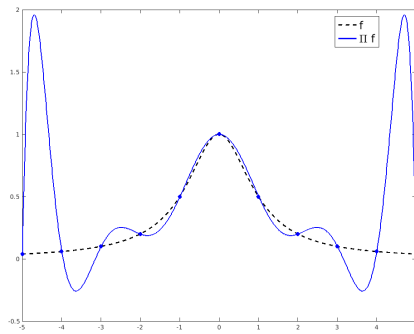
Se  $f \in C^1(I)$  allora considerando i nodi di Chebyshev-Gauss-Lobatto il polinomio interpolatore  $\Pi_n f$  è tale che  $\Pi_n f \rightarrow f$  per  $n \rightarrow \infty$ , ovvero

$$\lim_{n \rightarrow \infty} \max_{x \in I} |E_n f(x)| = 0.$$

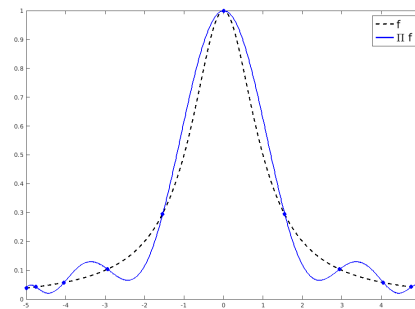
L'interpolazione polinomiale sui nodi non equispaziati di Chebyshev-Gauss-Lobatto, grazie al lemma precedente è convergente.

### Esempio 5.3

Consideriamo la funzione di Runge  $f = \frac{1}{1+x^2}$  sull'intervallo  $[-5, 5]$ . Se cerchiamo di interpolare tale funzione con nodi equispaziati (a) osserviamo che il polinomio interpolante, all'aumentare del numero di nodi esibisce forti oscillazioni agli estremi dell'intervallo. Questo fenomeno è noto come *fenomeno di Runge* ed è chiaramente un effetto indesiderato in quanto l'approssimazione peggiora invece che convergere. Osserviamo invece che, utilizzando i nodi di Chebyshev, il fenomeno è ridotto e l'approssimazione converge.



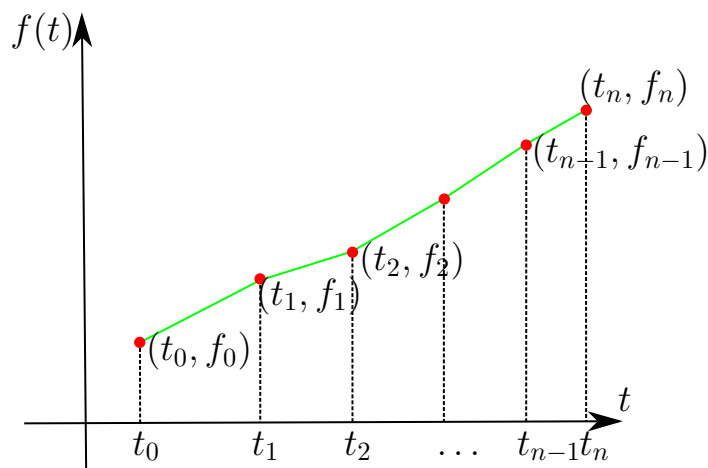
(a)



(b)

## Interpolazione lineare composta

L'idea in questo caso è quella di costruire un interpolante andando a suddividere l'intervallo di interesse in sotto-intervalli e quindi usare dei polinomi di grado basso in ciascuno di essi. Un esempio di interpolazione composta lineare è data dal seguente grafico.

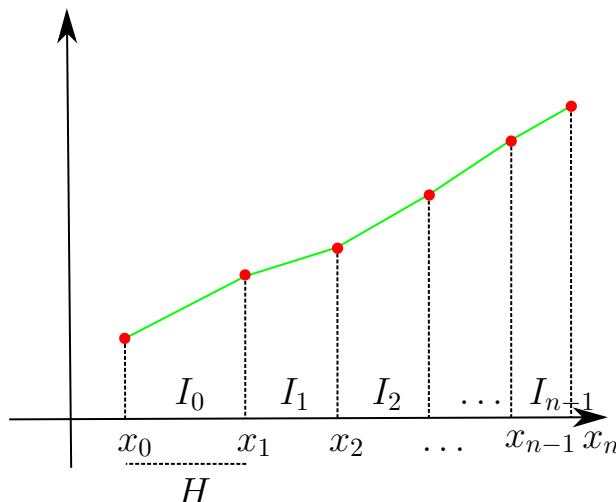


Su ciascun intervallo  $I_i$ , con  $i = 0, \dots, n-1$ , posso fare un'interpolazione locale di grado basso. Data una funzione  $f \in C^2(I)$  e dati  $n+1$  nodi equispaziati  $x_0, \dots, x_n$



nell'intervallo  $I$ , abbiamo

$$\Pi_1^H f(x) = f(x_i) + \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} (x - x_i) \quad \text{per } x \in I_i$$



In questo caso l'errore risulta dato da

$$\max_{x \in I} |E(x)| = \max_{x \in I} |f(x) - \Pi_1^H f(x)| \leq \frac{H^2}{8} \max_{x \in I} |f''(x)|,$$

dove  $H = (b - a)/n$  e con  $I = [a, b]$ . Abbiamo quindi che

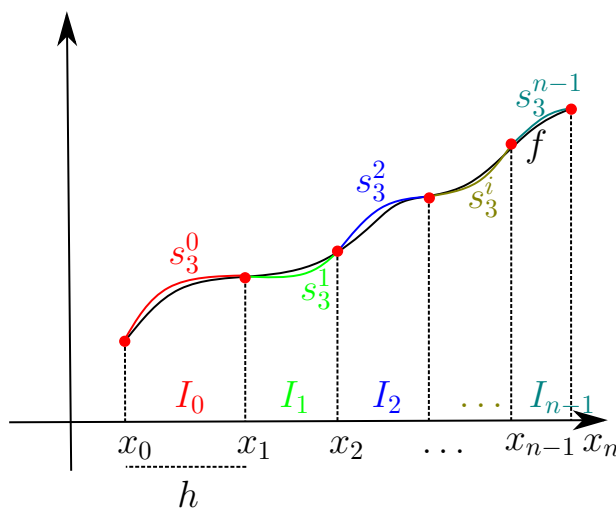
$$\max_{x \in I} |E(x)| \xrightarrow{H \rightarrow 0} 0 \quad \text{dato che} \quad \max_{x \in I} |E(x)| \sim CH^2 \quad \text{dove} \quad C = \frac{1}{8} \max_{x \in I} |f''(x)|.$$

## Splines cubiche

Utilizzando l'interpolazione composta lineare, le derivate, a partire dalla prima, sono discontinue sui nodi. Si può ottenere una interpolazione composta che garantisca un certo grado di regolarità, ovvero tale che

$$\Pi^c f \in C^0 \quad \text{ed anche} \quad \Pi^c f \in C^1 \quad \text{e in aggiunta} \quad \Pi^c f \in C^2.$$

Per questo utilizziamo le splines cubiche che garantiscono una regolarità della soluzione anche tra un sotto-intervallo e quello successivo. Abbiamo quindi



$$\text{su } I_0 \rightarrow s_3^0 \in \mathbb{P}^3(I_0)$$

$$\text{su } I_1 \rightarrow s_3^1 \in \mathbb{P}^3(I_1)$$

$$\text{su } I_2 \rightarrow s_3^2 \in \mathbb{P}^3(I_2)$$

$$\text{su } I_3 \rightarrow s_3^3 \in \mathbb{P}^3(I_3)$$

in cui ognuna delle  $S$  è data da un polinomio cubico,

$$s_3^i(x) = a_0^i + a_1^i x + a_2^i x^2 + a_3^i x^3,$$

mentre l'interpolante globale viene definito da  $s_3(x) = s_3^i(x)$  per  $x \in I_i$ . Dati  $n$  sotto-intervalli, devo determinare  $n$  funzioni cubiche abbiamo quindi  $4n$  incognite ( $n$  sotto-intervalli e 4 parametri per ogni curva per essere determinata) e  $4n - 2$  equazioni che esprimono

$$n + 1 \text{ interpolazione } s_3(x_i) = f(x_i)$$

$$n - 1 \text{ continuità } C^0 \text{ ai nodi } s_3^{i+1}(x_i) = s_3^i(x_i)$$

$$n - 1 \text{ continuità } C^1 \text{ ai nodi } (s_3^{i+1})'(x_i) = (s_3^i)'(x_i)$$

$$n - 1 \text{ continuità } C^2 \text{ ai nodi } (s_3^{i+1})''(x_i) = (s_3^i)''(x_i)$$

Mancano quindi due condizioni per poter chiudere il sistema, vi sono diverse alternative. Una possibilità è quella di impostare la condizione

$$s_3''(x_0) = s_3''(x_n) = 0$$

si parla in questo caso di spline cubiche naturali. Oppure possiamo richiedere che  $s_3'''$  sia continua in  $x_1$  e  $x_{n-1}$ , parliamo quindi di spline not-a-knot.

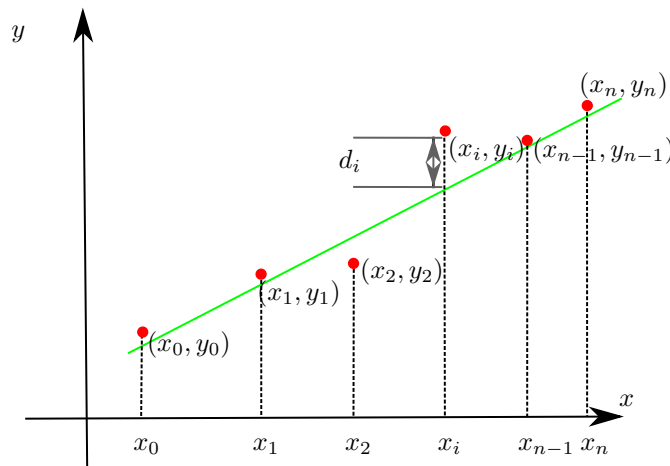
Abbiamo il seguente risultato teorico che ci garantisce la buona approssimazione utilizzando le spline, data da

$$\max_{x \in I} |f^{(r)}(x) - s_3^{(r)}(x)| \leq CH^{4-r}$$

dove  $r$  è il grado di derivazione.

## Metodo dei minimi quadrati

L'interpolazione classica richiede di costruire un polinomio  $\pi_n(x) \in \mathbb{P}^n$  che sia interpolatorio, ovvero tale che  $\pi_n(x_i) = y_i$  per tutti gli  $i = 0, \dots, n$ . Tuttavia, se considerassimo nodi equi-spaziati aumentando l'ordine di approssimazione otterremmo delle oscillazioni. Il metodo dei minimi quadrati permette di costruire un'approssimazione *non interpolante* della funzione  $f$ , con  $y_i = f(x_i)$ , ossia tale che il polinomio ottenuto non passi necessariamente per i nodi  $(x_i, y_i)$ . Si veda la figura seguente



Dati quindi  $n + 1$  punti distinti, cerco un polinomio  $\hat{\pi}$  di grado  $1 \leq m < n$ , che minimizza la distanza dell'approssimante da  $(x_i, y_i)$ . Più precisamente  $\hat{\pi} \in \mathbb{P}^m$  è tale che

$$\sum_{i=0}^n [y_i - \hat{\pi}(x_i)]^2 \leq \sum_{i=0}^n [y_i - p_m(x_i)]^2$$

dove  $p_m$  è un qualunque polinomio di grado  $m$ . Alternativamente, posso scrivere che tra tutti i polinomi  $p_m \in \mathbb{P}^m$  cerco  $\hat{\pi}$  che minimizza la somma dei quadrati delle differenze  $d_i = y_i - \hat{\pi}(x_i)$ .

In Python l'interpolazione Lagrangiana può essere calcolata usando il comando

```
import numpy as np
a = np.polyfit(x, y, n)
```

avendo  $n + 1$  punti e grado polinomiale di approssimazione pari a  $n$ . In uscita otteniamo il vettore  $a = [a_n, a_{n-1}, \dots, a_0]$  tale per cui

$$\pi_n(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n.$$

L'approssimazione ai minimi quadrati è invece ottenuta tramite

```
a = np.polyfit(x, y, m)
```

avendo  $n+1$  punti e grado polinomiale di approssimazione pari a  $m < n$ . Nel caso si volesse calcolare la retta di regressione lineare allora si può utilizzare il comando precedente prendendo  $m = 1$ .

**Esempio 5.4**

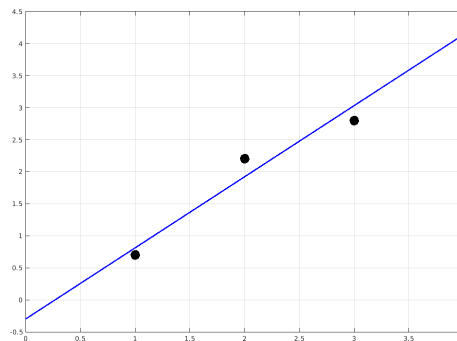
Consideriamo i seguenti punti  $(x_i, y_i)$  con  $x = [1, 2, 3, 4]$ ,  $y = [0.7, 2.2, 2.8, 4.2]$ . Vogliamo calcolare la retta approssimante ai minimi quadrati, di equazione  $\hat{\pi}(x) = a_0 + a_1x$ . Notiamo che, imponendo l'appartenenza dei 4 punti alla retta otteniamo 4 equazioni per due sole incognite, ossia il seguente sistema rettangolare  $Xa = y$  con

$$X = \begin{bmatrix} 1 & x_0 \\ 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \end{bmatrix}, \quad a = \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}, \quad y = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

Se utilizziamo il comando

```
import numpy as np
a = np.linalg.lstsq(X, y, rcond=None)[0]
```

in Python otteniamo una soluzione, nonostante il sistema sia rettangolare, in particolare  $a_0 = -0.3$ ,  $a_1 = 1.11$  ossia l'intercetta all'origine e la pendenza della retta approssimante, in figura.



Tale soluzione corrisponde alla soluzione del sistema  $X^T X a = X^T y$  in cui, premoltiplicando per  $X^T$  otteniamo un sistema quadrato  $2 \times 2$ . Il codice equivalente a quello di prima è dato da

```
a = np.linalg.solve(X.T @ X, X.T @ y)
```

Detto  $d$  il vettore  $y - Xa$ , che contiene le distanze fra i punti dati e la retta ai minimi quadrati, il prodotto  $d^T d$  fornisce la somma dei quadrati di tali distanze; dimostrare per esercizio che la soluzione del sistema  $X^T X a = X^T y$  corrisponde alla ricerca del minimo di  $d^T d$ .