1 Lezione del 14-04-25

Abbiamo finora presupposto di avere una serie di punti (x_j, y_j), con $j = 0, ..., k$, tratti da una funzione $f : [a, b] \to \mathbb{R}$, cioè definita su un certo intervallo $[a, b]$, con $x_j \in [a, b]$ e $y_j = f(x_j) \in \mathbb{R}$. Da questi punti volevamo ricavare un polinomio $p(x)$ di grado al più k tale che questo interpolasse i punti, cioè verificasse:

$$p(x_j) = y_j$$

Da questa configurazione avevamo visto più metodi per il calcolo effettivo di $p(x)$, e avevamo ipotizzato che il modo migliore per aumentarne l'accuratezza, cioè ridurre il massimo della funzione:

$$r(x) = f(x) - p(x), \quad x \in [a, b]$$

fosse aumentare k.

Questo però aveva diversi aspetti negativi:

- Avere k maggiore significa aumentare i punti campionati, cosa che potrebbe non essere sempre fattibile;
- Non è detto che k maggiore aumenti l'accuratezza: avevamo visto infatti il *fenomeno di Runge* agli estremi, per cui il limiter dell'errore all'aumentare di k non era zero, cioè:

  $$\lim_{k \to +\infty} |r(x)| \neq 0$$

  Avevamo visto un modo per migliorare questa situazione, che era usare nodi non equispaziati ma equispaziati su una circonferenza e proiettati sull'asse reale: questi erano i *nodi di Chebyshev*.

Abbiamo in generale, però, che aumentare il grado k non è la situazione ottimale, e si preferirebbe continuare con funzioni polonomiali di grado basso. Introduciamo per questo esatto motivo l'**interpolazione polinomiale a tratti**.

1.1 Interpolazione polinomiale a tratti

Decidiamo quindi di spezzare il dominio $[a, b]$ in sottointervalli di tipo $[x_{i-1}, x_i]$ e di usare in ogni sottointervallo polinomi di grado basso. In ogni sottointervalloconsideriamo quindi polinomi $p_i(x)$ di grado al più $s < k$ tali che $p_i(x)$ interpola f in x_{i-1} e x_i, cioè:

$$p_i(x) : [x_{i-1}, x_i] \to \mathbb{R}, \quad p_i(x_{i-1}) = y_{i-1}, \quad p_i(x_i) = y_i$$

Vediamo quindi alcuni casi particolari di questo tipo di interpolazione al variare di s, presa la tabella di punti:

	$\mathbf{x_j}$	$\mathbf{y_j}$
P_0	-13	5
P_1	-4	-3
P_2	1	2
P_3	13	-1

1.1.1 s = 1: interpolazione lineare a tratti

Nel caso più semplice l'interpolazione lineare a tratti si riduce al definire una serie di rette che collegano ogni coppia di punti x_{i-1}, x_i, in forma:

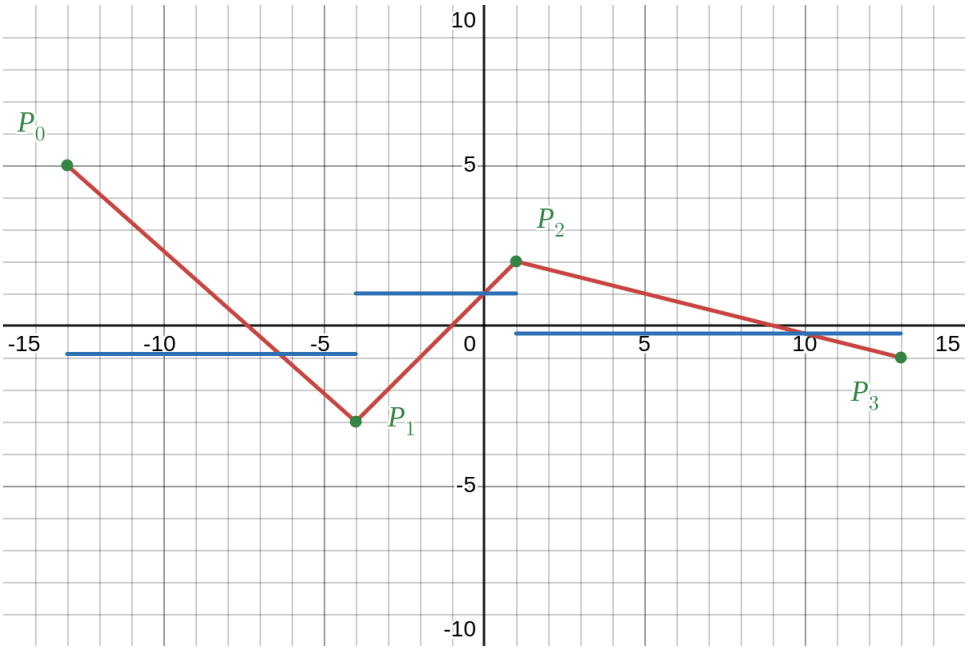
$$l_i(x) = m_i(x) + q_I, \quad i = 1, ..., k$$

In particolare, per i 4 punti che abbiamo preso vorremo definire:

$$l(x) = \begin{cases} l_1(x) = \frac{y_1 - y_0}{x_1 - x_0}(x - x_0) + y_0, & x_0 \leq x < x_1 \\ l_2(x) = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) + y_1, & x_1 \leq x < x_2 \\ l_3(x) = \frac{y_3 - y_2}{x_3 - x_2}(x - x_2) + y_2, & x_2 \leq x \leq x_3 \end{cases}$$

direttamente prendendo i fasci di rette in ogni punto x_0, x_1, x_2 (tutti tranne l'ultimo) e imponendo i coefficienti angolari.

Sul grafico, questo tipo di interpolazione (assieme alla derivata prima) avrà l'aspetto:



da dove si nota chiaramente che con questo approccio la continuità non è assicurata nemmeno al primo grado.

1.1.2 s = 2: spline quadratiche

Una soluzione più "liscia" si può avere sfruttando le cosiddette **spline quadratiche**:

Definizione 1.1: Spline quadratica

Dati $k+1$ punti (x_j, y_j) in un intervallo $[a, b]$ con $j = 0, ..., k$ si definisce spline quadratica ($s = 2$) relativa ai $k+1$ punti la funzione definita a tratti $S_2(x)$ che rispetta le condizioni:

1. $S_2(x)$ è un polinomio di grado 2 se ristretto ad un intervallo $[x_{i-1}, x_i]$;
2. $S_2(x_i) = y_i$ per ogni punto $i = 0, ..., k$;
3. $S_2(x) \in C^1([a, b])$.

In questi termini, l'interpolazione lineare a tratti vista finora rappresenta una sorta di interpolazione per *"spline lineari"*, mentre nella prossima sezione vedremo le *spline cubiche*. Vediamo quindi che le condizioni imposte equivalgono a dire:

$$\begin{cases} p_i(x_i) = y_i, \quad i = 1, ..., k \\ p_i(x_{i-1}) = y_{i-1}, \quad i = 1, ..., k \\ p_i'(x_i) + p_{i+1}'(x_i), \quad i = 1, ..., k-1 \end{cases}$$

cioè abbiamo complessivamente $3k - 1$ condizioni. Di contro, guardando alla prima condizione si ha che ogni tratto dell'interpolazione avrà forma:

$$q_i(x) = a_i x^2 + b_i x + c_i, \quad i = 1, ..., k$$

cioè $3k$ parametri complessivi.

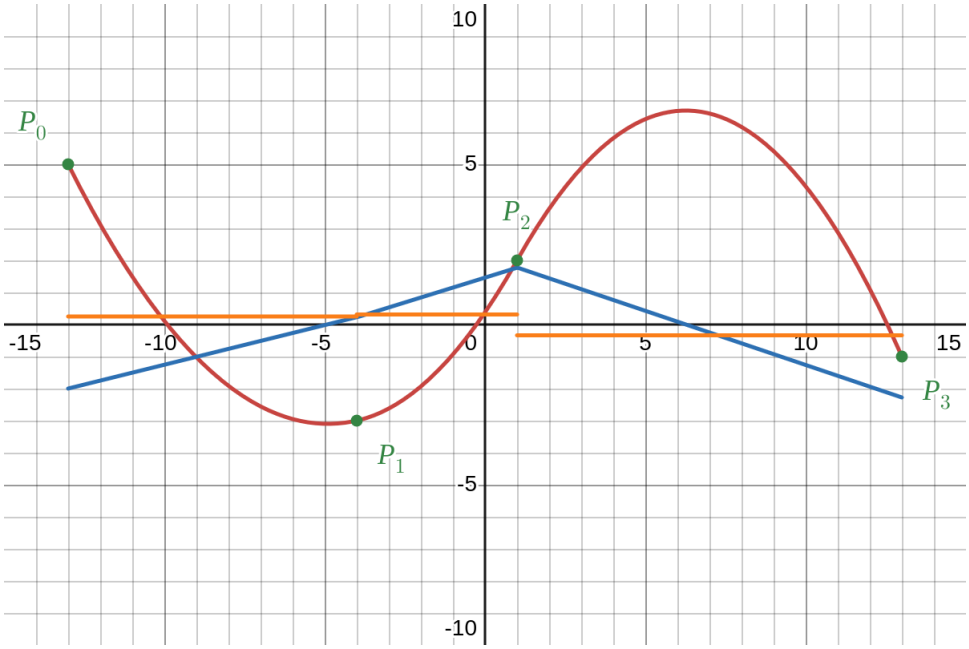
Visto che $3k$ parametri per $3k-1$ condizioni sono troppi, dobbiamo introdurre un'altra condizione: diciamo che la derivata in x_0 deve essere uguale ad un valore fisso d_0.

A questo punto basterà trovare a_i, b_i e c_i in funzione di x_{i-1}, x_i e la derivata in x_{i-1} (che chiamiamo d_{i-1}), che saranno:

$$\begin{cases} a_i = \frac{y_i - y_{i-1}}{(x_{i-1} - x_i)^2} + \frac{d_{i-1}}{x_{i-1} - x_i} \\ b_i = d_{i-1} - 2a_i x_{i-1} \\ c_i = y_i - a_i x_i^2 - b_i x_i \end{cases}$$

Calcolando tali valori per ognuno dei k tratti dell'interpolazione si trovano i parametri di ogni quadratica interpolante.

Facendo ciò sull'esempio di prima, preso $d_0 = -2$ e tracciando il grafico, si ha (assieme alla derivata prima e seconda):



da dove notiamo quindi di aver guadagnato la continuità al primo grado, ma ancora non al secondo. In verità le spline quadratiche non sono molto utili per via delle limitazioni che ci impongono, e vengono usate di rado in contesti reali. Passiamo quindi a polinomi di grado più alto, molto più di largo uso, cioè $s = 3$.

1.1.3 $s = 3$: spline cubica

Una soluzione ancora più "liscia" si può quindi avere sfruttando le cosiddette **spline cubiche**. Queste sono il tipo più comune di spline, e vengono usate in svariati contesti, fra cui ad esempio per tracciare linee "smussate" in computer grafica.

Definiamo quindi:

Definizione 1.2: Spline cubica

Dati $k+1$ punti (x_j, y_j) in un intervallo $[a, b]$ con $j = 0, ..., k$ si definisce spline cubica ($s = 3$) relativa ai $k+1$ punti la funzione definita a tratti $S_3(x)$ che rispetta le condizioni:

1. $S_3(x)$ è un polinomio di grado 3 se ristretto ad un intervallo $[x_{i-1}, x_i]$;
2. $S_3(x_i) = y_i$ per ogni punto $i = 0, ..., k$;
3. $S_3(x) \in C^2([a, b])$.

Vediamo quindi che le condizioni imposte equivalgono a dire:

$$\begin{cases} p_i(x_i) = y_i, & i = 1, ..., k \\ p_i(x_{i-1}) = y_{i-1}, & i = 1, ..., k \\ p_i'(x_i) + p_{i+1}'(x_i), & i = 1, ..., k-1 \\ p_i''(x_i) + p_{i+1}''(x_i), & i = 1, ..., k-1 \end{cases}$$

con una riga in più di condizioni rispetto al caso quadratico dato dal salto da continuità C^1 a C^2, cioè abbiamo complessivamente $4k-2$ condizioni. Di contro, guardando alla prima condizione si ha che ogni tratto dell'interpolazione avrà forma:

$$c_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i, \quad i = 1, ..., k$$

cioè $4k$ parametri complessivi.

Come nel caso quadratico, visto che $4k$ parametri per $4k-2$ condizioni sono troppi, dovremo introdurre 2 nuove condizioni, che modificheranno il tipo di problema che andremo a risolvere:

- **Spline naturale:** si considera questo tipo di spline quando non si ha alcun tipo di informazione oltre i punti da interpolare. In questo caso si assume quindi:

  $$p_1''(x_0) = p_k''(x_k) = 0$$

  cioè derivata seconda nulla agli estremi di interpolazione $x_0 = a$ e $x_k = b$ di $[a, b]$;

- **Spline periodica:** quando si ritiene che la funzione da approssimare sia periodica, si possono adottare le condizioni:

  $$p_1''(x_0) = p_k''(x_k), \quad p_1'(x_0) = p_k'(x_k)$$

- **Spline vincolata:** infine, se si conoscono le derivate prime negli estremi di interpolazione $x_0 = a$ e $x_k = b$ si possono imporre le condizioni:

  $$p_1'(x_0) = f'(x_0), \quad p_k'(x_k) = f'(x_k)$$

Un altro tipo di spline vincolata potrebbe essere quello da usare nel caso in cui si conoscono derivata prima e seconda nel primo estremo di interpolazione $x_0 = a$. In questo caso si impongono le condizioni:

$$p_1'(x_0) = f'(x_0) \quad p_1''(x_0) = f''(x_0)$$

Questo equivale al caso che abbiamo considerato per la spline quadratica (chiamando $f'(x_0) = d_0$ e $f''(x_0) = s_0$), salvo aver dovuto introdurre un ulteriore grado di derivazione.

In ogni caso, per punti equispaziati i può dimostrare il seguente teorema:

Teorema 1.1: Esistenza di spline cubiche

Presi $x_0, ..., x_k$ nodi equispaziati su un intervallo $[a, b]$, cioè:

$$x_i = a + ih, \quad h = \frac{b-a}{k}, \quad i = 0, ..., k$$

e le valutazioni $y_0, ..., y_k$ di una certa funzione $f(x)$ nei nodi x_i, esiste unica la spline naturale $S_3(x)$ passante per i punti (x_i, y_i), $i = 0, ..., k$.

Vediamo quindi due procedimenti di calcolo per due dei casi appena definiti, cioè il primo (spline naturale) e l'ultimo (spline vincolata in x_0). Le formule per il calcolo degli altri due casi saranno in qualche modo ricavate in modo equivalente a quelle del primo caso.

- Un primo approccio può essere quello di risolvere ognuno dei k sottoproblemi di interpolazione fra x_{i-1} e x_i applicando l'interpolazione di Hermite, e portandosi dietro le derivate prime m_i (che ancora non conosciamo), interpretate come $f'(x_i)$. Si determinano quindi gli m_i imponendo le condizioni sulla derivata seconda agli estremi (abbiamo detto la *condizione naturale*).

  Presi quindi punti equispaziati come sopra, cioè:

  $$x_i = a + ih, \quad h = \frac{b-a}{k}, \quad i = 0, ..., k$$

  questo procedimento porta alla formazione di spline del tipo:

  $$p_i(x) = \left[f(x_{i-1}) + \left(m_{i-1} + \frac{2f(x_{i-1})}{h}\right)(x - x_{i-1})\right]\left(\frac{x - x_i}{h}\right)^2 + \left[f(x_i) + \left(m_i - \frac{2f(x_i)}{h}\right)(x - x_i)\right]\left(\frac{x - x_{i-1}}{h}\right)^2$$

  direttamente dall'interpolazione di Hermite, con gli m_i soluzioni del seguente sistema lineare:

  $$\begin{pmatrix} 2 & 1 & 0 & ... & 0 \\ 1 & 4 & 1 & ... & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & ... & 1 & 4 & 1 \\ 0 & ... & 0 & 1 & 2 \end{pmatrix} \begin{pmatrix} m_0 \\ m_1 \\ m_2 \\ \vdots \\ m_k \end{pmatrix} = \frac{3}{h} \begin{pmatrix} f(x_1) - f(x_0) \\ f(x_2) - f(x_0) \\ f(x_3) - f(x_1) \\ \vdots \\ f(x_{k-1}) - f(x_{k-2}) \\ f(x_k) - f(x_{k-1}) \end{pmatrix}$$

da dove si nota la matrice A a *"striscia"*, data dall'interdipendenza fra i tratti della spline.

Per la dimostrazione seguiremo i passaggi in quest'ordine:

1. Prima troviamo una forma per il polinomio interpolante di Hermite $p_i(x)$ nei nodi x_{i-1} e x_i, noti i valori y_{i-1}, y_i e le derivate prime m_{i-1}, m_i;
2. Poi calcoliamo la forma delle derivate seconde del polinomio interpolante agli estremi, in modo da preparare il prossimo passaggio;
3. Imponiamo le condizioni della spline cubica naturale, cioè derivate seconde agli estremi nulle e continuità C^2 nei punti intermedi.

Procediamo quindi con la dimostrazione.

1. Vorremo interpolare ogni tratto x_{i-1}, x_i con un polinomio $p_i(x)$ di terzo grado che rispetti:
$$\begin{cases} p_i(x_{i-1}) = f(x_{i-1}) \\ p_i(x_i) = f(x_i) \\ p_i(x_{i-1}) = m_{i-1} \\ p_i(x_i) = m_1 \end{cases}$$
dove i $k+1$ valori $m_0, m_1, ..., m_k$ sono le derivate prime che ancora non conosciamo. Avremo quindi da Hermite che vorremmo prendere le basi di Lagrange, che posto $x_i - x_{i-1} = h$ (come si ha da nodi equispaziati) sono:
$$e_{i-1}(x) = \frac{x - x_i}{x_{i-1} - x_i} = -\frac{x - x_i}{h}, \quad e_i(x) = \frac{x - x_{i-1}}{x_i - x_{i-1}} = \frac{x - x_{i-1}}{h}$$
di cui calcoliamo subito le derivate, in quanto ci serviranno in seguito:
$$e'_{i-1}(x) = -\frac{1}{h}, \quad e'_i(x) = \frac{1}{h}, \quad e''_{i-1}(x) = e''_i(x) = 0$$
Avremo quindi le basi di Hermite:
$$h_{0,i-1}(x) = (1 - 2e'_{i-1}(x_{i-1})(x - x_{i-1}))e^2_{i-1}(x) = \left(1 + 2\frac{x - x_{i-1}}{h}\right)\left(\frac{x - x_i}{h}\right)^2$$
$$h_{0,i}(x) = (1 - 2e'_i(x_i)(x - x_i))e^2_i(x) = \left(1 - 2\frac{x - x_i}{h}\right)\left(\frac{x - x_{i-1}}{h}\right)^2$$
$$h_{1,i-1}(x) = (x - x_{i-1})e^2_{i-1}(x) = (x - x_{i-1})\left(\frac{x - x_i}{h}\right)^2$$
$$h_{1,i}(x) = (x - x_i)e^2_i(x) = (x - x_i)\left(\frac{x - x_{i-1}}{h}\right)^2$$
da cui il polinomio interpolante finale, raccogliendo le basi quadrate comuni e qualche altro termine, è:
$$p_i(x) = h_{0,i-1}(x)\, y_{i-1} + h_{0,i}(x)\, y_i + h_{1,i-1}(x)\, m_{i-1} + h_{1,i}(x)\, m_i$$
$$= \left(1 + 2\frac{x - x_{i-1}}{h}\right)\left(\frac{x - x_i}{h}\right)^2 y_{i-1} + \left(1 - 2\frac{x - x_i}{h}\right)\left(\frac{x - x_{i-1}}{h}\right)^2 y_i$$

$$+(x-x_{i-1})\left(\frac{x-x_i}{h}\right)^2 m_{i-1}+(x-x_i)\left(\frac{x-x_{i-1}}{h}\right)^2 m_i$$
$$=\left(y_{i-1}+\left(m_{i-1}+\frac{2y_{i-1}}{x_i-x_{i-1}}\right)(x-x_{i-1})\right)\left(\frac{x-x_i}{x_i-x_{i-1}}\right)^2$$
$$+\left(y_i+\left(m_i-\frac{2y_i}{x_i-x_{i-1}}\right)(x-x_i)\right)\left(\frac{x-x_{i-1}}{x_i-x_{i-1}}\right)^2$$

che posto $y_i=f(x_i)$ è esattamente quanto dato prima:

$$p_i(x)=\left[f(x_{i-1})+\left(m_{i-1}+\frac{2f(x_{i-1})}{h}\right)(x-x_{i-1})\right]\left(\frac{x-x_i}{h}\right)^2$$
$$+\left[f(x_i)+\left(m_i-\frac{2f(x_i)}{h}\right)(x-x_i)\right]\left(\frac{x-x_{i-1}}{h}\right)^2$$

2. A questo punto resta da ricavare il sistema lineare, che è ciò che ci darà i valori degli m_i. Per fare ciò calcoliamo la derivata seconda del polinomio interpolante, partendo dal calcolare le derivate secondo delle singole basi di Hermite:

$$h''_{0,i-1}(x)=\frac{2}{h^3}(h+6x-2x_{i-1}-4x_i)$$
$$h''_{0,i}(x)=\frac{2}{h^3}(h-6x+2x_i+4x_{i-1})$$
$$h''_{1,i-1}(x)=\frac{1}{h^2}(6x-2x_{i-1}-4x_i)$$
$$h''_{1,i}(x)=\frac{1}{h^2}(6x-2x_i-4x_{i-1})$$

da cui:

$$p''_i(x)=\frac{2}{h^3}(h+6x-2x_{i-1}-4x_i)\,y_{i-1}+\frac{2}{h^3}(h-6x+2x_i+4x_{i-1})\,y_i$$
$$+\frac{1}{h^2}(6x-2x_{i-1}-4x_i)\,m_{i-1}+\frac{1}{h^2}(6x-2x_i-4x_{i-1})\,m_i$$

Ottenuta la derivata seconda del polinomio interpolante, calcoliamola nei punti x_{i-1} e x_i. Per x_{i-1}, ad esempio, si avrà:

$$p''_i(x_{i-1})=\frac{2}{h^3}(h+6x_{i-1}-2x_{i-1}-4x_i)\,y_{i-1}+\frac{2}{h^3}(h-6x_{i-1}+2x_i+4x_{i-1})\,y_i$$
$$+\frac{1}{h^2}(6x_{i-1}-2x_{i-1}-4x_i)\,m_{i-1}+\frac{1}{h^2}(6x_{i-1}-2x_i-4x_{i-1})\,m_i$$
$$=\frac{2}{h^3}(h+4x_{i-1}-4x_i)\,y_{i-1}+\frac{2}{h^3}(h-2x_{i-1}+2x_i)\,y_i$$
$$+\frac{1}{h^2}(4x_{i-1}-4x_i)\,m_{i-1}+\frac{1}{h^2}(2x_{i-1}-2x_i)\,m_i$$
$$=\frac{2}{h^3}(h-4h)y_{i-1}+\frac{2}{h^3}(h+2h)y_i+\frac{1}{h^2}(-4h)m_{i-1}+\frac{1}{h^2}(-2h)m_i$$
$$=-\frac{6}{h^2}y_{i-1}+\frac{6}{h^2}y_i-\frac{4}{h}m_{i-1}-\frac{2}{h}m_i=\frac{6\,(y_i-y_{i-1})}{h^2}-\frac{4m_{i-1}+2m_i}{h}$$

e con passaggi simili si calcola in x_i, per cui:

$$p_i''(x_{i-1}) = \frac{6\,(y_i - y_{i-1})}{h^2} - \frac{4m_{i-1} + 2m_i}{h}$$

$$p_i''(x_i) = \frac{2m_{i-1} + 4m_i}{h} - \frac{6\,(y_i - y_{i-1})}{h^2}$$

3. Avremo a questo punto da applicare le condizioni della spline cubica naturale, cioè derivata seconda nulla agli estremi:

$$\begin{cases} p_1''(x_0) = 0 \implies 2m_0 + m_1 = \frac{3}{h}\,(f(x_1) - f(x_0)) \\ p_k''(x_k) = 0 \implies 2m_k + m_{k-1} = \frac{3}{h}\,(f(x_k) - f(x_{k-1})) \end{cases}$$

e derivate seconde continue nei punti intermedi:

$$p_i''(x_i) = p_{i+1}''(x_i) \implies m_{i-1} + 4m_i + m_{i+1} = \frac{3}{h} f(x_{i+1}) - f(x_{i-1})$$

da cui il sistema lineare che corredava la forma del polinomio di Hermite, e quindi la tesi. □

- Se si vuole imporre il vincolo "a sinistra" della derivata prima e seconda, si può procedere in maniera analoga a quanto fatto nel caso quadratico, cioe prendere i tratti polinomiali:

$$c_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i, \quad i = 1, ..., k$$

e trovare gli a_i, b_i, c_i e d_i in funzione di x_{i-1}, x_i e le due derivate consecutive in x_{i-1} (che chiamiamo d_{i-1} e s_{i-1}, rispettivamente primo e secondo grado).

Visto che trovare direttamente i quattro problemi può essere difficile, decidiamo invece di trovare le 4 basi ortogonali b_1, b_2, b_3 e b_4 (di cui le prime 3 equivalenti a quelle di Hermite, e l'ultima corrispondente alla condizione derivata seconda in x_{i-1}), che rispettano quindi le condizioni:

$$b_1 : \begin{cases} b_1(x_{i-1}) = 1 \\ b_1(x_i) = 0 \\ b_1'(x_{i-1}) = 0 \\ b_1''(x_{i-1}) = 0 \end{cases}, \quad b_2 : \begin{cases} b_2(x_{i-1}) = 0 \\ b_2(x_i) = 1 \\ b_2'(x_{i-1}) = 0 \\ b_2''(x_{i-1}) = 0 \end{cases}$$

$$b_3 : \begin{cases} b_3(x_{i-1}) = 0 \\ b_3(x_i) = 0 \\ b_3'(x_{i-1}) = 1 \\ b_3''(x_{i-1}) = 0 \end{cases}, \quad b_4 : \begin{cases} b_3(x_{i-1}) = 0 \\ b_3(x_i) = 0 \\ b_3'(x_{i-1}) = 0 \\ b_3''(x_{i-1}) = 1 \end{cases}$$

Queste si potranno ricavare, direttamente dall'imposizione delle condizioni, come:

1. $b_1\,(x, x_{i-1}, x_i) = a_{b1}\,(x_{i-1}, x_i)\,x^3 + b_{b1}\,(x_{i-1}, x_i)\,x^2 + c_{b1}\,(x_{i-1}, x_i)\,x + d_{b1}\,(x_{i-1}, x_i)$, con:
   - $a_{b1}\,(x_{i-1}, x_i) = \frac{1}{(x_{i-1} - x_i)^3}$;

- $b_{b1}(x_{i-1}, x_i) = -3a_{b1}(x_{i-1}, x_i)\, x_{i-1}$;
- $c_{b1}(x_{i-1}, x_i) = 3a_{b1}(x_{i-1}, x_i)\, x_{i-1}^2$;
- $d_{b1}(x_{i-1}, x_i) = 1 - a_{b1}(x_{i-1}, x_i)\, x_{i-1}^3$.

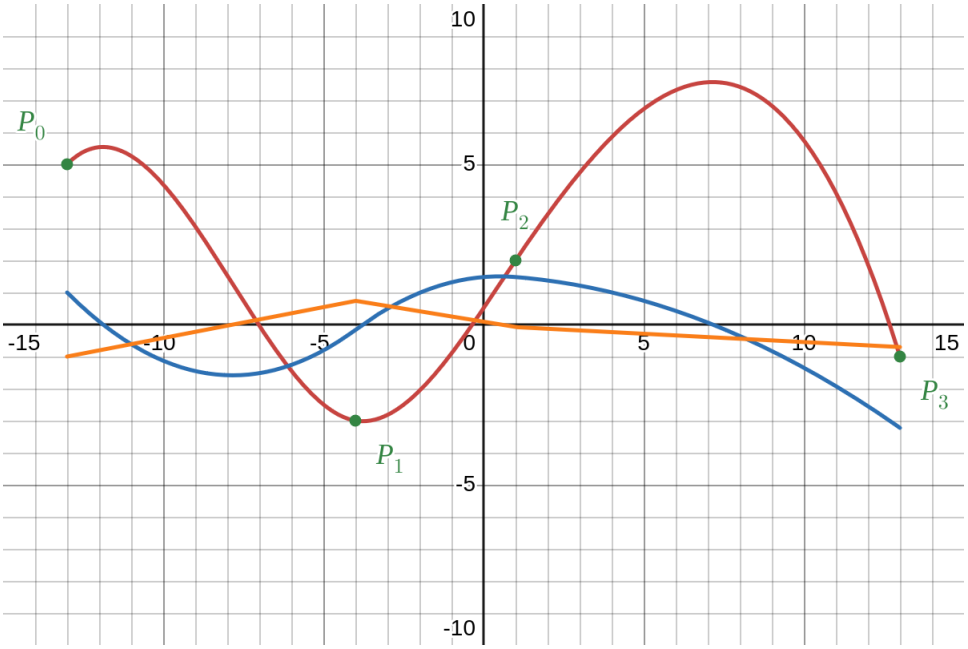
2. Questa è banalmente $b_2(x, x_{i-1}, x_i) = 1 - b_1(x, x_{i-1}, x_i)$;
3. $b(x, x_{i-1}, x_i) = a_{b3}(x_{i-1}, x_i)\, x^3 + b_{b3}(x_{i-1}, x_i)\, x^2 + c_{b3}(x_{i-1}, x_i)\, x + d_{b3}(x_{i-1}, x_i)$, con:
   - $a_{b3}(x_{i-1}, x_i) = \frac{x_{i-1} - x_i}{(x_i - x_{i-1})^3}$;
   - $b_{b3}(x_{i-1}, x_i) = -3a_{b3}(x_{i-1}, x_i)\, x_{i-1}$;
   - $c_{b3}(x_{i-1}, x_i) = 1 + 3a_{b3}(x_{i-1}, x_i)\, x_{i-1}^2$;
   - $d_{b3}(x_{i-1}, x_i) = -a_{b3}(x_{i-1}, x_i)\, x_{i-1}^3 - x_{i-1}$.
4. $b(x, x_{i-1}, x_i) = a_{b4}(x_{i-1}, x_i)\, x^3 + b_{b4}(x_{i-1}, x_i)\, x^2 + c_{b4}(x_{i-1}, x_i)\, x + d_{b4}(x_{i-1}, x_i)$, con:
   - $a_{b4}(x_{i-1}, x_i) = \frac{(x_i - x_{i-1})^2}{2(x_{i-1} - x_i)^3}$;
   - $b_{b4}(x_{i-1}, x_i) = \frac{1}{2} - 3a_{b4}(x_{i-1}, x_i)\, x_{i-1}$;
   - $c_{b4}(x_{i-1}, x_i) = 3a_{b4}(x_{i-1}, x_i)\, x_{i-1}^2 - x_{i-1}$;
   - $d_{b4}(x_{i-1}, x_i) = \frac{1}{2}x_{i-1}^2 - a_{b4}(x_{i-1}, x_i)\, x_{i-1}^3$.

A questo punto basterà definire ogni tratto polinomiale c_i come:

$$c_i(x) = b(x, x_{i-1}, x_i)\, y_{i-1} + b(x, x_{i-1}, x_i)\, y_i + b(x, x_{i-1}, x_i)\, d_{i-1} + b(x, x_{i-1}, x_i)\, s_{i-1}$$

dove gli d_{i-1} e s_{i-1} sono rispettivamente la derivata prima e seconda in ogni punto, data per il punto x_0 e ricavata dal tratto precedente per ogni tratto successivo.

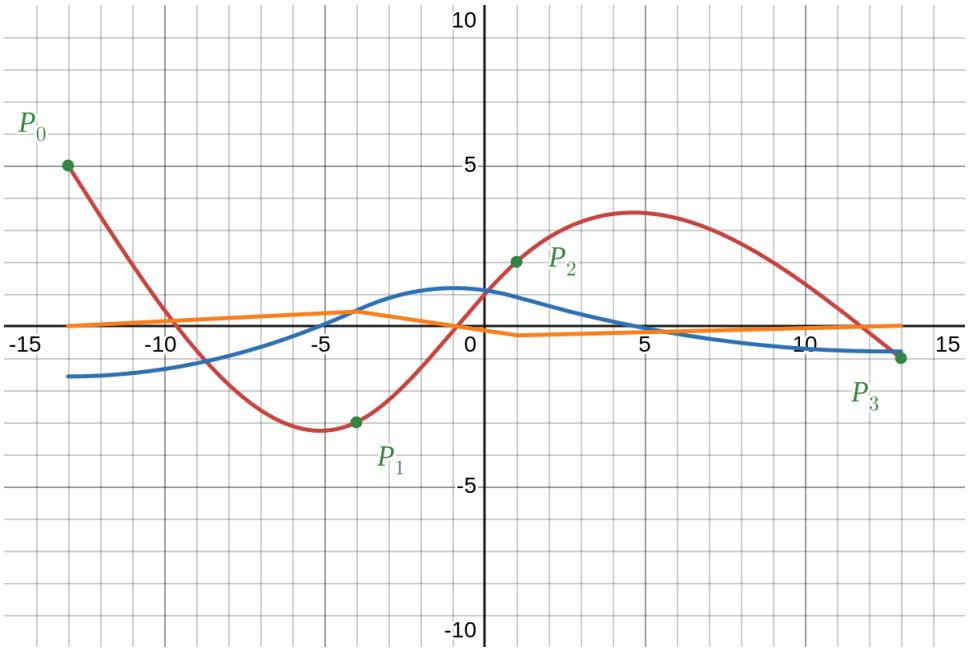
Un interpolazione di questo tipo per l'esempio considerato nelle scorse sezioni ha l'aspetto, prese $d_0 = 1$ e $s_0 = -1$ e tracciate derivata prima e seconda:



da dove notiamo che la derivata fino al secondo grado resta continua, da cui la condizione 3 è rispettata.

Vediamo che in verità si può risolvere in qualche modo anche lo scorso tipo di problema con questo metodo, semplicemente imponendo $s_0 = 0$ e cercando d_0 perché risulti $s_k = 0$.

Nell'esempio, questo si ottiene per $d_0 \approx -1.574$, con relativo grafico:



Esempi di questi tipi di interpolazione si possono trovare sempre su Desmos, al link `https://www.desmos.com/calculator/gm1idqauiy` (da qui sono state generate le figure).

1.2 Approssimazione ai minimi quadrati

Un altro approccio per l'approssimazione di insiemi di punti, specialmente nel caso questi siano imponenti in dimensioni, può essere dato dall'**approssimazione ai minimi quadrati**.

Un caso banale di questo tipo di approssimazione può essere quello della *regressione lineare*: dato un insieme di $k+1$ punti (x_j, y_j), si cerca la retta che minimizza la distanza quadratica da ogni punto. In qusto caso la retta prende il nome di **funzione modello**.

Generalizziamo quindi questo approssimazione in 2 direzioni:

- Ammettiamo che si abbiano più punti che funzioni modello (che possiamo intendere come le basi polinomiali usate finora);
- Amettiamo l'utilizzo di funzioni modello non polinomiali.

Formalmente, quindi, dati $k+1$ punti $(x_0, y_0), ..., (x_k, y_k)$ punti con $y_j = f(x_j)$ e $m+1$ funzioni modello $G_0(x), ..., G_m(x)$ con $m \leq x$ cercheremo l'approssimante $\Phi(x) \approx f(x)$ della forma:

$$\Phi(x) = \sum_{i=0}^{m} G_i(x) \cdot c_i, \quad c_i \in \mathbb{R}$$

con c_i da trovare, cioè la combinazione lineare ottima (definiremo cosa significa ottima fra poco) delle funzioni modello.

- Ad esempio, la regressione lineare quella data dalle funzioni modello:

$$G_0(x) = 1, \quad G_1(x) = x$$

approssimante $k + 1 = \#$ punti nel grafico;

- Potremmo avere altri tipi di funzioni modello. Ad esempio potremmo avere:

$$G_0(x) = e^{2x}, \quad G_1(x) = \sin\left(\frac{x}{2}\right), \quad G_2(x) = \frac{1}{3x^2 + 4}$$

In questo caso l'approssimante avrebbe una forma del tipo:

$$\Phi(x) = c_0 e^{2x} + c_1 \sin\left(\frac{x}{2}\right) + \frac{c_2}{3x^2 + 4}$$

A questo punto il problema è capire cosa significa dire che $\Phi(x)$ passa *"vicino"* ai punti (x_j, y_j), cioè qual'è la funzione da ottimizzare per trovare i c_i ottimi in modo che risulti $\Phi(x) \approx f(x)$. Dobbiamo quindi definire una qualche funzione di errore $\psi(c_0, ..., c_m) : \mathbb{R}^{m+1} \to \mathbb{R}^+$.

Vediamo come definire tale funzione.

- Un primo approccio potrebbe essere il semplice *scarto*:

$$\psi(c_0, ..., c_m) = \sum_{i=0}^{k} (\Phi(x_i) - y_i)$$

Questa ha il problema di poter dare cancellazione, cioè errori di punti diversi potrebbero annullarsi, per dare valori di ψ bassi quando in realtà la funzione è molto lontana da $f(x)$;

- Un approccio migliore è quindi quello di prendere lo *scarto assoluto*:

$$\psi(c_0, ..., c_m) = \sum_{i=0}^{k} \left|\Phi(x_i) - y_i\right|$$

In questo caso non soffriremo di cancellazione, ma avremo il problema che ψ non è differenziabile (si avranno punti angolosi dati dal valore assoluto);

- L'approccio migliore risulta quindi:

$$\psi(c_0, ..., c_m) = \sum_{i=0}^{k} (\Phi(x_i) - y_i)^2$$

cioè lo *scarto quadratico*, che risolve sia i problemi della cancellazione che della continuità C^1.

Il metodo dei minimi quadrati consisterà quindi nello scegliere:

$$c = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_m \end{pmatrix} \in \mathbb{R}^{m+1}$$

tale che:

$$c = \min_{c \in \mathbb{R}^{m+1}} \psi(c)$$

cioè c è punto di minimo di $\psi(c)$.

Dovremo quindi minimizzare:

$$\psi(c) = \sum_{i=0}^{k} (\Phi(x_i) - y_i)^2 = \sum_{i=0}^{k} \left(\sum_{j=0}^{m} c_j \cdot G_j(x_i) - y_i \right)^2$$

cioè in forma matriciale, posti:

$$A = \begin{pmatrix} G_0(x_0) & G_1(x_0) & ... & G_m(x_0) \\ \vdots & & & \vdots \\ G_0(x_0) & G_1(x_1) & ... & G_m(x_k) \end{pmatrix}, \quad c = \begin{pmatrix} c_0 \\ \vdots \\ c_m \end{pmatrix}, \quad b = \begin{pmatrix} y_0 \\ \vdots \\ y_k \end{pmatrix}$$

con $A \in \mathbb{R}^{(k+1)\times(m+1)}$, $c \in \mathbb{R}^{m+1}$, $b \in \mathbb{R}^{k+1}$, equivale a:

$$\psi(c) = \sum_{i=0}^{k} ((Ac)_i - b_i)^2 = \left| Ac - b \right|_2^2$$

Stiamo quindi minimizzando la norma quadratica del residuo di $Ac = b$, che dalle dimensioni della matrice A e i vettori c e b è un sistema sovradeterminato (più equazioni che incognite):

$$c^* = \min_{c \in \mathbb{R}^{m+1}} |Ac - b|_2$$

Possiamo quindi risolvere interpretando il sistema come uno di quelli visti in 11.1, quindi sfruttando le **equazioni normali** o la **decomposizione QR**.

1.2.1 Implementazione MATLAB di un approssimatore ai quadrati minimi

Vediamo quindi come implementare questo approccio in MATLAB.

Abbiamo già a disposizione una funzione, `least_squares()`, definita in 12.2.9 per la risoluzione di sistemi sovradeterminati usando la decomposizione QR. Ci mancherà quindi da scrivere una funzione che calcola il valore della matrice A, valutando le funzioni in base G_i, e che campiona una certa funzione $f(x)$ in determinati punti x_i per generare un vettore di y_i, magari usando la `linspace()`, così da avere un insieme di nodi equispaziati del tipo:

$$x_i = l + ih, \quad i = 0, ..., k-1$$
$$h = \frac{u - l}{k - 1}$$

Per il problema di come rappresentare le G_i usiamo dei *function handle*, dichiarati come segue:

```
% un function handle alla funzione f(x) = x^2
f = @(x) x.^2
% si usa come: f(2) -> 4
```

Una base potrà quindi essere costruita racchiudendo più handle di questo tipo in un *array di celle*.

Prendiamo per adesso la base:

$$G_0(x) = e^x, \quad G_1(x) = \frac{1}{x}, \quad G_2(x) = \sin(x)$$

Avremo allora una funzione del tipo:

```
function c = approx_trasc(f, l, u, n)
    % le funzioni in base
    funcs = {
      @(x) exp(x);
      @(x) 1/x;
      @(x) sin(x)
    };

    % costruisce la matrice G
    function G = build_G(X)
        m = height(funcs);

        G = zeros(n, m);

        for i = 1:n
            for j = 1:m
                G(i, j) = funcs{j}(X(:, i));
            end
        end
    end

    % campiona f
    X = linspace(l, u, n);
    Y = f(X);

    % costruisci G
    G = build_G(X);

    % usa il risolutore ai quadrati minimi
    c = least_squares(G, Y');
end
```

Attraverso questa funzione potremo calcolare i coefficienti c che minimizzano lo scarto quadratico sugli `n` punti fra `l` e `u`.

Prendiamo l'esempio della funzione:

$$f(x) = (2x - 3)(x - 3)(x - 4)(x - 5)$$

Definiremo questa, in MATLAB, come segue:

```
f = @(x) (2.*x - 3).*(x - 3).*(x - 4).*(x - 5);
```

Potremo allora trovare i coefficienti c di un approssimazione in forma:

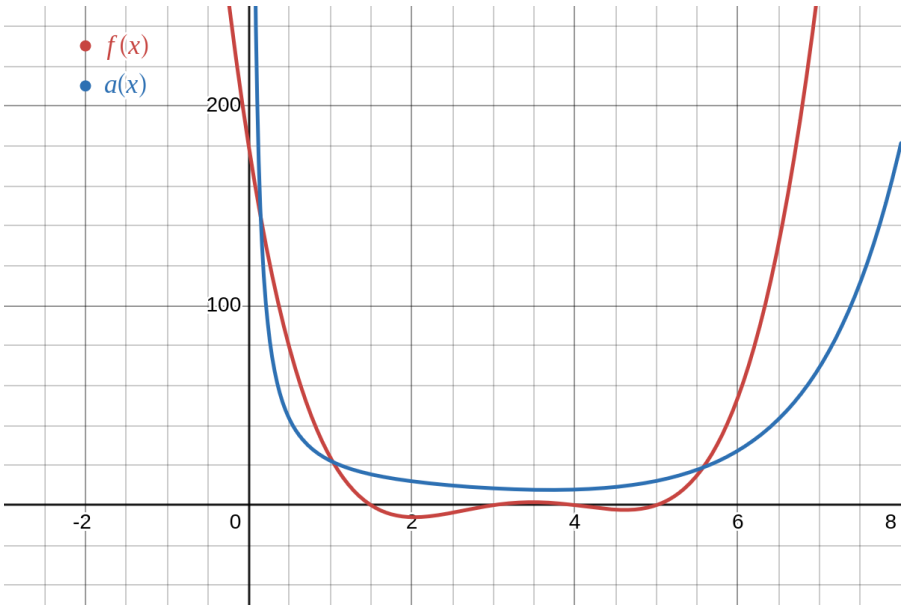
$$a(x) = c_0 \cdot e^x + c_1 \cdot \frac{1}{x} + c_2 \cdot \sin(x)$$

usando la funzione MATLAB appena implementata:

```
>> approx_trasc(f, 0.1, 6, 100)

ans =
    0.0596
   21.2591
    1.8092
```

cioè $c_0 = 0.0596$, $c_1 = 21.2591$, $c_2 = 1.8092$, che sul grafico dà:



Per un errore medio di:

$$\varepsilon = \sqrt{\sum_{i=i}^{100} (f(x_i) - a(x_i))^2} \approx 163.141$$

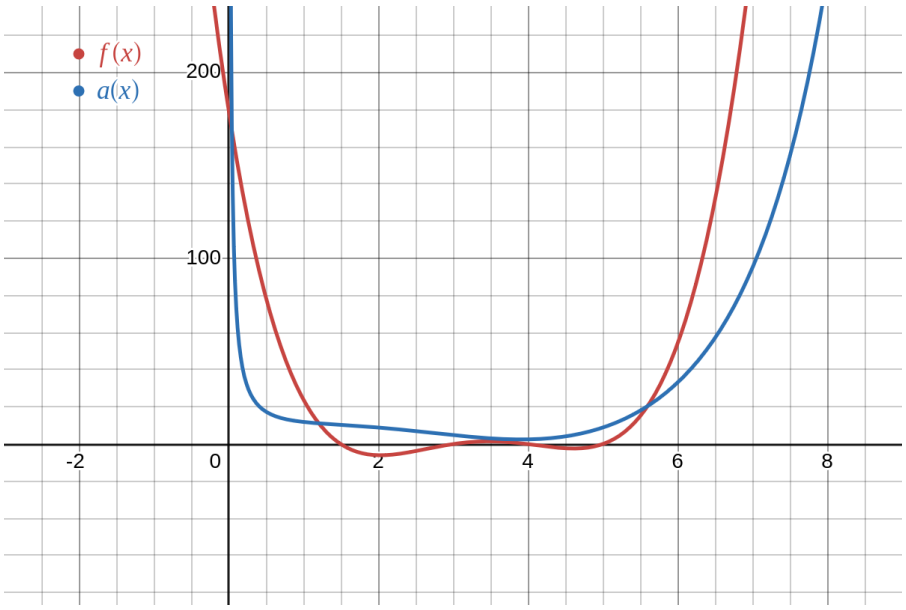
Notiamo quindi che la funzione non passa necessariamente per nessuno dei punti campianati, ma riduce l'errore medio lungo tutta la regione campionata.

Potremmo renderci conto che la maggior parte dell'errore è data dal fatto che cerchiamo di approssimare la funzione nella regione $0.1 \sim 0.7$, che ci porta ad avere un errore più grande nella zona vicino all'asse x. Se fossimo interessati ad ottenere un'approssimazione più accurata in questa regione, potremmo richiamare la funzione aumentando il bound l:

```
>> approx_trasc(f, 0.7, 6, 100)

ans =
    0.0825
    7.4361
    5.0514
```

cioè $c_0 = 0.0825$, $c_1 = 7.4361$, $c_2 = 5.0514$, che sul grafico dà:



Per un errore medio di:

$$\varepsilon = \sqrt{\sum_{i=i}^{100} (f(x_i) - a(x_i))^2} \approx 106.392$$

sul nuovo intervallo, che è già più accurato.