Recap:

$(x_i, y_i) \quad i = 0, \dots, n \qquad x_i \neq x_j \text{ se } i \neq j$

$p(x_i) = y_i$

$$p(x) = \sum_{i=0}^{n} y_i \, \mathcal{L}_i(x) \qquad \mathcal{L}_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}$$

Adesso ci chiediamo un altra cosa: abbiamo visto che se cerchiamo di risolvere il problema, risolvendo il sistema lineare con la matrice di Vandermonde le cose vanno male perché la matrice di Vandermonde è mal condizionata.
Invece qua, cosa succede se cambio leggermente i dati?
I dati di questo problema sono le y_i , supponiamo che siano isolazioni, che siano valori che ho calcolato con una funzione che io non conosco… qualcosa calcolato con il calcolatore, dove quindi ho commesso un errore.

ABOUT IL CONDIZIONAMENTO

Supponiamo di introdurre degli errori sui dati

$(x_i, \tilde{y}_i) \quad i = 0, \dots, n \qquad \varepsilon_i = y_i - \tilde{y}_i$

Il mio polinomio che era: $p(x) = \sum_{i=0}^{n} y_i \, \mathcal{L}_i(x)$

Ora diventa: $\tilde{p}(x) = \sum_{i=0}^{n} \tilde{y}_i \, \mathcal{L}_i(x)$

Errore tra polinomi interpolatori

$$|p(x) - \tilde{p}(x)| = \left| \sum_{i=0}^{n} y_i \mathcal{L}_i(x) - \sum_{i=0}^{n} \tilde{y}_i \mathcal{L}_i(x) \right| = \left| \sum_{i=0}^{n} (y_i - \tilde{y}_i) \mathcal{L}_i(x) \right| \leq \sum_{i=0}^{n} |y_i - \tilde{y}_i| \, |\mathcal{L}_i(x)|$$

disuguaglianza di Cauchy-Schwarz

$$= \sum_{i=0}^{n} |\varepsilon_i| \, |\mathcal{L}_i(x)| \leq \|\varepsilon\|_\infty \sum_{i=0}^{n} |\mathcal{L}_i(x)|$$

Questo è quello che mi può far variare l'errore

faccio un'altra maggiorazione: anziché prendere ogni ε_i, ci prendo la norma infinito

Per determinare il numero di condizionamento, noi abbiamo sempre messo in rapporto l'errore relativo di entrambe (?).

Oss:

$$\|p(x)\|_\infty = \max_{x \in I_x} |p(x)| \geq \max_{i=0,\dots,n} |p(x_i)|$$

$$= \max_{i=0,\dots,n} |y_i| = \|y\|_\infty$$

Errore relativo sui risultati

Errore relativo sui dati

$$\frac{\|p(x) - \tilde{p}(x)\|_\infty}{\|p(x)\|_\infty} \leq \frac{\|\varepsilon\|_\infty \, \left\| \sum_{i=0}^{n} \mathcal{L}_i(x) \right\|_\infty}{\|y\|_\infty} = \frac{\|y - \tilde{y}\|_\infty}{\|y\|_\infty} \left\| \sum_{i=0}^{n} \mathcal{L}_i(x) \right\|_\infty$$

COSTANTE di Lebesgue: Λ_n

= numero di condizionamento del nostro problema di interpolazione

Il numero di condizionamento del problema di interpolazione attraverso i polinomi fondamentali di Lagrange dipende da come sono distribuiti i nodi nell'intervallo di interpolazione.

Oss: a piccole perturbazioni sui dati corrisponderanno piccole variazioni sul polinomio interpolatore se la costante di Lebesgue è piccola [rispetto all'ordine di grandezza dei dati]

Oss: La costante di Lebesgue dipende dalla distribuzione dei nodi

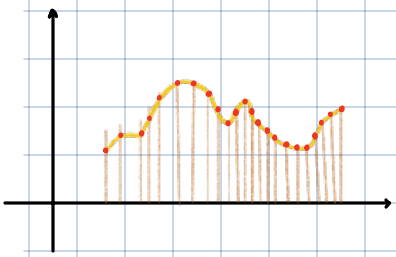
Se ho nodi equispaziati, la costante di Lebesgue, si comporta

$$\Lambda_n \simeq \frac{2^{n+1}}{e\, n \log n} \xrightarrow[n \to +\infty]{} \infty$$

Questo ci da già un'idea che se cerco di approssimare i miei dati numerosi con un polinomio interpolatore, il rischio è che il mio errore possa espandere, aumentando il numero di dati.
Questo lo vedremo proprio con la funzione di Runge.
Oss: i nodi di Chebyshev, che sono diversamente distribuiti nell'intervallo [-1, 1], hanno una costante di Lebesgue più piccola e quindi si comportano meglio nella funzione di Runge.

Abbiamo visto che l'idea di tutto questo è osservare che, se io utilizzo per approssimare questi dati e funzioni, un unico polinomio interpolatore e se ho tanti dati, le cose potrebbero andarmi anche male o perchè i nodi sono equispaziati o perchè introduco una piccola variazione sul dato e quindi la costante di Lebesgue mi fa esplodere i risultati, anche se i nodi non sono equispaziati, il teorema visto a lezione scorsa non ci garantisce una convergenza, però lo possiamo usare per adottare una strategia un po' più efficace:
Se iniziamo ad avere tanti dati, ci conviene ancora trovare un unico polinomio che interpola questi dati?
Una strategia potrebbe essere: anziché trovare un unico polinomio interpolatore che passi per questi punti, potrebbe essere considerare un polinomio interpolatore a tratti con grado fissato, cioè dire per ogni copia di nodi, vado a considerare un polinomio lineare che collega questa coppia di punti.

INTERPOLAZIONE COMPOSITA DI LAGRANGE



Ho un intervallo [a,b] che suddivido in m sottointervalli $I_k = [a_k, a_{k+1}] \quad k = 0, \dots, m-1$
In ogni sottoinervallo I_k faccio interpolazione semplice con polinomi di Lagrange di grado n.
Questi nodi li chiamo $\{x_i^{(k)}\} \quad 0 \le i \le n$ e saranno n+1 su ogni intervallo I_k
Riprendiamo il teorema della volta scorsa

$$|E_n(x)| = \frac{|f^{(n+1)}(C_x)|}{(n+1)!} \cdot |w_{n+1}(x)| \le \frac{M}{(n+1)!} |w_{n+1}(x)| \le \frac{M}{(n+1)!} |I_x|^{n+1}$$

E vediamo che il mio polinomio approssimante dipende da queste quantità.
Vediamo qual è il vantaggio di aver suddiviso prima in tanti sottointervalli.
Su M e (n+1)! non ci possiamo fare molto, però quello su cui abbiamo migliorato è l'ampiezza dell'intervallo I_x, perchè su ogni intervallo abbiamo ridotto l'ampiezza.

$$\|f - P_n^c\|_\infty \le c\, H^{n+1} \|f^{(n+1)}\|_\infty$$

Polinomio composito di grado n: prima ho diviso i m sottointervalli e poi ho applicato interpolazione semplice di grado n

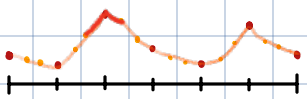
ampiezza dei sottointervallo in cui ho diviso ab → $\frac{|b-a|}{m}$ (se lo divido in maniera uniforme), altrimenti $H = \max_{k=0,\dots,m-1} |I_k|$

Come posso cercare di far convergere e ridurre questo errore? Ora ho qualcosa che posso far tendere a 0, perchè se io faccio tendere m all'infinito, H tende a 0, quindi questo errore riesco in qualche modo a rimpicciolirlo.

Avere una migliore approssimazione è suddividere il mio intervallo di interpolazione in tanti sottointervalli, in ognuno di essi applicare l'interpolazione di grado semplice o basso (1 o 2 o 3). Questa è la strategia che di solito si adotta per avere un migliore approssimazione dei dati di una funzione e si chiama interpolazione composita di Lagrange.

Esempio:

SPLINE CUBICHE INTERPOLANTI



Che cosa sono?

Divido in m sottointervalli

Quindi cerco di creare il polinomio composito, dato che la spline è cubica vuol dire che su ogni sottointervallo io voglio che la spline sia di grado 3, il mio polinomio interpolatore sia di grado 3.

Poi siccome voglio che in ogni sototintervallo sia di gtrado 3, di quanti punti dovrò avere su ogni sottointervallo? 4, 2 li ho già che sono gli estremi, me ne servono altri 2 che sono interni.

Se mi accontentassi di accontentassi di avere un polinomio di grado 3 a tratti, interpolante nei punti • avrei un polinomio come disegnato. Cosa succede qua? Questo polinomio interpolante composito è continuo, perchè in ogni punto che sto considerando, il limite da destra e da sinistra in qualche modo è uguale, ma non è derivabile perchè le rette tangenti fanno un angolo.

Allora le spline cubiche chiedono qualcosa di più, le seguenti proprietà:

p_3^c

- p_3^c è un polinomio di grado 3 su ogni intervallo $I_k \quad k = 0, \dots, m-1$
- $p_3^c(x_i) = y_i \quad i = 0, \dots, m$ (voglio che sia interpolante i tutti i nodi che dividono i sottointervalli
- p_3^c abbia anche derivata prima e seconda continue nei nodi

$$p_3^c(x) = \begin{cases} a_3^{(0)}x^3 + a_2^{(0)}x^2 + a_1^{(0)}x + a_0^{(0)} & x \in I_0 \\ a_3^{(1)}x^3 + a_2^{(1)}x^2 + a_1^{(0)}x + a_0^{(0)} & x \in I_1 \\ \vdots \\ a_3^{(m-1)}x^3 + a_2^{(m-1)}x^2 + a_1^{(m-1)}x + a_0 & x \in I_{m-1} \end{cases}$$

Quante incognite ho? 4*m gradi di libertà

Devo imporre la seconda condizione $(p_3^c(x_i) = y_i \quad i = 0, \dots, m)$

$$\begin{cases} p_0(x_0) = y_0 \quad p_0(x_1) = y_1 \\ p_1(x_1) = y_1 \quad p_1(x_2) = y_2 \\ \vdots \\ p_{m-1}(x_{m-1}) = y_{m-1} \quad p_{m-1}(x_m) = y_m \end{cases}$$

Per imporre che sia interpolante ho imposto 2 condizioni ad ogni sottointervallo: 2*m condizioni di interpolazione. E queste sono quelle che ci assicurano già la continuità della funzione.

Ora ci manca la terza condizione: vogliamo che questo polinomio abbia derivata prima e seconda continue nei nodi:

$$\begin{cases} p_0'(x_1) = p_1'(x_1) \\ p_1'(x_2) = p_2'(x_2) \\ p_2'(x_3) = p_3'(x_3) \\ \vdots \\ p_{m-2}'(x_{m-1}) = p_{m-1}'(x_{m-1}) \end{cases}$$

Per avere derivata prima continua nei nodi interni

$$\begin{cases} p_0''(x_1) = p_1''(x_1) \\ p_1''(x_2) = p_2''(x_2) \\ p_2''(x_3) = p_3''(x_3) \\ \vdots \\ p_{m-2}''(x_{m-1}) = p_{m-1}''(x_{m-1}) \end{cases}$$

Per avere derivata seconda continua nei nodi interni

E sono m-1 condizioni, e stessa situazione anche nella derivata seconda

Invece di imporre il passaggio per ulteriori due punti interni che mi devo andare a ricavare, sto chiedendo maggiori regolarità attorno ai nodi per salutare i gradi di libertà.
Alla fine però quante condizioni ho racimolato?

- 4m gradi di libertà [coefficienti dei polinomi cubici definiti in ogni sottointervallo]
- 2m di interpolazione
- m-1 per la derivata prima
- m-1 per la derivata seconda

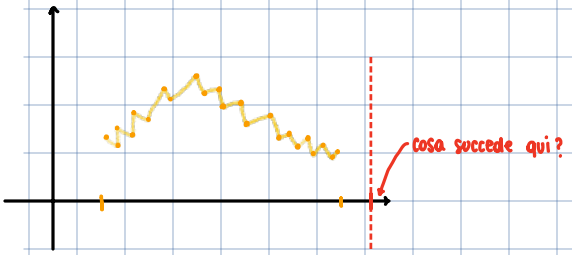
Per un totale di 8m-2 condizioni, quindi ho ancora altri due gradi da vincolare, altrimenti non mi passa una cubica da qua.
Quindi devo dare altre due condizioni e le "scelgo io":
Ad esempio di default in Matlab, il comando "*spline*", usa spline che hanno questa ulteriore condizione:

$\frac{d^3 P_3^c}{dx^3}(x_1)$ sia continua $\frac{d^3 P_3^c}{dx^3}(x_{m-1})$ sia continua SPLINE "NOT-A-KNOT"

Es: Interpolare con "interpolazione lagrangiana semplice", "composita lineare", "spline" i seguenti punti:

```
x = [-55 : 10 : 65]
y = [-3.25, -3.37, -3.35, -3.2, -3.12, -3.02, -3.02, -3.07, -3.17, -3.32, -3.3, -3.22, -3.1]
```

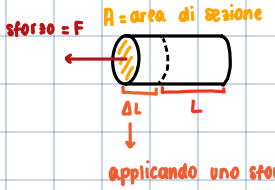
Ultima strategia riguardo approssimazioni di dati e funzioni:
Se ci sono davvero molti punti e magari non ci interessa davvero sapere cosa succede nell'intervallo che sta tra questi punti, ma ci interessa sapere cosa succederà dopo cioè in un punto esterno a questo set di dati.



La strategia potrebbe essere: uso un polinomio di grado basso, che decido io a priori, che non interpola più questi punti, non chiedo più che passi per questi punti, e che si discosti il meno possibile dai punti.
L'idea è: se decido che è una parabola, un polinomio di secondo grado, voglio trovare i coefficienti di questa parabola. Devo trovare il modo che mi trovi qualcosa che si discosti poco.

METODO DEI MINIMI QUADRATI

Partiamo con una retta: l'idea è trovare la retta che meglio approssima un insieme di dati.
Esempio:
Cerchiamo il legame tra lo sforzo σ e la deformazione ε di un disco intervertebrale



$\sigma = \frac{F}{A}$ $\varepsilon = \frac{\Delta L}{L}$

Cerco di trovare una legge lineare che collega questo sforzo a questa deformazione avendo a

disposizione delle misurazioni:

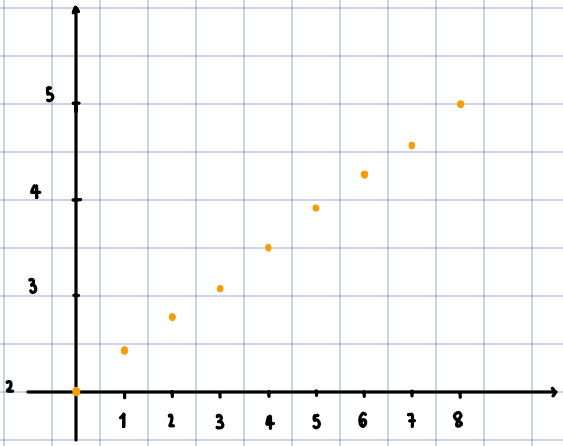
SFORZI: σ	0.00	0.06	0.14	0.25	0.31	0.47	0.60	0.70
DEFORMAZIONI: ε	0.00	0.08	0.14	0.20	0.23	0.25	0.28	0.29

Voglio stimare la deformazione $\sigma = 0.9 \quad \left(\frac{100\,N}{cm^2}\right)$

Prima cosa che osserviamo è che dove voglio stimare la deformazione è fuori dal mio intervallo di dati che conosco.

Esempio:

x_i =	0	1	2	3	4	5	6	7	8
y_i =	2.0	2.4	2.75	3.1	3.5	3.9	4.25	4.6	5



Vedo che i punti sono più o meno su una retta, come posso fare a trovare questa retta che può approssimare bene questo set di punti?
Avendo la retta due gradi di libertà, devo trovare due coefficienti: fissare m e q $\quad y = mx + q$
Decidiamo che imponiamo il passaggio per due punti 0 e 8:

$$\begin{cases} y_0 = m x_0 + q \longrightarrow q = 2 \\ y_8 = m x_8 + 2 \longrightarrow m = \frac{3}{8} \end{cases}$$

Come possiamo stimare se abbiamo fatto le cose fatte bene? Vediamo rispetto ai valore di y e x che abbiamo, se ci siamo discostati di molto.
Non sappiamo ancora se è interpolante int tutti i punti, ma se lo fosse avremmo che:

$$y_i = m x_i + q \qquad i = 0, \dots, 8$$
$$y_i - m x_i - q = 0$$

Se non fosse interpolante però, r_i è una buona stima dell'errore che ho commesso (=RESIDUO)

$$r_i = y_i - m x_i - q$$

Vediamo il residuo nei nostri punti:

x_i	0	1	2	3	4	5	6	7	8
y_i	2.0	2.4	2.75	3.1	3.5	3.9	4.25	4.6	5
r_i	0	0.025	0	-0.025	0	0.025	0	-0.025	0

In questo caso non ci è nemmeno andata male, siamo stati fortunati, se avessimo preso 0 e 5 sarebbe andata un po' peggio. Come faccio a scegliere la migliore retta che approssima i punti?
Siccome abbiamo analizzato i residui, proviamo a prendere la retta per cui la somma dei residui sia nulla $\quad \sum_{i=0}^{n} r_i = 0$

Va sempre bene questa scelta? Si può fare questa scelta?

Oss: non posso univocamente determinare una retta imponendo un solo vincolo.
Quello che cercheremo di minimizzare non è il residuo, ma il quadrato dei residui, così da svincolarci dal segno.

$\bar{p}(x)$ è il polinomio per cui

$$\sum_{i=0}^{n} \left[y_i - \bar{p}(x_i)\right]^2 \leq \sum_{i=0}^{n} \left[y_i - p(x_i)\right]^2 \qquad \forall p \in P_1$$

La retta dei minimi quadrati è quella retta che minimizza questa quantità di resto, qualsiasi altro polinomio di primo grado ha una sommatoria dei residui al quadrato maggiore.
P_1 è lo spazio dei polinomi di grado 1.
Il mio polinomio è del tipo

$$\bar{p}(x) = mx + q$$

$$\sum_{i=0}^{n} \left[y_i - (mx_i + q)\right]^2 = \sum_{i=0}^{n} y_i^2 + (mx_i + q)^2 - 2y_i(mx_i + q)$$

$$= \sum_{i=0}^{n} y_i^2 + m^2x_i^2 + q^2 + 2mx_iq - 2y_im_i - 2mx_iq$$

→ Funzione le cui incognite sono m, q

$= \Phi(m, q)$

Dato che q ed m sono le mie incognite, di grado è la funzione per la variabile q? E in m? È di grado 2.
Questa è un'espressione per un paraboloide con un unico punto di minimo, che è da trovare che ci assicura la disugualianza.
Il punto di minimo si ottiene azzerando lei derivate prime parziali (una rispetto ad m e una rispetto a q).
Tutto quello che non è m o q va trattato come fosse una costante, che se derivata = 0.

$$\begin{cases} \dfrac{\partial \Phi}{\partial m} = \displaystyle\sum_{i=0}^{n} \left[2mx_i^2 + 2qx_i - 2x_iy_i\right] = 0 \\ \dfrac{\partial \Phi}{\partial q} = \displaystyle\sum_{i=0}^{n} \left[2q + 2mx_i - 2y_i\right] = 0 \end{cases}$$

Impongo due vincoli e vado a trovare m e q.

$$\begin{cases} \displaystyle\sum_{i=0}^{n} 2mx_i^2 + 2qx_i - 2x_iy_i = 2m\sum x_i^2 + 2q\sum x_i - 2\sum x_iy_i = 0 \quad \bigstar \\ \displaystyle\sum_{i=0}^{n} 2q + 2mx_i - 2y_i = 2q(n+1) + 2m\sum x_i - 2\sum y_i = 0 \quad \bigstar \end{cases}$$

Da $\bigstar \rightarrow m = \dfrac{\sum y_i - (n+1)q}{\sum x_i}$ sostituisco in $\bigstar$

$$\begin{cases} \bar{q} = \dfrac{\sum y_i \sum x_i^2 - \sum x_i(\sum x_iy_i)}{(n+1)\sum x_i^2 - (\sum x_i)^2} \\ \bar{m} = \dfrac{(n+1)\sum x_iy_i - \sum x_i \sum y_i}{(n+1)\sum x_i^2 - (\sum x_i)^2} \end{cases}$$

inserisco in $\bigstar$

Oss: il procedimento si può applicare anche per trovare polinomi ai minimi quadrati di grado superiore ad 1. In Matlab con il comando “polyfit”

Oss: l’approssimazione ai minimi quadrati si può fare anche con fattori non polinomiali. In Matlab con il comando nlinfit.