

Contents

1 Numeri finiti

1.1 Numeri binari ed in base N

I numeri hanno diversi modi per essere rappresentati, tra cui quello più comune che è la **rappresentazione decimale**, dove ogni numero è formato da una lista di caratteri compresi tra $[0, 9]$ dove ad ogni cifra è associata una potenza del 10

Esempio: $147.3 = 1 \cdot 10^2 + 4 \cdot 10^1 + 7 \cdot 10^0 + 3 \cdot 10^{-1}$

In generale, comunque, per convertire un numero da una base n a base 10:

teorema

Dato un numero in base n $(a_k a_{k-1} \dots a_1 a_0)_n$, la sua conversione in base 10 è data dalla formula:

$$\sum_{i=0}^k a_i \cdot n^i$$

dove a_i sono le cifre in base n e n è la base.

Per forza di cose i calcolatori operano con i numeri *in base 2*, le cui cifre vengono chiamate **bit**, esempio 101101 (base 2). Per convertire un numero dalla base 10 alla base 2 bisogna dividerlo in potenze di due

$$37 \text{ (base 10)} = 32 + 4 + 1 = 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 100101 \text{ (base 2)}$$

In generale occorre dividerlo per due finché non si riduce ad uno ad esempio:

$$(35)_{10} = (100011)_2$$

$$35 : 2 = 17 \quad \text{resto } 1$$

$$17 : 2 = 8 \quad \text{resto } 1$$

$$8 : 2 = 4 \quad \text{resto } 0$$

$$4 : 2 = 2 \quad \text{resto } 0$$

$$2 : 2 = 1 \quad \text{resto } 0$$

$$1 : 2 = 0 \quad \text{resto } 1$$

2 hklklk

2.1 Che cos'è il floating point

Dato che in informatica non è possibile implementare il concetto di infinito (a differenza della matematica) non si è in grado di rappresentare tutto l'insieme dei reali; pertanto occorre, per forza di cose, approssimarlo. Il metodo che oggi è comune per il calcolo scientifico è noto come sistema **floating point** che permette una rappresentazione di ampio intervallo della retta reale con una distribuzione uniforme degli errori

2.2 Definizione formale

Si definisce **insieme dei numeri numeri macchina** (floating point) con t cifre significative, con base β , con $p \in [L, U]$ e con le cifre d_i dove $0 \leq d_i \leq \beta - 1$, $i = 1, 2, \dots$ e $d_1 \neq 0$

$$\mathbb{F}(\beta, t, L, U) = \{0\} \cup \{x \in \mathbb{R} = \text{sign}(x)\beta^p \sum_{i=1}^t d_i \beta^{-i}\}$$

Usualmente U è positivo e L negativo

Questo insieme è, quindi, un'approssimazione all'intervallo preciso di numeri $[min, max] \subseteq \mathbb{R}$ che dipendono da U e L . Se voglio rappresentare un numero maggiore/minore di questo intervallo avrò un errore chiamato **overflow/underflow**

2.3 Gap

con questa notazione non è impossibile scrivere ogni numero presente nella retta dei numeri Reali \mathbb{R} , ma tra due numeri vi è sempre una certa distanza quindi è pertanto vero che $\mathbb{F} \subseteq \mathbb{R}$. Questo concetto introduce la definizione di Gap:

Chiamiamo **gap** la distanza da un numero al suo successivo

È, inoltre, vero che il gap rimane costante quando l'esponente non cambia, mentre se nell'insieme \mathbb{F} il successivo di un numero ha un esponente diverso il suo gap sarà diverso di quello del suo precedente con l'esponente uguale. Facciamo un esempio per rendere il tutto più chiaro:

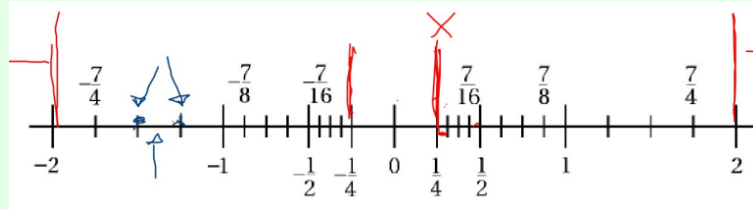
Sia $\beta = 2, t = 3, L = -1, U = 2$ allora

$$\mathbb{F}(2, 3, -1, 2) = \{0\} \cup \{0.100 \times 2^p, 0.101 \times 2^p, 0.110 \times 2^p, 0.111 \times 2^p, p = -1, 0, 1, 2\}$$

Questo insieme rappresenta 33 numeri compreso lo 0:

	$.101(-1)$	$.110(-1)$	$.111(-1)$
	$\frac{5}{16}$	$\frac{6}{16}$	$\frac{7}{16}$
$.100(-1)$	$.101(0)$	$.110(0)$	$.111(0)$
$\frac{1}{4}$	$\frac{5}{8}$	$\frac{6}{8}$	$\frac{7}{8}$
$.100(0)$	$.101(1)$	$.110(1)$	$.111(1)$
$\frac{1}{2}$	$\frac{5}{4}$	$\frac{6}{4}$	$\frac{7}{4}$
$.100(1)$	$.101(2)$	$.110(2)$	$.111(2)$
1	$\frac{5}{2}$	$\frac{6}{2}$	$\frac{7}{2}$
$.100(2)$			
2			

La rappresentazione di \mathbb{F} nella retta dei numeri reali è questa



Risulta quindi evidente l'aumento dell'ampiezza degli intervalli definiti dal valore p , in ognuno dai quali vengono localizzati le t cifre della mantissa; ne risulta di conseguenza una diradazione delle suddivisioni verso gli estremi sinistro e destro della retta reale.

Il numero t , quindi, fissa la precisione di rappresentazione di ogni numero; infatti la retta viene suddivisa in intervalli $[\beta^p, \beta^{p+1}]$ di ampiezza crescente e in questi intervalli viene rappresentato lo stesso numero $(\beta - 1)\beta^{t-1}$ di valori, generando, così, una suddivisione molto densa per valori vicini allo 0 e più rada per valori grandi in valore assoluto. Di conseguenza si ha che **maggiore è il numero t di cifre della mantissa, minore sarà l'ampiezza dei intervalli e quindi migliore l'approssimazione introdotta**

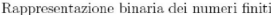
2.4 floating point nei calcolatori

La rappresentazione più comune del sistema nei calcolatori è definita dallo Standard IEEE 745 il cui insieme è $\mathbb{F}(2, 52, 1023, 1024)$ ed è descritta dalla formula

$$n = (-1)^s \cdot 2^{e-1023} \cdot (1 + f)$$

Dove:

- In memoria si traduce in un array contenente questi bit:



Quindi $n = (-1)^1 \times 2^3 \times (1 + 0,5) = -12,0$.

Inoltre sia $x \in \mathbb{R}$ dove $x = \pm\beta^p \cdot d_0d_1 \dots d_t, d_{t+1}, \dots$ con $L \leq p \leq U$, nel caso in cui $x \notin \mathbb{F}$ rappresento x con un elemento di \mathbb{F} che indico con $fl(x) = \pm\beta^p \cdot d_0d_1 \dots d_t$. Questa operazione si chiama **troncamento**

Il sistema floating point dato che è un'approssimazione è ovvio che la precisione di rappresentazione che offre non è perfetta, in quanto i numeri infiniti (come il π) vengono delineati con una serie finita di numeri finiti. La differenza tra la sua approssimazione e il suo valore reale è detta **errore di arrotondamento**

klkjkljkljklj

Sia $x \in \mathbb{R}$ un numero reale e sia $fl(x)$ (dove $fl()$ è la funzione arrotondamento) il numero x arrotondato, allora l'errore relativo di arrotondamento è definito dalla seguente formula:

$$\frac{|fl(x) - x|}{|x|} \leq \frac{1}{2}\beta^{1-t}$$

La quantità $esp = \frac{1}{2}\beta^{1-t}$ è detta **precisione macchina** nel sistema floating point, ed è il più piccolo numero macchina positivo tale che:

$$fl(1 + esp) > 1$$

2.7 aritmetica floating point

Il calcolatore esegue operazioni solo con numeri rappresentabili da calcolatore stesso, quindi in questo caso dai numeri scritti in forma floating point (per i numeri reali), il punto è che **le usuali operazioni aritmetiche non sono chiuse nell'insieme \mathbb{F}** , quindi presi 2 numeri dall'insieme \mathbb{F} e computati tramite operazioni aritmetiche, il risultato di tale computazione potrebbe appartenere nell'insieme \mathbb{R} . Pertanto **i risultati delle varie computazioni dovranno essere sempre arrotondati ad un numero che stia in \mathbb{F}**

2.7.1 operazione floating point

L'operazione floating point (o di macchina) è definita

$$\odot : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$$

Ogni operazione provoca un piccolo errore detto **errore di arrotondamento**:

$$\left| \frac{(x \odot y) - (x \cdot y)}{x \cdot y} \right| < eps$$

L'operazione floating point nei calcolatori viene strutturata in due fasi:

1. **Esegui l'operazione esatta:** (es. $z = x + y$) dove viene fatta utilizzando più bit utilizzati di quelli utilizzati per memorizzare il risultato, non è accessibile all'utente ma viene memorizzato in un registro interno alla cpu
2. **Trocamento del risultato:** (es. $x \oplus y = fl(z)$), dove una volta che il calcolo è stato fatto in precisione estesa il risultato viene arrotondato alla precisione del risultato

$$R(x) = 0.715 \cdot 20^{13} \quad R(y)$$

Esempio: $x = 2.15 \times 10^{12}, y = 1.25 \times 10^{-5}$

$$z = x - y$$

Viene calcolata come:

$$x = 2.15 \times 10^{12}$$

$$y = 0.000000000000000000125 \times 10^{12}$$

→ $x - y = 2.1499999999999999 \times 10^{12} \rightarrow 0$?

arrotondato a 2.15×10^{12} = \times

3 Robe

3.1 Ricerca della radice tramite bisezione :)

Prima di tutto si analizzi il calcolo della radice di una equazione (anche non lineare)

$$f(x) = 0$$

Esempio:

Ci si chiede se esistono tali implicazioni:

- Esistenza e unicità della soluzione
- Esistenza di algoritmi per calcolare soluzione

Innanzitutto si tenga conto di tale teorema

teorema

Teorema di Bolzano:

Sia $F : [a, b] \in \mathbb{R} \rightarrow \mathbb{R}$ continua in $[a, b]$ e $f(a) \cdot f(b) < 0 \Rightarrow \exists x^* : F(x^*) = 0$

Per la dimostrazione si vedano gli appunti/libri di analisi

3.1.1 Primo algoritmo di bisezione: n-iterativo

Intuitivamente l'algoritmo di bisezione ha due fasi:

1. se $f(a) \cdot f(c) = 0$ la radice è nell'intervallo $[a, c]$ che diventa la nuova versione dell'intervallo $[a, b]$
2. altrimenti, se $f(c) \cdot f(b) = 0$ la radice è nell'intervallo $[c, b]$ che diventa la nuova versione dell'intervallo $[a, b]$

Ecco qui descritto l'algoritmo di bisezione in pseudocodice:

Questo tipo di soluzione per la bisezione ha un numero n di iterazioni per approssimare la soluzione, questa soluzione è molto semplice ma non la migliore. Vi è tuttavia una soluzione migliore

3.1.2 Secondo algoritmo di bisezione: tolleranza d'errore

Nella pratica è molto più utile definire una costante ϵ che vuole rappresentare l'errore di approssimazione massimo, di modo che un'approssimazione \bar{r} della radice reale r garantisca che $|r - \bar{r}| \leq \epsilon$. In altre parole si vuole garantire che $r \in [\bar{r} - \epsilon, \bar{r} + \epsilon]$.

Nel nostro caso \bar{r} è il valore di $c = (b - a)/2$ mentre impostiamo $\epsilon = (b - a)/2$. Si imposti inoltre un **errore di tolleranza** e_{tol} per iterare le operazioni fino a che non si incontrino. Di seguito è riportato l'algoritmo:

Algorithm 1: primo algoritmo di Bisezione (Function f, int a, int b, int n)

Data: Funzione continua $f(x)$, continua in a e b con $f(a) \cdot f(b) < 0$, un numero N di iterazioni

Result: Approssimazione della radice

```

1 for 1 to n do
2    $c \leftarrow \frac{a+b}{2}$ ;
3   if  $f(c) = 0$  then
4     return  $c$  ; // Trovata la radice esatta
5   if  $f(a) \cdot f(c) < 0$  then
6      $b \leftarrow c$  ; // La radice è nell'intervallo  $[a, c]$ 
7   else
8      $a \leftarrow c$  ; // La radice è nell'intervallo  $[c, b]$ 
9 return  $c$  ; // Approssimazione della radice

```

Algorithm 2: Algoritmo di Bisezione

Data: Funzione continua $f(x)$, estremi a e b con $f(a) \cdot f(b) < 0$, tolleranza ϵ

Result: Approssimazione della radice x^* con $|f(x^*)| < \text{Toccalarivoluzioneeperlesighedelchea2euro}\epsilon$

```

1 while  $|b - a| > \epsilon$  do
2    $c \leftarrow \frac{a+b}{2}$ ; if  $f(c) = 0$  then
3     return  $c$  ; // Trovata la radice esatta
4   if  $f(a) \cdot f(c) < 0$  then
5      $b \leftarrow c$  ; // La radice è nell'intervallo  $[a, c]$ 
6   else
7      $a \leftarrow c$  ; // La radice è nell'intervallo  $[c, b]$ 
8 return  $\frac{a+b}{2}$  ;
; // Approssimazione della radice

```

3.2 Autovalore e autovettori

3.2.1 Definizione

Data una matrice A di dimensione $n \times n$ se vale:

$$Ax = \lambda x$$

con $\lambda \in \mathbb{R}$ e $x \in \mathbb{R}^n$, λ si dice autovalore di A e x autovettore di A associato a λ

Data questa definizione, e richiamando i vari concetti dell'algebra lineare, abbiamo che se vale $Av_i = \lambda_i v_i, \forall i \in 1, \dots, n$ allora vale anche:

$$[Av_1, Av_2, \dots, Av_n] = [\lambda_1 v_1, \lambda_2 v_2, \dots, \lambda_n v_n]$$

$$\text{quindi: } A[v_1, v_2, \dots, v_n] = [v_1, v_2, \dots, v_n] \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{pmatrix}$$

Questa bellissima scrittura si può tranquillamente riassumere come:

$$AV = DV$$

3.2.2 decomposizione agli autovalori

sia $A \in M^{n \times n}$, allora una decomposizione agli autovalori di A è una fattorizzazione (ovvero un prodtto) di questo tipo:

$$A = VDV^{-1}$$

Dove $V = [v_1, v_2, \dots, v_n]$ è la matrice con colonne gli autovettori v_i di A e D è una matrice diagonale che ha sulla diagonale gli autovalori di A associati alle colonne di V

Se A ha una decomposizione agli autovalori allora si dice che A è **diagonalizzabile**; in caso contrario A si dice **non diagonalizzabile** (o defettiva).

Se A ha n autivalori distinti, allora A è diagonalizzabile

3.2.3 Vettori ortogonali e ortonormali

Siano v_1, v_2, \dots, v_m vettori di R^n , si dicono **ortognonali** se:

$$v_i^T v_j = 0, \quad \text{se } i \neq j$$

Si può scrivere anche:

$$\langle v_i, v_j \rangle = 0, \quad \text{se } i \neq j$$

Ovvero il prodotto scalare

Questa relazione di vettori si collega molto bene con i vettori ortonormali:

Siano v_1, v_2, \dots, v_m vettori di R^n , si dicono **ortonormali** se sono ortogonali w di lunghezza unitaria, ovvero $v_i^T v_j = \|v_i\| = \delta_{ij}$ dove

$$\delta_{ij} = \begin{cases} 1 & \text{se } i = j \\ 0 & \text{se } i \neq j \end{cases}$$

3.2.4 Matrici ortogonali

Una matrice W di dimensione $n \times n$ si dice ortogonale se le sue colonne sono vettori ortonormali

Un esempio semplice:

$$W = \begin{pmatrix} \frac{1}{\sqrt{3}} & 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & 1 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{3}} & 0 & 0 \end{pmatrix}$$

le matrici ortogonali hanno diverse proprietà:

1. $W^T = W^{-1}$
2. $\forall x \in R^n, \|x\|_2 = \|Wx\|_2$

3.2.5 Fattorizzazione in Valori Singolar (SVD)

La fattorizzazione in valori singolari non è altro che uno strumento che permette di rappresentare una matrice $A \in M^{m \times n}$ attraverso una matrice diagonale.

Vediamo come:

teorema

Sia $A \in M^{m \times n}$ una matrice di rango k con $k \leq n \leq m$. Allora esistono:

- una matrice ortogonale $U \in M^{m \times m}$
- una matrice ortogonale $V \in M^{n \times n}$
- una matrice diagonale $\Sigma \in M^{m \times n}$

tali che:

$$A = U\Sigma V^T, \quad \Sigma = \begin{pmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$$

- Gli elementi $\sigma_1 \geq \sigma_2 \geq \dots \sigma_n \geq 0$ sono i cosiddetti **valori singolari di A**
- Mentre il rango k è uguale al numero di valori singolari positivi, cioè $k = \text{rango}(A) \iff (\sigma_1 \geq \sigma_2 \geq \dots \sigma_k > 0 \wedge \sigma_{k+1} = \dots = \sigma_n = 0)$
- Le colonne di $U = (u_1, u_2, \dots, u_m)$ e di $V = (v_1, v_2, \dots, v_n)$ sono, rispettivamente, i **vettori singolari sinistri e destri di A** associati ai valori singolari σ_i e formano una base ortonormale di R^m e R^n , rispettivamente, poichè vale la relazione:

$$Au_i = \sigma_i v_i$$

Questo teorema porta a un'implicazione piuttosto interessante, infatti si ha:

$$A^T A = (U\Sigma V^T)^T (U\Sigma V^T) = V\Sigma^T U^T U\Sigma V^T$$

dato che U è ortogonale si "annulla" con la sua trasposta, quindi continua con

$$V\Sigma^T \Sigma V^T = V \begin{pmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n^2 \end{pmatrix} V^T$$

Inoltre:

$$\sigma_i = \sqrt{\lambda_i}, i = \dots, n$$

Dove λ_i non è che l'autovalore di $A^T A$, si ha quindi:

- $\sigma_1 = \sqrt{\lambda_{max}} = \sqrt{\rho} = \|A\|$
- $\sigma_k = \sqrt{\lambda_{min}} \Rightarrow \|A^{-1}\| = \frac{1}{\sigma_k}$

3.2.6 numero di condizione

Si dice **numero di condizione** di A tale valore:

$$K_2(A) = \frac{\sigma_1}{\sigma_k}$$

Dove σ_k è il valore singolare minimo di A e k è il rango della matrice, quindi $k \leq \min(m, n)$

Un'altra definizione di questo valore è:

$$K_2(A) = \|A\| \cdot \|A^{-1}\|$$

Un **numero di condizionamento basso** (vicino a 1) indica che la matrice è ben condizionata. Ciò significa che piccole variazioni negli input producono piccole variazioni negli output, viceversa un numero di condizionamento alto indica che la matrice è mal condizionata. Ciò significa che piccole variazioni negli input possono causare grandi variazioni negli output

3.3 Problema dei minimi quadrati

Si consideri il seguente sistema:

$$Ax = b$$

Dove $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $x \in \mathbb{R}^n$ e $m > n$. È facile verificare che risulta indeterminato, dato che il numero di equazioni è superiore al numero di incognite, non ammettendo, quindi, alcuna soluzione.

Consci di questo fatto alquanto deprecabile e accettata la sua inalterabilità, si vuole però cercare la x tale che l'espressione $Ax - b$ non sia 0 ma "quasi", ovvero il più piccolo valore possibile. Si è così giunti al concetto di problema ai minimi quadrati:

Si definisce il **problema ai minimi quadrati (LSQ)**, che consiste nel determinare il vettore $x \in \mathbb{R}^n$ che minimizzi il vettore dei residui $r = Ax - b$:

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2 = \min_{x \in \mathbb{R}^n} \|r\|_2^2$$

dove $r = (r_1, r_2, \dots, r_n) = (Ax_1 - b_1, \dots, Ax_n - b_n)$

Si giunge così al seguente teorema:

teorema

Sia $A \in \mathbb{R}^{m \times n}$ con $m > n$ e $rg(A) = k \leq n$, allora il problema

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2 = \min_{x \in \mathbb{R}^n} \|r\|_2^2$$

Ammette sempre almeno una soluzione, inoltre:

- $k = n$ (**quindi ha rango massimo**): il problema ha una ed una sola soluzione
- $k < n$: il problema ha infinite soluzioni e tali soluzioni formano un sottospazio di \mathbb{R}^n di dimensione $n - k$

3.3.1 Equazioni normali

Come si fa, quindi, a minimizzare questo di residuo? Bhe bisogna trovare un "punto di minimo" che ovviamente va trovato con le derivate (nel nostro caso dato che siamo in \mathbb{R}^n il gradiente). Intanto sappiamo che:

$$\begin{aligned}\|Ax - b\|_2^2 &= (Ax - b)^T (Ax - b) \\ &= (-b^T + x^T A^T) (Ax - b) \\ &= -b^T Ax + b^T b + x^T A^T Ax \\ &\quad \text{(sommando i termini simili)} \\ &\quad b^T Ax = x^T A^T b \\ &= x^T A^T Ax - 2x^T Ab + b^T b\end{aligned}$$

Si pone:

$$f(x) = x^T A^T Ax - 2x^T Ab + b^T b$$

Questa è un'equazione $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$, condizione necessaria (ma non sufficiente) che un punto x^* sia di minimo è che $\nabla f(x^*) = 0$ dove:

$$\nabla f = \left(\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n} \right)$$

Dato che f non è altro che una sommatoria di tre addendi, non mi resta che fare il gradiente di questi per poi addizionarli. Quindi:

- $\nabla(x^T Ax) = 2A^T Ax$
- $\nabla(2x^T Ab) = 2A^T b$
- $\nabla(b^T b) = 0$

Allora:

$$\nabla f(x) = 2A^T Ax - 2A^T b$$

Imponendo che il gradiente sia nullo si ottiene il cosiddetto **Sistema delle equazioni normali**

$$A^T Ax = A^T b$$

Se $A^T A$ è simmetrica e definita positiva allora A ha rango massimo, è possibile quindi risolvere il sistema con un opportuno metodo adatto alla matrice $A^T A$, ad esempio utilizzando la fattorizzazione di Cholesky, si ottiene una matrice L t.c. $A^T A = LL^T$ e si risolvono nell'ordine i due sistemi:

$$\begin{cases} Ly = A^T b \\ L^T x = y \end{cases}$$

Tuttavia, il calcolo di $A^T A$ può rendere il problema eccessivamente mal condizionato poiché $K(A^T A) = K(A)^2$, ovvero il numero di condizionamento della matrice $A^T A$ è approssimativamente il quadrato del numero di condizionamento della matrice A . Ricordiamo che un problema mal condizionato è più difficile da risolvere numericamente, poiché anche piccoli errori di arrotondamento nei calcoli possono causare errori significativi nella soluzione finale

3.3.2 Decomposizione in valori singolari (SVD)

Se il rango $k < \min(m, n)$ allora vi sono infinite di soluzioni, ma ne esiste una sola \bar{x} di norma minima. Tale soluzione si può ottenere tramite SVD

teorema

Sia $A \in \mathbb{R}^{m \times n}$, con $rg(A) = k \leq n$ e sia $A = U\Sigma V^T$ la sua decomposizione in valori singolari, allora il vettore:

$$x^* = \sum_i^k \frac{u_i^T b}{\sigma_i} v_i$$

è la soluzione di minima norma al problema:

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2$$

per la proprietà del vettore ortogonale di non alterare le norme di vettori e in corrispondenza di tale soluzione e si ottiene

$$\|Ax - b\| = \|U^T Ax - U^T b\| = \|U^T A V V^T x - U^T b\|$$

Posto $y = V^T x \in \mathbb{R}^n$ e $g = U^T b \in \mathbb{R}^m$ si ha:

$$\|r\| = \|Ax^* - b\| = \sum_{i=k+1}^n (u_i^T b)^2$$

Moltiplicando U^T a destra e a sinistra si ottiene:

$$\|Ax - b\| = \|\Sigma y - g\| = \sum_{i=1}^k (\sigma_i y_i - g_i)^2 + \sum_{i=k+1}^n g_i^2$$

Per minimizzare tale quantità è sufficiente scegliere:

$$y_i = \frac{g_i}{\sigma_i} = \frac{U^T b}{\sigma_i}$$

poichè $x^* = Vy$ risulta:

$$x^* = \sum_{i=1}^k \frac{u_i^T b}{\sigma_i} v_i$$

Allora in corrispondenza di tale soluzione, la norma del residuo è:

$$\|r\| = \sum_{i=k+1}^n (g_i)^2 = \sum_{i=k+1}^n (u_i^T b)^2$$

3.3.3 Matrice pseudoinversa

Qui viene riportata la definizione di una matrice con proprietà molto utili:

Sia $A \in \mathbb{R}^{m \times n}$, con $rg(A) = k \leq n$ e sia $A = U\Sigma V^T$ la sua decomposizione in valori singolari. Si definisce **pseudoinversa** la matrice:

$$A^+ = V\Sigma^+ U^T$$

Dove:

$$(\Sigma)_{ij} = \begin{cases} \frac{1}{\sigma_i} & \text{se } i = j \wedge i \leq k \\ 0 & \text{altrimenti} \end{cases}$$

Ovvero:

$$\Sigma^+ = \begin{pmatrix} \frac{1}{\sigma_1} & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \frac{1}{\sigma_2} & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \cdots & 0 \\ 0 & 0 & \cdots & \frac{1}{\sigma_k} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & \cdots & 0 \end{pmatrix}_{m \times n}$$

Questa giga matrice gode delle seguenti proprietà:

1. $AA^+A = A$
2. $A^+AA^+ = A^+$
3. $(AA^+)^T = AA^+$
4. $(A^+A)^T = A^+A$

La pseudoinversa di una matrice rettangolare A permette di scrivere la soluzione del problema dei minimi quadrati in modo simile alla soluzione $x = A^{-1}b$ di un sistema lineare quadrato, cioè

$$x^* = V\Sigma U^T b \implies x^* = A^+ b$$

Tuttavia sebbene la pseudoinversa sia teoricamente potente per analizzare il problema dei minimi quadrati e comprendere le proprietà delle soluzioni, il calcolo diretto della pseudoinversa di una matrice è spesso inefficiente e computazionalmente oneroso, soprattutto per matrici di grandi dimensioni. Questo è dovuto principalmente al costo della decomposizione ai valori singolari (SVD), che richiede un notevole sforzo computazionale. Pertanto non è uno strumento adeguato per calcolarle poiché computazionalmente costoso.

Si può inoltre definire il numero di condizionamento di $A \in \mathbb{R}^{m \times n}$ in termini di pseudoinversa:

$$K(A) = \|A\| \|A^+\|$$

È il numero di condizionamento in norma -2 (o condizionamento spettrale):

$$K(A)_2 = \|A\|_2 \cdot \|A^+\|_2 = \begin{cases} \frac{\sigma_1}{\sigma_n} & k = \text{vg}(A) \\ \frac{\sigma_1}{\sigma_k} & k = \text{rg}(A) \end{cases}$$

Dove:

- **k=vg(A)**: quando k è uguale al numero di colonne di A , si sta considerando la situazione in cui la matrice ha tutte le colonne linearmente indipendenti
- **k=rg(A)**: quando k è uguale al rango effettivo di A , ovvero il numero massimo di colonne linearmente indipendenti

Il parametro di regolarizzazione, come già visto nel caso dei problemi inversi 1D, può essere scelto in modo euristico oppure applicando il criterio di Massima Discrepanza. Per verificare sperimentalmente il parametro “ottimale” nel caso in cui si abbia la soluzione esatta (cioè in un problema test) è quello di verificare quale parametro rispetto a quelli scelti produce l’errore minore rispetto alla “ground truth” x_{GT} , utilizzando per esempio l’errore relativo già definito in precedenza:

$$ER = \frac{\|x - x_{GT}\|_2^2}{\|x_{GT}\|}$$

4 Algoritmi per il calcolo di sistemi lineari

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{cases}$$

Con m equazioni ed n incognite. E sia $m = n$, corrispondente ad un sistema quadrato

Sia $Ax = b$ con

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ a_{21} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix}$$

e

$$x \in \mathbb{R}^n = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, b \in \mathbb{R}^n = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Allora si ha che il sistema $Ax = b$ ha **una ed una sola soluzione se e solo se A non è singolare** ($\det A \neq 0$)

$$Ax = b \iff A^{-1}Ax = A^{-1}b \iff x = A^{-1}b$$

Dove **il Calcolo di A^{-1} complessità computazionale $O(n^3)$**

A questo proposito ci vengono in aiuto diversi metodi per il calcolo:

- **Metodi Diretti:** la soluzione viene calcolata in un numero finito di passi modificando la matrice del problema in modo da rendere più agevole il calcolo della soluzione.

Es.

- Matrici triangolari: Metodi di Sostituzione
- Fattorizzazione LU
- fattorizzazione di Cholesky,

- **Metodi Iterativi:** Calcolo di una soluzione come limite di una successione di approssimazioni x_k , senza modificare la struttura della matrice A . Adatti per sistemi di grandi dimensioni con matrici sparse (pochi elementi non nulli)

4.1 Metodi diretti

Iniziamo con la definizione di matrici triangolari:

Una matrice $M \in \mathbb{R}^{n \times n}$ è detta **triangolare inferiore** se:

$$L = \begin{pmatrix} l_{11} & 0 & 0 & \dots & 0 \\ l_{21} & l_{22} & 0 & \dots & 0 \\ l_{31} & l_{32} & l_{33} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \dots & l_{nn} \end{pmatrix}$$

Dove l_{ij} per $i < j$, ossia tutti gli elementi sopra la diagonale (con $j > i$) sono nulli

Da cui discende la definizione di **sistema triangolare inferiore**:

Un **sistema triangolare inferiore** è un sistema lineare di equazioni in cui la matrice dei coefficienti è una matrice triangolare inferiore, e si presenta nella seguente forma:

$$\begin{pmatrix} a_{1,1} & 0 & 0 & \dots & 0 \\ a_{2,1} & l_{2,2} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ a_{n-1,1} & a_{n-1,2} & \dots & a_{n-1,n-1} & 0 \\ a_{n,1} & a_{n,2} & a_{n,3} & \dots & a_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n-1} \\ b_n \end{pmatrix}$$

e in un sistema lineare:

$$\begin{cases} a_{1,1}x_1 = b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 = b_2 \\ a_{3,1}x_1 + a_{3,2}x_2 + a_{3,3}x_3 = b_3 \\ \vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + \dots + a_{n,n}x_n = b_n \end{cases}$$

Adesso c'è contrapposto con il **sistema triangolare superiore**

Una matrice $M \in \mathbb{R}^{n \times n}$ è detta triangolare superiore se:

$$L = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ 0 & a_{2,2} & a_{2,3} & \cdots & a_{2,n} \\ 0 & 0 & a_{3,3} & \cdots & a_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{n,n} \end{pmatrix}$$

Dove $u_{ij} = 0$ per $i > j$, ossia tutti gli elementi sotto la diagonale (con $i > j$) sono nulli.

Da cui discende la definizione di **sistema triangolare superiore**

Un **sistema triangolare superiore** è un sistema lineare di equazioni in cui la matrice dei coefficienti è una matrice triangolare superiore, e si presenta nella seguente forma:

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n-1} & a_{1,n} \\ 0 & a_{2,2} & a_{2,3} & \cdots & a_{2,n-1} & a_{2,n} \\ 0 & 0 & a_{3,3} & \cdots & a_{3,n-1} & a_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & a_{n-1,n-1} & a_{n-1,n} \\ 0 & 0 & 0 & \cdots & 0 & a_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{n-1} \\ b_n \end{pmatrix}$$

e in un sistema lineare:

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n = b_1 \\ a_{2,2}x_2 + a_{2,3}x_3 + \cdots + a_{2,n}x_n = b_2 \\ a_{3,3}x_3 + \cdots + a_{3,n}x_n = b_3 \\ \vdots \\ a_{n,n}x_n = b_n \end{cases}$$

4.1.1 algoritmo delle sostituzioni

Quando si ha a che fare con matrici triangolari la soluzione può essere facilmente calcolata attraverso l'**algoritmo delle sostituzioni**, che si differenzia nel caso di matrici triangolari inferiori o superiori

- Nel caso di matrici triangolari inferiori l'algoritmo prende il nome di **sos-**

tituzioni all'indietro ed è descritto dal sistema qui sotto:

$$\begin{cases} x_n = \frac{b_n}{a_{nn}} \\ x_i = \frac{b_i - \sum_{k=i+1}^n a_{ik} x_k}{a_{ii}} \end{cases} \quad i = n-1, n-2, \dots, 1$$

- Nel caso di matrici triangolari superiori l'algoritmo prende il nome di **sostituzioni in avanti** ed è descritto dal sistema qui sotto:

$$\begin{cases} x_1 = \frac{b_1}{a_{11}} \\ x_i = \frac{b_i - \sum_{k=1}^{i-1} a_{ik} x_k}{a_{ii}} \end{cases} \quad i = 2, 3, \dots, n$$

In entrambi i casi il numero di operazioni richiesto è $\simeq \frac{n(n+1)}{2} = O(n^2)$

4.1.2 Fattorizzazione LU

Considerata una matrice $A \in \mathbb{R}^{n \times n}$, una fattorizzazione LU fattorizza la matrice A nella forma

$$A = LU$$

con

- L matrice $n \times n$ triangolare inferiore
- U matrice $n \times n$ triangolare superiore

Per trasformare una matrice A in una matrice LU , è possibile utilizzare l'algoritmo di Gauss per ridurla a una matrice triangolare superiore U ed occorrerà inserire le operazioni di eliminazione nella matrice L (inizializzata a una matrice identità). È possibile dimostrare che $A = L \times U$

Ecco l'algoritmo in pseudocodice, assumendo che la matrice A sia ben condizionata e pertanto non si necessita uno scambio di righe:

Algorithm 3: Fattorizzazione LU

Input: Intero n , Matrice $A[1..n, 1..n]$
Output: Array $B[1,2]$ contenente le matrici L e U

```
1 Let  $B$  be a new Array[2];
2 Let  $L$  be a new Matrix[1..n, 1..n];
3 Let  $U$  be a new Matrix[1..n, 1..n];
4 for  $i = 1$  to  $n$  do
5   for  $j = 1$  to  $n$  do
6     if  $i == j$  then
7        $L[i, j] \leftarrow 1$ ;
8     else
9        $L[i, j] \leftarrow 0$ ;
10 Let  $p$  be a new Real; // dichiaro il pivot
    // colonne per pivot
11 for  $i = 1$  to  $n$  do
12    $p \leftarrow A[i, i]$ ; // aggiorno il pivot
    // righe
13   for  $j = i + 1$  to  $n$  do
14      $L[j, i] \leftarrow \frac{A[j, i]}{p}$ ; // fattore moltiplicativo in L
15     for  $k = i$  to  $n$  do
16        $A[j, k] \leftarrow A[j, k] - L[j, i] \times A[i, k]$ ;
17  $B[1] \leftarrow L$ ;
18  $B[2] \leftarrow A$ ;
19 return  $B$ ;
```

Ecco un esempio pratico:

Supponiamo di voler fattorizzare la matrice:

$$A = \begin{pmatrix} 2 & 3 \\ 4 & 7 \end{pmatrix}$$

Poi inizializzo L come una matrice identità e $U = A$

$$L = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad A = \begin{pmatrix} 2 & 3 \\ 4 & 7 \end{pmatrix}$$

Poi eseguo l'algoritmo di Gauss in U avendo come pivot 2, in questo caso occorre solo modificare solo la seconda in quanto la matrice ha solo 2 righe. Il fattore moltiplicativo nella riga 2 e colonna 1 è $l_{2,1} = \frac{4}{2} = 2$, si può procedere così con l'eliminazione Gaussiana ottenendo:

$$U = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$$

Si inserisca, inoltre, $l_{2,1}$ nella matrice L :

$$L = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$$

La fattorizzazione, così, è finita e si ha $A = L \times U$:

$$\begin{pmatrix} 2 & 3 \\ 4 & 7 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$$

4.1.3 Eliminazione di Gauss

Si eliminano le incognite in modo sistematico per trasformare il sistema lineare in uno equivalente con matrice a struttura triangolare superiore, più precisamente si introduce una successione di matrici tale che:

$$A^{(k)} = (a_{ij}^{(k)}), \quad k = 1, \dots, n$$

Con $A^{(1)} = A$ e $A^{(n)} = U$ e dove la matrice $A^{(k+1)}$ ha tutti gli elementi $\{a_{ij}^{(k+1)}, j+1 \leq i \leq n, 1 \leq j \leq k\}$ nulli. Ovvero essa differisce da $A^{(k)}$ per avere **gli elementi sottodiagonali della colonna k-esima nulli**. Risparmio ulteriori spiegazioni di come funziona e passo direttamente alla formula:

$$m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \\ a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}$$

Per portare a termine

Inoltre è possibile che bisogna scambiare le righe, ad esempio quando $\det(A_k) = 0$ e pertanto il $a_{kk} = 0$, Perciò occorre scambiarli con altre righe (ad esempio la riga p). Si noti che lo scambio di righe i, j di una matrice A equivale a moltiplicarla per una matrice P^{ij} tale che:

$$(P^{ij})_{km} = \begin{cases} \delta_{km} & k \neq i, j \\ 0 & k = m = i, j \\ 1 & k = i, m = j \\ 1 & k = j, m = i \end{cases}$$

La quale verrà chiamata **matrice di permutazione**

4.1.4 algoritmo di gauss rivisto come metodo di fattorizzazione LU

Si può dimostrare che l'algoritmo di Gauss realizza la seguente fattorizzazione della matrice A :

$$A = LU$$

Dove:

- $U = A^{(n)}$ è una matrice triangolare superiore
- L è la matrice triangolare inferiore dei moltiplicatori, ovvero:

$$L = \begin{cases} L_{ij} = m_{ij} & i < j \\ L_{ij} = 1 & i = j \\ L_{ij} = 0 & j < i \end{cases}$$

dimostrazione

Sia $M_k = I - m_k e_k^T$, dove:

- $(e_k)_j = \delta_{kj}$ quindi $e_k^T = [0, 0, \dots, 1, \dots, 0]$ Dove 1 è alla posizione

- $m_k = \begin{pmatrix} 0 \\ \dots \\ m_{k+1,k} \\ m_{k+2,k} \\ \dots \\ m_{nk} \end{pmatrix}$

Che equivale in termini di componenti è uguale a:

$$(M_k)_{ip} = \delta_{ip} - (m_k e_k^T)_{ip} = \delta_{ip} - m_{ik} \delta_{kp}$$

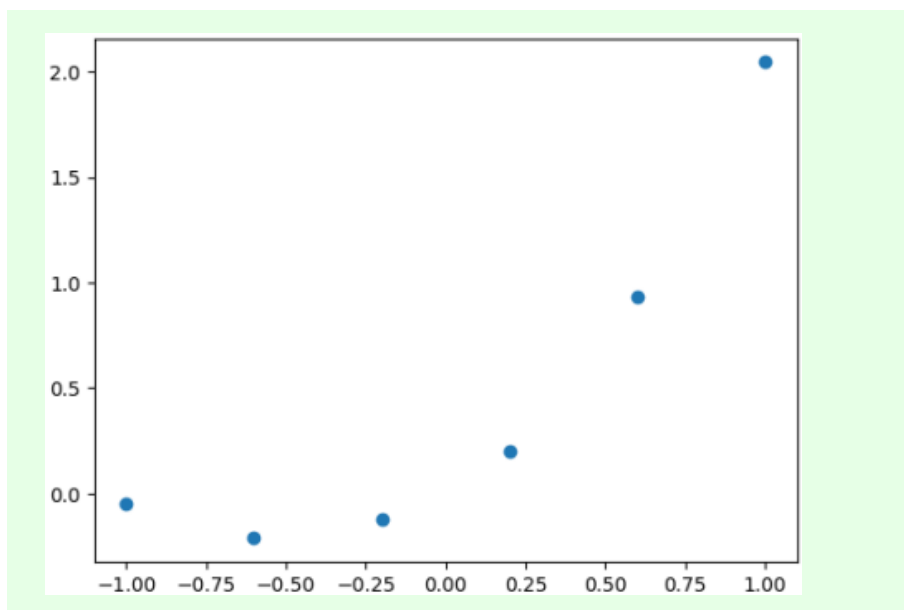
Dalle formule dell'algoritmo di gauss si ha:

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ij}^{(k)} a_{ij}^{(k)} = a_{ij}^{(k)} - m_{ik} \delta_{kk} a_{kj}^{(k)} = \sum_p (\delta_{ip} - m_{ik} \delta_{kp}) a_{pj}^{(k)}$$

5 Problema di interpolazione

Dati $n + 1$ punti distinti $(x_0), (x_1, y_1), \dots, (x_n, y_n)$ con $x_0 < x_1 < \dots < x_n$ chiamati **nodi**, si definisce il **problema di interpolazione** quel problema che consiste nel trovare una funzione (polinomiale) $p(x)$ chiamata **interpolante**, tale che $\forall k \in [0, n], p(x_k) = y_k$

Esempietto con alcuni nodi:



Si può risolvere il polinomio di grado n come:

$$p(x) = c_0 + c_1x + c_2x^2 + \dots + c_nx^n$$

Per identificare questo polinomio, ordunque, serve **calcolare i coefficienti** $c_i, i \in [1, n]$

5.1 Esistenza dell'unicità

teorema

$\forall i \in [1, n], \forall (x_i, y_i)$ con i nodi t_i , distinti tra loro, esiste un unico polinomio di grado minore od uguale a n , che indichiamo $P_n(x)$ con e chiamiamo polinomio interpolatore dei valori y_i nei nodi t_i , tale che

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

Per la dimostrazione serve prima enunciare un lemma (che non verrà dimostrato)

lemma

Sia $r_n(x)$ un polinomio a coefficienti reali di grado n , il numero massimo di zeri distinti è n , ovvero:

- Un polinomio di grado n può annullarsi in al massimo n punti distinti.
- Se un polinomio di grado n si annulla in più di n punti distinti, allora deve essere il polinomio nullo, cioè identicamente uguale a zero per ogni valore di x

dimostrazione

Assumiamo che esistano due polinomi distinti di grado n , ovvero $p_n(x)$ e $q_n(x)$ che soddisfino entrambi le relazioni nodali precedenti. La loro differenza $p_n(x) - q_n(x)$ è ovvio verificare che è un polinomio di grado n che si annulla in $n + 1$ punti distinti. Per il fatto che il polinomio si annulla in $n + 1$ punti e per il lemma enunciato si ha che $p_n(x) - q_n(x) = 0$, quindi $p_n(x) = q_n(x)$. Per assurdo dimostrato
Q.e.d

5.2 Calcolo del polinomio di interpolazione

5.2.1 Metodo classico

È ovvio verificare che i coefficienti del polinomio di interpolazione di $n + 1$ punti $(x_i, y_i) = (x_i, f(x_i))$ si potrebbero calcolare risolvendo il sistema lineare:

$$\begin{cases} c_0 + c_1x_1 + \dots + c_nx_1^n = y_1 \\ c_0 + c_1x_2 + \dots + c_nx_2^n = y_2 \\ \dots \\ c_0 + c_1x_n + \dots + c_nx_n^n = y_n \end{cases}$$

Tuttavia la matrice dei coefficienti di questo sistema, detta **matrice di Vandermonde** può essere molto mal condizionata. Inoltre la risoluzione del sistema

lineare richiede almeno $O(\frac{n^3}{3})$

5.2.2 Metodo del polinomio interpolatore di Lagrange

Introduciamo i polinomi base di lagrange

Si chiamano **polinomi base di Lagrange** di grado n particolari funzioni tali che:

$$\phi_k(x_j) = \delta_{k,j}$$

dove $k = j \implies \delta_{k,j} = 1$ e $k \neq j \implies \delta_{k,j} = 0$

La funzione $\phi_k(x)$ può essere scritta come:

$$\phi_k(x) = \prod_{j=0, j \neq k}^n \frac{x - x_j}{x_k - x_j}$$

Si può notare, infatti, che include tutti i punti x_j tranne x_k , quindi l'indice j percorre tutti gli indici da 0 a n e nel caso in cui la x in input sia diversa da x_k vi sarà un x_j tale che $x_j = x$, in quel caso il numeratore farà 0 e renderà nulla la productoria. Nel caso, invece, la x in input sia uguale a x_k avremo $\frac{x_k - x_j}{x_k - x_j} \forall j$ rendendo la productoria uguale ad 1.

Definito il polinomio posso scrivere il polinomio di interpolazione di Lagrange:

Si chiama **polinomio interpolatore di Lagrange** tale funzione:

$$p_n(x) = y_0\phi_0(x) + y_1\phi_1(x) + \dots + y_n\phi_n(x) = \sum_{k=0}^n y_k\phi_k(x)$$

Infatti soddisfa le condizioni di interpolazione

$$p_n(x_i) = \sum_{k=0}^n y_k\phi_k(x_i) = \sum_{k=0}^n y_k\delta_{i,k} = y_i$$

5.3 Errore di un polinomio interpolatore di una funzione

Iniziamo con la definizione di un polinomio interpolatore di una funzione

Sia $f[0, n] \rightarrow \mathbb{R}$ una funzione continua nel suo dominio e sia $y_i = f(x_i), \forall i \in [0, n]$. Sia anche $p_n(x)$ il polinomio interpolatore nei punti $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$. Allora $p_n(x)$ è detto **polinomio interpolatore di f** e si denota:

$$p_n(f)$$

Attraverso il polinomio di Lagrange è possibile quantificare l'errore che si commette sostituendo ad una funzione f il suo polinomio interpolante attraverso il seguente teorema

teorema

Sia I un intervallo limitato, e si considerino $n + 1$ nodi di interpolazione distinti $x_i, i = 0, \dots, n$ in I . Sia f derivabile con continuità fino all'ordine $n + 1$ in I . Allora

$$\forall x \in I. \exists \eta \in I : E(x) = f(x) - p_n(x) = \frac{f^{(n+1)}(\eta)}{(n+1)!} \prod_{i=0}^n (x - x_i)$$

E dove ovviamente $E(x_i) = 0$ dove $i = 0, \dots, n$

L'obiettivo di questo teorema, quindi, è fornire una formula da cui ricavare un **errore di interpolazione** $E(x)$, che è la differenza tra la funzione originale $f(x)$ e il polinomio interpolante $p_n(x)$ nei punti in cui non conosciamo il valore esatto di $f(x)$ (ovvero tutti i punti diversi dai nodi)

Da questo **teorema non riusciamo a dedurre che**, qualsiasi sia la distribuzione dei nodi, $\max_{x \in I} |E(X)| \rightarrow 0$ per $n \rightarrow \infty$ ovvero (**che il massimo valore dell'errore** $E(x) = f(x) - p_n(x)$ su un intervallo I **tende a zero al crescere di n**). Esistono, infatti, alcune funzioni, come la funzione di Runge, in cui per determinate scelte dei nodi, per esempio equidistanti dove $\max_{x \in I} |E(X)| \rightarrow \infty$ per $n \rightarrow \infty$. Alla luce di questa considerazione si ha che ad un aumento del grado n del polinomio interpolatore non corrisponde necessariamente un miglioramento nella ricostruzione di una funzione f .

Il fenomeno di Runge può essere evitato utilizzando opportune distribuzioni di nodi. In particolare, su un arbitrario intervallo $[a, b]$ consideriamo i cosiddetti **nodi di Chebyshev- Gauss-Lobatto**:

$$x_i = \frac{a+b}{2} - \frac{b-a}{2} \cos\left(\frac{i\pi}{n}\right) \quad \text{con } i = 0, \dots, n$$

Notiamo che $x_i = \cos\left(\frac{i\pi}{n}\right)$ se $[a, b] = [-1, 1]$. Si può dimostrare che se f è una funzione continua e derivabile con continuità in $[a, b]$ il polinomio interpolante $p_n(x)$ converge a $f(x)$ per $n \rightarrow \infty$

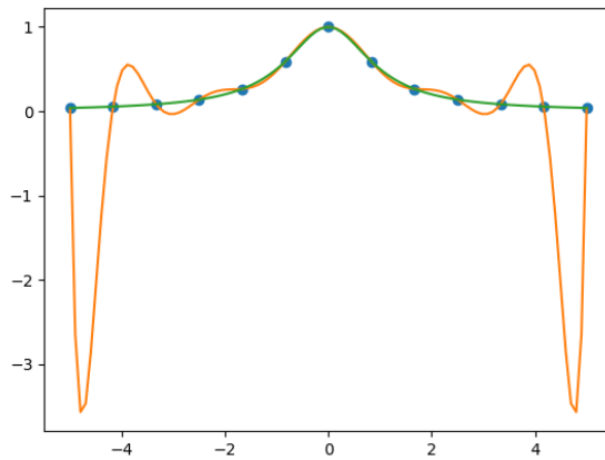


Figure 1: In questa immagine troviamo la funzione di Runge (in arancione) e il suo polinomio interpolante (in verde), dove la scelta dei nodi ha causato un errore di interpolazione elevato.

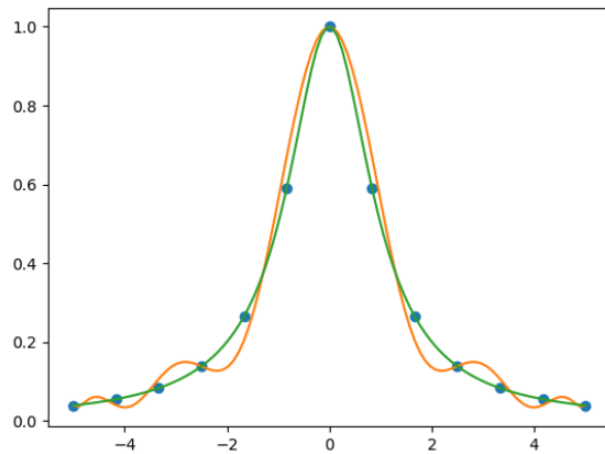


Figure 2: in questa immagine la scelta dei nodi ha causato un errore di interpolazione relativamente basso

5.4 Condizionamento dell'interpolazione polinomiale

Alle volte non si dispone della funzione $f(x_i)$ relativi ai nodi x_i , con $i = 0, \dots, n$ in un intervallo I ma solo della sua approssimazione, denotata $\tilde{f}(x_i)$. La perturbazione $f(x_i) - \tilde{f}(x_i)$ potrebbe essere dovuta ad esempio all'effetto degli errori di arrotondamento o ad errori di misura. Se chiamiamo $\tilde{p}_n x$ il polinomio

interpolatore dei nodi $(x_i, \tilde{f}(x_i))$, si ha:

$$\max_{x \in I} |p_n(x) - \tilde{p}_n(x)| = \max_{x \in I} \left(\sum_{i=0}^n (f(x_i) - \tilde{f}(x_i) \phi_i(x)) \right) = \Lambda_n(x) \max |f(x_i) - \tilde{f}(x_i)|$$

Dove $\Lambda_n(x)$ è il numero di condizione che in questo caso è:

$$\Lambda_n(x) = \max_{x \in I} \left| \sum_{i=0}^n \phi_i(x) \right|$$

Che viene detta **costante i Lebesgue** che nel caso dell'interpolazione polinomiale su nodi equispaziati la costante di Lebesgue assume il valore:

$$\Lambda_n(x) \simeq \frac{2^{n+1}}{n(\log(n) + \gamma)e}$$

Dove $\gamma \simeq 5.47721$ è la **costante di Eulero**. Ciò comporta che per n grande questo tipo di interpolazione potrebbe essere instabile

6 Problemi diretti

Nel mondo reale i vari fenomeni sono descritti dalla fisica attraverso un problema diretto come

$$\text{causa} \rightarrow \text{modello} \rightarrow \text{effetto}$$

Per esempio, se voglio calcolare l'accelerazione di un'auto, ne individuo i vari passaggi

- La causa: ovvero la forza generata dal motore
- Modello: la leggi di newton che forniscono le formule e le leggi apposite (supponendo un sistema inerziale)
- Effetto: calcolo dell'accelerazione

Quindi acquisisco in input l'origine dell'effetto (ovvero la forza) attraverso un modello (leggi di newton) elaboro questo dato ed infine ne calcolo l'effetto finale (l'accelerazione). In si può scrivere "matematica": $x \rightarrow A \rightarrow y$, che traslato in algebra lineare si ha così:

$$Ax \rightarrow y$$

Si giunge così alla definizione di **problema diretto**

Si definisce **problema diretto** il processo mediante il quale si determina l'effetto dato un sistema e le cause iniziali

In altre parole: input \rightarrow output

7 Problemi inversi

D'altra parte esistono dei casi in cui si riesce solo ad osservare l'effetto di un fenomeno ed occorre, partendo da questo, risalire alla causa, da questo concetto abbiamo la definizione di **problema inverso**

Un **problema inverso** è un tipo di problema matematico in cui, dato l'effetto o l'output osservato, si cerca di determinare le cause o i parametri del sistema che lo hanno generato

Che in termini matematici significa determinare una x conoscendo A e y , ovvero output \rightarrow input.

Tuttavia **nel mondo reale i dati y sono sempre affetti da errore (detto anche rumore)** rappresentato dal vettore δ che non sono solo errori di rappresentazione ma anche legati a problemi di tipo fisico, forti di questa consapevolezza il problema diventa quindi:

$$Ax = y + \delta = y^\delta$$

7.1 Problemi inversi lineari con matrici mal condizionate

Vi sono delle classi di problemi la cui matrice A relativa al modello matematico del problema è mal condizionata, tali problemi si dicono **mal posti**. In questo caso il problema viene risolto come un problema di minimi quadrati (che permette di avere anche una matrice rettangolare):

$$\min_x \|Ax - y^\delta\|_2^2$$

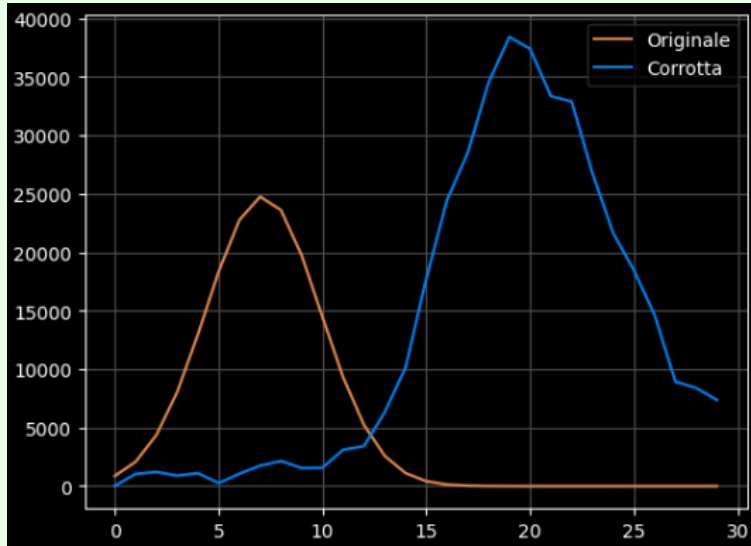
la cui soluzione è

$$x^* = \sum_{i=1}^n \frac{u_i^T y^\delta}{\sigma_i} v_i$$

7.2 Perturbazione dell'errore

A causa di un mal condizionamento di A si ha che, in presenza di un errore δ , si ha che questo errore venga eccessivamente amplificato nella soluzione dei minimi quadrati

Si consideri una soluzione "vera" (ground truth) denotata con x_{GT} a cui si applica una matrice A che rappresenta il sistema di acquisizione dati per ottenere il vettore $y : Ax = y$
 Al vettore y si aggiunge un vettore δ che rappresenta il rumore e che e' preso come vettore random da una distribuzione normale con media nulla: $y^\delta = y + \delta$
 Ecco il grafico di questo scempio:



Questo accade perché quando una matrice A è mal condizionata i suoi valori singolari σ_i sono molto piccoli, pertanto il numeratore della sommatoria è molto grande, per capire meglio si osservino i calcoli:

$$\min_x \|Ax - y^\delta\|_2^2 = \min_x \|Ax - (y + \delta)\|_2^2$$

La cui soluzione è:

$$x^* = \sum_{i=1}^n \frac{u_i^T y^\delta}{\sigma_i} v_i = \sum_{i=1}^n \frac{u_i^T (y + \delta)}{\sigma_i} v_i = \sum_{i=1}^n \frac{u_i^T y}{\sigma_i} v_i + \sum_{i=1}^n \frac{u_i^T \delta}{\sigma_i} v_i$$

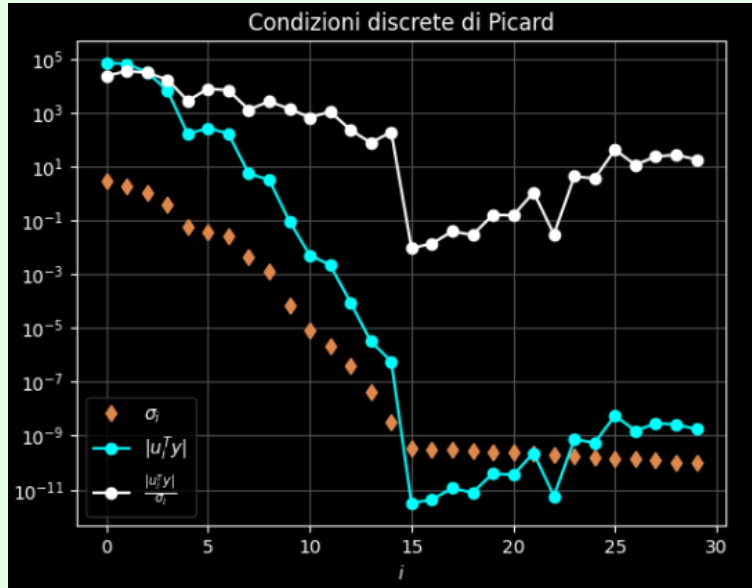
È proprio la seconda sommatoria il problema! È facile verificare che più i σ_i sono piccoli più $u_i^T \delta$ è matematicamente "amplificato"

7.2.1 Condizioni di Picard

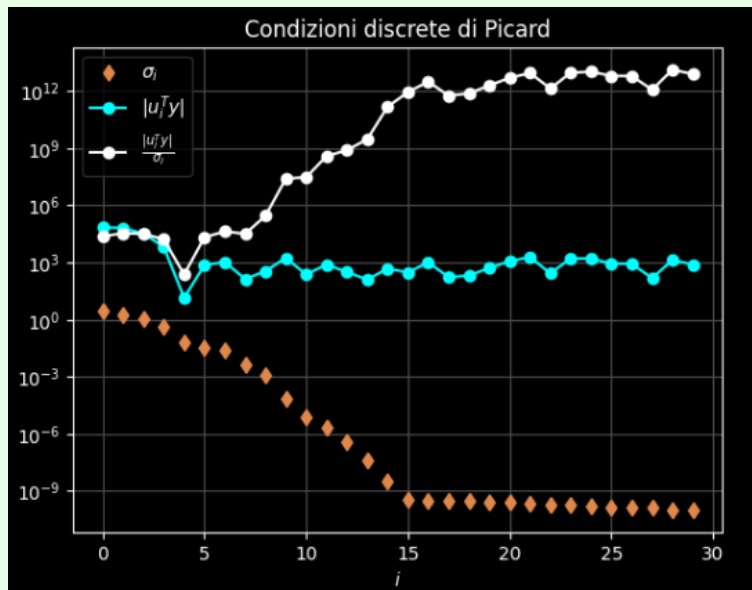
La **condizione discreta di Picard** è un criterio utilizzato per diagnosticare la "malcondizionatura" nei problemi inversi lineari e per capire la stabilità della soluzione. La condizione discreta di Picard si basa sull'analisi dei **valori singolari** σ_i e i cosiddetti **coefficienti di Fourier** $u_i^T y^\delta$

La **condizione discreta di Picard** afferma che, affinché il problema sia ben posto, i coefficienti di Fourier $u_i^T y^\delta$ devono decrescere almeno altrettanto velocemente dei valori singolari σ_i

Si consideri questo esempio:



In questo caso la condizione di Picard è rispettata fino all'indice $i = 15$
 Si consideri quest'altro esempio:



In questo caso la condizione di Picard è rispettata fino all'indice $i = 5$

7.3 Metodi di regolarizzazione

Per affrontare la malcondizionatura e ottenere una soluzione più stabile e accettabile, è **necessario modificare il problema attraverso tecniche dette metodi di regolarizzazione**. Esistono diverse tecniche di regolarizzazione, verranno riportate qui sotto

7.3.1 Decomposizione in Valori Singolari Troncata

Nel metodo di regolarizzazione denominato TSVD (Truncated Singular Value Decomposition) la soluzione del problema di minimi quadrati:

$$\min_x \|Ax - y^\sigma\|_2^2$$

viene calcolata come:

$$x_{TSVD} = \sum_{i=1}^K \frac{u_i^T y^\delta}{\sigma_i} v_i$$

Con $K < n$. Posso anche scrivere la soluzione come:

$$x_{TSVD} = \sum_{i=1}^n f_i \frac{u_i^T y^\delta}{\sigma_i} v_i$$

dove i coefficienti f_i sono detti **fattori di filtro**, dove in questo caso sono dati da:

$$f_i = \begin{cases} 1 & i \leq K \\ 0 & i > K \end{cases}$$

E dove un buon valore K è quello dopo il quale non sono più soddisfatte le condizioni di Picard

7.3.2 Regolarizzazione di Tikhonov

La **regolarizzazione di Tikhonov**, invece, utilizza fattori di filtro che siano numeri in $[0, 1]$ e che quindi “pesino” le componenti della somma in modo più graduale. In particolare, **devono essere pesate di più le componenti di indice piccolo, associate ai valori singolari più grandi, e meno quelle di indice grande, associate ai valori singolari più piccoli**

Il problema diventa quindi:

$$x_{tikh} = \min_x \|Ax - y^\sigma\|_2^2 + \lambda \|Lx\|_2^2$$

Dove:

- $\lambda > 0$ è il parametro di regolarizzazione che pesa la parte di congruenza con i dati e la parte di regolarità della soluzione
- L è la matrice Identità

Si può anche scrivere in questo modo:

$$x_{tikh} = \min_x \|Mx - Y\|_2^2$$

Dove:

- M è la matrice di dimensione $2m \times n$ che ha come blocchi in colonna rispettivamente la matrice A e la matrice λI
- Y è il vettore di dimensione $2m$ che ha come vettori colonna rispettivamente y^δ e il vettore nullo

Si può inoltre dimostrare che la soluzione di Tikhonov posso anche scriverla tramite i fattori di filtro come:

$$x_{tikh} = \sum_{i=1}^n f_i \frac{u_i^T y^\delta}{\sigma_i} v_i$$

dove:

$$f_i = \frac{\sigma_i^2}{\sigma_i^2 + \lambda}$$

7.3.3 Principio di massima discrepanza

Si giunge, tuttavia, ad un problema: la scelta del parametro di regolarizzazione nel metodo di Tikhonov. Infatti questa scelta è sicuramente la parte più delicata della regolarizzazione, un parametro troppo piccolo non toglie il rumore dalla soluzione, mentre un parametro troppo grande produce una soluzione troppo regolare, in cui non è più presente il rumore ma per esempio le oscillazioni o i picchi che ci sono nella soluzione esatta vengono troppo ridotti.

Teoricamente il valore ottimale del parametro λ_{opt} è quello che minimizza:

$$\lambda_{opt} = \min_{\lambda} \|x_{\lambda} - x_{GT}\|_2^2$$

dove x_{λ} è la soluzione calcolata in corrispondenza di un certo λ .

Non esiste un metodo sempre efficiente per scegliere il parametro di regolarizzazione. Sicuramente in pratica spesso si usa una tecnica euristica che consiste nel provare alcuni parametri e scegliere quello che produce la soluzione che ci sembra migliore (trial and test). La tecnica sicuramente più utilizzata in pratica è il principio di massima discrepanza.

Il **Principio di Massima Discrepanza** afferma che il parametro di regolarizzazione λ dovrebbe essere scelto in modo tale che la norma del residuo (cioè la differenza tra i dati osservati e quelli ricostruiti) sia proporzionale alla stima della norma del rumore nei dati

Formalmente:

$$\|Ax_\lambda - y^\delta\| = v_{DP}|\delta|_2^2$$

Dove:

- x_λ è la soluzione regolarizzata con parametro λ
- v è un fattore di proporzionalità (spesso scelto vicino a 1)
- $\|\delta\|$ è la norma dell'errore (spesso è impossibile conoscerla ma si può fare una stima)

Quindi il residuo deve essere uguale alla norma del rumore, quindi provo tanti λ , quello più vicino a questa soluzione è quello migliore

immagini Un problema piuttosto complesso è la conversione di una immagine da analogica a digitale che, ovviamente, deve essere gestita tramite i pixel. Astruendo matematicamente un'immagine, infatti, è possibile trarre una matrice $M \times N$ che rappresenta i vari pixel ed i suoi coefficienti, ai fatti, rappresentano quali sfumature di colore verranno mostrate ed in quale posizione. Tuttavia se un'immagine di grigi è rappresentabile da un'unica matrice che avrà come coefficienti soltanto valori interi da un minimo di 0 ad un massimo di 255 (0 rappresenta il colore nero e 255 rappresenta il colore bianco), un'immagine a colore è ottenuta tramite la sovrapposizione di tre rappresentazioni della stessa immagine: una in scala di rossi, una in scala di verdi ed una in scala di blu; sarà quindi richiesto che il colore di ogni pixel di questa immagine sia rappresentato da vettore di tre valori, ciascuno per ogni intensità del colore rosso, verde e blu

Sia $A \in \mathbb{R}^{M \times N}$ e sia $k \in \mathbb{N}$, diremo che: una matrice B di dimensioni $(M + 2k) \times (N + 2k)$ **estende** la matrice A se e solo se vale:

$$\forall (i, j) \in \{\dots, M\} \times \{1, \dots, N\}. A_{(i, j)} = B_{(i+k, j+k)}$$

Esempietto:

$$A = \begin{pmatrix} 7 & 8 & 9 \\ 12 & 13 & 14 \\ 17 & 18 & 19 \end{pmatrix}$$

Si può affermare che che la seguente matrice

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{pmatrix}$$

estende A , dove si ha $k = 1$

Chiaramente, per ogni matrice ne esistono infinite che la estendono

Sia $K \in \mathbb{R}^{D \times D}$ con D dispari. Si definisce il **centro di matrice** della matrice K come l'elemento di coordinate $(\frac{D+1}{2}, \frac{D+1}{2})$

Sia la matrice di dimensioni 5×5

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{pmatrix}$$

Il suo centro di matrice è $(\frac{5+1}{2}, \frac{5+1}{2}) := 13$

Sia $A \in \mathbb{R}^{M \times N}$ e sia $B \in \mathbb{R}^{(M+(D-1)) \times (N+(D-1))}$ con D dispari che estenda A . $\forall (i, j) \in \{1, \dots, M\} \times \{1, \dots, N\}$ si definisce $W_{(i,j)} \in \mathbb{R}^{D \times D}$ si definiscono **sotto-matrici** di B , il cui centro è $B_{(i+\frac{D-1}{2}, j+\frac{D-1}{2})}$

Sia $A \in \mathbb{R}^{3 \times 3}$

$$\begin{pmatrix} 7 & 8 & 9 \\ 12 & 13 & 14 \\ 17 & 18 & 19 \end{pmatrix}$$

Occorre trovare una matrice $(3+(D-1)) \times (3+(D-1)) = (3+2) \times (3+2)$ che estenda A , ad esempio:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \\ 21 & 22 & 23 & 24 & 25 \end{pmatrix}$$

ne segue che, per esempio, la matrice $W_{(1,3)}$ è la sotto-matrice di B , di dimensioni 3×3 , il cui centro sia $B_{(2,4)}$. Di seguito:

$$W_{(1,3)} = \begin{pmatrix} 3 & 4 & 5 \\ 8 & 9 & 10 \\ 13 & 14 & 15 \end{pmatrix}$$

Sia $A \in \mathbb{R}^{M \times N}$ e $K \in \mathbb{R}^{D \times D}$, con D un numero naturale dispari. Sia B una matrice di dimensioni $(M + (D - 1)) \times (N + (D - 1))$ che estenda A . $\forall (i, j) \in \{1, \dots, M\} \times \{1, \dots, N\}$, siano $W_{(i,j)}$ le sotto-matrici di dimensione $D \times D$ definite precedentemente. Definiamo la matrice $C \in \mathbb{R}^{M \times N}$, come segue:

$$\forall (i, j) \in \{1, \dots, M\} \times \{1, \dots, N\} : C_{i,j} = \sum_{m=1}^{\frac{D-1}{2}} \sum_{n=1}^{\frac{D-1}{2}} K(m, n) W_{(i,j)}(m, n)$$

Tale matrice C , che verrà anche denotata con $C = [K * A | B]$, è chiamata **convoluzione** di K e A rispetto all'estensione B . All'interno del contesto di questa definizione, la matrice K prenderà il nome di **nucleo** di convoluzione o kernel di convoluzione.

8 Estensione di una matrice

Sia k un numero naturale e B una matrice di dimensioni $(M + 2k) \times (N + 2k)$. Definiamo la funzione:

$$\mathcal{P} : \mathbb{R}^{M \times N} \rightarrow \mathbb{R}^{(M+2k) \times (N+2k)} \quad X \rightarrow \mathcal{P}(X)$$

Tale che:

$$\forall (i, j) \in \{1, \dots, M\} \times \{1, \dots, N\} : X_{(i,j)} = \mathcal{P}(X)_{(i+k, j+k)}$$

e:

- le prime k colonne di $\mathcal{P}(X)$ siano uguali alle prime k colonne di B
- le ultime k colonne di $\mathcal{P}(X)$ siano uguali alle ultime k colonne di B
- le prime k righe di $\mathcal{P}(X)$ siano uguali alle prime k righe di B
- le ultime k righe di $\mathcal{P}(X)$ siano uguali alle ultime k righe di B

La funzione \mathcal{P} così definita prenderà il nome di **patter di estensione di ordine k rispetto alla matrice B** .

È chiaro che, per come è definita la funzione $\mathcal{P}(X)$, questa estenderà sempre la matrice X .

Definiamo **funzione di convoluzione** la seguente:

$$[K * (\cdot) | \mathcal{P}] : \mathbb{R}^{M \times N} \rightarrow \mathbb{R}^{M \times N} \quad X \rightarrow [K * X | \mathcal{P}(X)]$$

- Dove K è un nucleo di convoluzione di dimensioni $D \times D$ fissato
- \mathcal{P} è un pattern di estensione di ordine $\frac{D-1}{2}$, rispetto ad una qualche matrice $B \in \mathbb{R}^{(M+(D+1) \times N+(D-1))}$, sempre fissata

Si prendi come esempio tale immagine



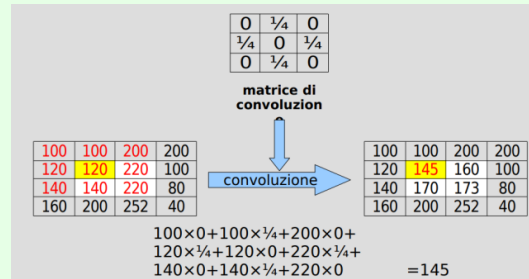
E si supponi sia una matrice A e si supponga tale matrice abbia, come nucleo di convoluzione K , tale matrice:

$$K = \begin{pmatrix} 0 & \frac{1}{4} & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & \frac{1}{4} & 0 \end{pmatrix}$$

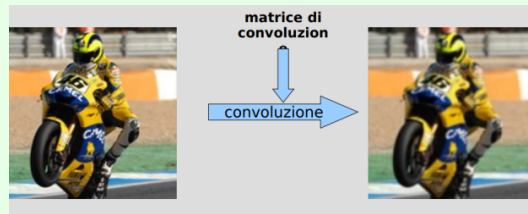
Ora non resta che svolgere la convoluzione in sè. Ecco come:

- consideriamo una sotto-matrice W , sempre di dimensioni $D \times D$, della matrice B
- svolgiamo il prodotto scalare fra: la prima colonna di W e la prima riga di K , la seconda colonna di W e la seconda riga di K , così via fino alla D -esima colonna di W e alla D -esima riga di K
- sommiamo tutti i prodotti scalari ottenuti

- il valore così determinato, diventerà il valore del pixel al centro della sotto-matrice W considerata
- ripetere per tutte le sotto-matrici
- rimuovere le condizioni al bordo



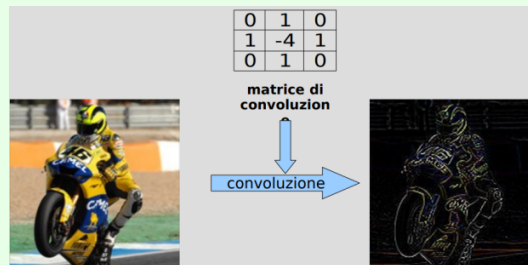
In questo caso, il kernel utilizzato sostituisce il valore di ogni pixel con la media aritmetica dei valori contenuti nei quattro pixel ad esso adiacenti. Ottenendo il seguente effetto di sfocatura:



Se si considera, invece, il seguente kernel:

$$K = (0101 - 41010)$$

Osserviamo che, poiché la somma dei coefficienti è 0, sappiamo che se un pixel ha approssimativamente lo stesso colore dei quattro che gli sono adiacenti, allora verrà rimpiazzato da un valore prossimo a zero; in altre parole, il pixel diverrà di colore nero. Ne segue che tutti e soli i pixel che avranno un valore non nullo, saranno quelli posizionati in mezzo a due regioni di colori contrastanti. Pertanto, questo è un filtro che evidenzia i bordi dell'immagine



9 Point Spread Function

Nel momento in cui un macchinario sensibile alla luce si accinge a fotografare, per esempio, ad una fonte luminosa puntiforme del tipo:

Si ha che l'immagine ottenuta sarà:

L'immagine che si ottiene è, quindi, un allargamento della fonte luminosa centrata nella fonte puntiforme. Questo fenomeno **è del tutto deterministico e legato all'apperacchiatura utilizzata per catturare l'immagine**.

Sia X la prima immagine e Y la seconda, matematicamente si ha che:

$$A(X) = Y$$

Dove \mathcal{A} è una funzione chiamata **Point Spread Function**. Ad essere più precisi riporterò qui la definizione di Point Spread Function

Sia x un immagine che rappresenta perfettamente il nostro oggetto originale. È definita \mathcal{A} Point Spread Function, tale funzione:

$$\mathcal{A}(x) = [K * x | \mathcal{P}]$$

Dove K è un kernel di convoluzione e \mathcal{P} è un pattern di estensione

9.1 PSF con errori non deterministici

Tuttavia gli errori non deterministici non sono solo gli unici errori che corrompono le immagini, nel corso dell'acquisizione dell'immagine fenomeni aleatori e sostanzialmente imprevedibili danneggiano ulteriormente la nitidezza dell'immagine ottenuta.

Adesso sia w un immagine i cui valori dei pixel siano stati selezionati aleatoriamente secondo la distribuzione del rumore Gaussiano bianco.

Si ha, quindi, un modello matematico **discreto** che spieghi come si sia formata l'immagine g (quella restituita dal nostro sistema di formazione di immagini), a partire da x , l'immagine che rappresenta veramente l'oggetto:

$$y^\delta = \mathcal{A}(x) + w = [W * x | \mathcal{P}] + w$$

dove:

- K è un qualche kernel di convoluzione
- \mathcal{P} è un pattern di estensione
- w è il medesimo di cui sopra
- \mathcal{A} è la point spread function

Alternativamente si può riscrivere il modello nella seguente forma matriciale:

$$y^\delta = Ax + w$$

Dove $A \in \mathbb{R}^{M \times N}$ che contiene le informazioni del nucleo di convoluzione ed è tale che:

$$Ax = K * x$$

Adesso ovviamente si vuole ardentemente sapere l'immagine originale x zio pera

9.1.1 Soluzione naive

La soluzione naive è quella ottenuta risolvendo il problema di minimi quadrati:

$$\min_x \|Ax - y^\delta\|_2^2$$

Essendo la matrice A di dimensione non molto piccola, calcolare la soluzione con la SVD è troppo costoso (ricordiamo che la complessità computazionale della fattorizzazione SVD è $O(4/3mn^2)$ per una matrice di dimensione $m \times n$). Quindi si risolvono le equazioni normali:

$$A^T Ax = A^T y^\delta$$

Il sistema ha matrice simmetrica e definita positiva (supponiamo A di rango massimo) e per ragioni computazionali è conveniente utilizzare un metodo iterativo. Quindi si applichiamo il CGLS al sistema delle equazioni normali

9.2 Metriche di valutazione

Le **metriche di valutazione** sono strumenti matematici utilizzati per quantificare la qualità di una ricostruzione, come nel caso della ricostruzione di un'immagine. Servono per confrontare un'immagine ottenuta (ricostruita) con il "ground truth" (l'immagine originale), con l'obiettivo di avere un numero che rappresenti quanto la ricostruzione sia accurata o fedele rispetto all'originale. Ci sono diverse metriche, proprio per avere diversi punti di vista

9.2.1 errore relativo

siano x_{GT} il "ground truth" e x l'immagine ricostruita, viene definito l'errore relativo ER tale valore:

$$ER = \frac{\|x - x_{GT}\|_2^2}{\|x_{GT}\|_2^2}$$

9.2.2 Rapporto Segnale-Rumore di Picco

Prima di definire il Rapporto Segnale-Rumore di Picco occorre prima definire l'errore quadratico medio

Siano M e N le dimensioni dell'immagine, si definisce **l'Errore Quadratico Medio** il seguente valore:

$$MSE = \frac{\|x - x_{GT}\|_2^2}{MN}$$

Questo valore misura la differenza media al quadrato tra i pixel dell'immagine originale e quelli della ricostruzione. adesso si può introdurre il PSNR

Sia $\max_{i,j} |x_{i,j}|$ è il valore massimo dell'immagine ricostruita (ad esempio, 255 per immagini in scala di grigi a 8 bit) e MNE l'errore quadratico medio, si definisce **Rapporto Segnale-Rumore di Picco** tale valore:

$$PSNR = 10 \cdot \log_{10} \left(\frac{(\max_{i,j} |x_{i,j}|)^2}{MSE} \right)$$

Un valore di PSNR più alto indica una qualità della ricostruzione migliore, perché significa che la differenza tra l'immagine ricostruita e l'originale è bassa

9.2.3 Indice di Similarità Strutturale (SSIM)

Il Structural Similarity Index (SSIM) è una metrica più complessa, progettata per essere più coerente con la percezione visiva umana. Considera non solo le differenze di intensità dei pixel, ma anche la struttura, luminanza e contrasto delle immagini. In altre parole, valuta quanto due immagini sono simili in termini di caratteristiche visive piuttosto che pixel per pixel. Più è vicino a uno, migliore è la qualità dell'immagine

9.2.4 Roba

Sia A una matrice $M \times N$, definiamo la funzione $\mathcal{L} : \mathbb{R}^{M \times N} \rightarrow \mathbb{R}^{N \times M}$ come segue:

$$\mathcal{L}(A)$$

è un vettore di dimensione $N \times M$, le cui:

- prime N componenti sono la prima riga di A , letta da sinistra a destra;
- seconde N componenti sono la seconda riga di A , letta da sinistra a destra;
- ...
- M -esime N componenti sono la M -esima riga di A , letta da sinistra a destra.

Analogamente si definisce la conversione lessicografica per colonne.

9.3 Regularizzazione di Tikhonov applicata alle immagini

Durante la soluzione di Naive si considerava il problema delle immagini come il seguente problema di minimo:

$$\min_x \|\cdot\|_2^2$$

Tuttavia dato che si ha un errore y^δ è necessario stabilizzare la funzione col metodo di regolarizzazione di Tikhonov:

$$\min_x \|Ax - y^\delta\|_2^2 + \lambda \|x\|_2^2$$

Dove λ è il parametro di Regularizzazione. Applicando le condizioni del primo ordine $\nabla(f) = 0$ si ha:

$$(A^T A + \lambda I)x = A^T y^\delta$$

È possibile quindi risolvere il problema tramite il metodo CGLS

9.3.1 Regularizzazione con Variazione totale

Una funzione di regolarizzazione alternativa a quella di Tikhonov e molto utilizzata nell'imaging è la funzione di Variazione Totale (TV) definita in questo modo:

$$TV(x) = \|\nabla(x)\|_1 = \sum_{i=1}^M \sum_{j=1}^n \sqrt{(x_{i+1,j} - x_{i,j})^2 + (x_{i,j+1} - x_{i,j})^2}$$

La funzione TV non è differenziabile nel punto $(0,0)$. Per ottenere la differenziabilità, si inserisce un piccolo parametro $\beta > 0$:

$$TV^\beta(x) = \sum_{i=1}^m \sum_{j=1}^n \sqrt{(x_{i+1,j} - x_{i,j})^2 + (x_{i,j+1} - x_{i,j})^2 + \beta^2}$$

Dove i valori di solito utilizzati per β sono nell'ordine di 10^{-3}

Il problema di regolarizzazione diventa quindi:

$$\min_x \|Ax - y^\delta\|_2^2 + \lambda TV^\beta(x)$$

Il metodo di regolarizzazione con TV ha il vantaggio di essere particolarmente efficace nell'eliminare il rumore e allo stesso tempo meglio preservare i contorni, anche in caso di basso rapporto segnale/rumore. Il suo uso è motivato dalla sua abilità nel recuperare le discontinuità nell'immagine, inoltre preserva i bordi dell'immagine rimuovendo piccoli dettagli come il rumore. All'aumentare del parametro di regolarizzazione l'immagine tende a una immagine costante a tratti