NUMERO: 1190 DATA: 22/10/2014

A P P U N T I

STUDENTE: Parisi

MATERIA: Calcolo Numerico

Prof. Scuderi

INDICE

Lezione n. 1 — Aritmetica calcolatori ed errori calcolo

Consideriamo il sistema di numerazione decimale o in base 10

Un numero reale "a" si può rappresentare con le cifre che vanno da 0 a 9 ed ogni numero reale si può esprimere come combinazione di potenze in base 10

- $a = 109,34 = 1\times10^{2} + 0\times10^{1} + 3\times10^{0} + 3\times10^{-1} + 4\times10^{-2}$

RAPPRESENTAZIONE FLOATING-POINT:

$\boxed{a = pN^{q}}$ p = numero reale q = intero N = base (per noi sarà sempre 10)

- $a = 0,015\cdot10^{-1} = 0,15\cdot10^{-2} = 0,0015\cdot10^{0}$

Spostando la virgola a destra e a sinistra bisogna modificare anche opportunamente la potenza in base 10 = N

Quindi per un numero reale "a" esistono più rappresentazioni floating-point, per cui la rappresentazione non è unica.

"a" è univocamente determinato se: $N^{-1} \le |p| < 1$ ⇒ in N=10 $0,1 \le |p| < 1$

In questo caso la prima cifra di p dopo il punto decimale deve essere diversa da zero.

Se questa condizione risulta soddisfatta, la rappresentazione floating point si dice normalizzata.

p e q sono quindi unici e li possiamo nominare come: mantissa e esponente o caratteristica rispettivamente.

- $a = 0,15\cdot10^{-2}$ — $p = 0,15$ — $q = -2$
  $a = 12,4\cdot10^{0} = 0,124\cdot10^{2}$ — $p = 0,124$ — $q = 2$
  $a = 0,0013\cdot10^{4} = 0,13\cdot10^{2}$ — $p = 0,13$ — $q = 2$

Ricapitolando: se conosciamo N, la coppia (p,q) individua univocamente il numero reale $a = pN^{q}$ nella rappresentazione normalizzata.

Per memorizzare "a" è sufficiente memorizzare $p^{e}q$ oppure il segno di a, |p| e q.

La memoria di un calcolatore è però limitata e quindi:

- |p| può avere al massimo t cifre
- $m \le q \le M$ con $m < 0$ = minimo esponente rappresentabile,
  $M > 0$ = massimo esponente rappresentabile.

UNDERFLOW e OVERFLOW:

Sia $a = pN^q$ con $N^{-1} \leq |p| < 1$: ← p minuscolo, mantissa normalizzata

- Se $q < m$, il numero non è rappresentabile e si ha un problema di underflow. Il numero reale viene approssimato allo zero e il processo di calcolo prosegue.
- Se $q > M$, il numero non è rappresentabile e si ha un problema di overflow. In questo caso il calcolatore arresta il processo di calcolo e segnala all'utente un errore.

Ci sono dei casi in cui possiamo risolvere il problema dell'overflow.

Supponiamo di avere: $N = 10$ t $-127 \leq q \leq 128$

$$y = \sqrt{x_1 \cdot x_2} \qquad \text{con } x_1 = 10^{100} \quad x_2 = 10^{50}$$

$$x_1 \cdot x_2 = 10^{150} \qquad q > 128 \longrightarrow \text{overflow}$$

Per aggirare il problema di overflow si può:

$$y = \sqrt{x_1 \cdot x_2} = \sqrt{x_1} \cdot \sqrt{x_2}$$

Se invece: $y = \sqrt{x_1^2 + x_2^2}$ con $x_1 = 10^{100}$ $x_2 = 10^{50}$

mantenendo l'espressione così, si verificherebbe overflow. Se invece:

$$y = \sqrt{x_1^2\left(1 + \left(\frac{x_2}{x_1}\right)^2\right)} = x_1\sqrt{1 + \left(\frac{x_2}{x_1}\right)^2}$$

non si ottiene overflow. Si potrebbe però ottenere un fenomeno di underflow se x_2 è un numero piccolo. Se si ottiene underflow il numero verrà approssimato a zero.

SE |p| HA PIÙ DI t CIFRE:

mantissa troncata

Sia $a = pN^q$ con $N^{-1} \leq |p| < 1$ e $m \leq q \leq M$

Per rappresentare questo numero in un sistema floating-point con t cifre, per la mantissa si ricorre ad una delle due tecniche di arrotondamento:

1- tecnica di troncamento: si esclude la parte a destra della t-esima cifra.

- $N = 10$ $t = 2$ $a = 0{,}153 \cdot 10^0$

$\bar{a} = 0{,}15 \cdot 10^0$ ⟶ troncamento.

2- tecnica di arrotondamento: si aggiunge $\frac{1}{2}N^{-t}$ a p e poi si tronca il risultato alla t-esima cifra.

- $N = 10$ $t = 2$ $a = 0{,}153$ $\frac{1}{2}N^{-t} = \frac{1}{2}10^{-2} = 0{,}005$

$0{,}153 + 0{,}005 =$

$0{,}158$

$\bar{a} = 0{,}15$ ← troncamento.

Se utilizzo la tecnica di troncamento: $\bar{p} = \bar{p}_1 \Rightarrow |p - \bar{p}| < N^{-t}$

Se utilizzo la tecnica di arrotondamento: $\bar{p} = \bar{p}_1 \Rightarrow |p - \bar{p}| \leq \frac{1}{2} N^{-t}$

$\bar{p} = \bar{p}_2 \Rightarrow |p - \bar{p}| \leq \frac{1}{2} N^{-t}$

L'errore assoluto non supera N^{-t} e $\frac{1}{2} N^{-t}$ rispettivamente.

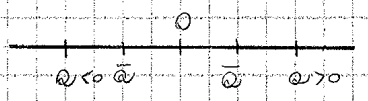
Le tecniche di arrotondamento soddisfano le seguenti disuguaglianze:

$$|a - \bar{a}| = \begin{cases} < N^{q-t} & \text{per troncamento} \\ \leq \frac{1}{2} N^{q-t} & \text{per arrotondamento} \end{cases}$$

$$\frac{|a - \bar{a}|}{|a|} \leq \frac{|a - \bar{a}|}{N^{q-1}} \quad \begin{cases} < N^{1-t} & \text{per troncamento} \\ \leq \frac{1}{2} N^{1-t} & \text{per arrotondamento.} \end{cases}$$

L'errore della tecnica di arrotondamento è la metà di quella di troncamento, quindi la prima è preferibile alla seconda in quanto la seconda può generare un errore maggiore (che può essere il doppio).

Inoltre il segno di $\frac{a - \bar{a}}{a}$ è costante (positivo) assia:



PRECISIONE DI MACCHINA o ERRORE DI ROUNDOFF:

E' la quantità:

$$eps = \begin{cases} N^{1-t} & \text{per troncamento} \quad ① \\ \frac{1}{2} N^{1-t} & \text{per arrotondamento} \quad ② \end{cases}$$

È una costante caratteristica di ogni aritmetica floating-point, dipende dalle cifre rappresentabili e dalla tecnica di arrotondamento ① o ②.

Questa costante rappresenta la massima precisione di calcolo su un calcolatore.

Se "ā" denota il numero di macchina corrispondente al numero reale "a", posto $\varepsilon = \frac{\bar{a} - a}{a}$ possiamo scrivere:

$$\bar{a} = a(1 + \varepsilon) \quad \text{con } |\varepsilon| \leq eps$$

$$\varepsilon = \frac{\bar{a} - a}{a} \longrightarrow a\varepsilon = \bar{a} - a \longrightarrow \bar{a} = a(1 + \varepsilon)$$

$$|\varepsilon| = \frac{|\bar{a} - a|}{|a|} \leq eps$$

Lezione 2 — Aritmetica calcolatori ed errori calcolo

CANCELLAZIONE NUMERICA:

Consiste in una perdita di cifre della mantissa e si verifica quando si esegue un operazione di sottrazione tra due numeri "quasi uguali" ossia con un certo numero di cifre in comune ed arrotondati a numeri di macchina.

- Per $N=10$ $t=5$ e tecnica di arrotondamento.

$a_1 = 0,157824831$

$a_2 = 0,157368212$

In aritmetica esatta: $a_1 - a_2 = 0,000476619 = 0,476619 \cdot 10^{-3}$

In aritmetica finita: $\bar{a}_1 = 0,15782$ $\bar{a}_2 = 0,15735$

$\bar{a}_1 \ominus \bar{a}_2 = 0,00047 = 0,47000 \cdot 10^{-3}$

Se i due operandi iniziali non fossero stati affetti da errore, il risultato sarebbe stato accurato. Gli errori iniziali amplificano l'errore dell'operazione finale.

Consideriamo: $a_1 = p_1 N^q$ e $a_2 = p_2 N^q$

Supponiamo che: $\bar{a}_1 = \bar{p}_1 N^q$ e $\bar{a}_2 = \bar{p}_2 N^q$ numeri di macchina.

I numeri reali sono arrotondati e quindi $p_1 \neq \bar{p}_1$ e/o $p_2 \neq \bar{p}_2$

Supponiamo che $\bar{p}_1$ e $\bar{p}_2$ abbiano t cifre e che le prime s cifre coincidano tra loro, allora:

$$\bar{a}_1 \ominus \bar{a}_2 = (\bar{p}_1 \ominus \bar{p}_2) N^q = (\bar{p}_1 - \bar{p}_2) N^q = \bar{p} N^{q-s}$$

ove solamente le prime $t-s$ cifre in $\bar{p}$ provengono dalle mantisse p_1 e p_2, le restanti s (poste uguali a zero) non hanno alcun significato.

E' quindi poco raccomandabile eseguire sottrazioni tra numeri quasi uguali perché il risultato sarà poco preciso, se i numeri sono stati arrotondati.

Talvolta manipolando le espressioni matematiche che definiscono un problema è possibile evitare il fenomeno della cancellazione numerica.

Quando questo non è possibile si dice che la cancellazione è insita nel problema.

PROBLEMA NUMERICO:

Si intende una descrizione chiara e non ambigua di una connessione funzionale

f: x (input) $\xrightarrow{f}$ y (output)

La connessione fra x e y può essere esplicita ($y = f(x)$) oppure implicita ($f(x,y)=0$)

Quando immetto i dati di input (x) nel calcolatore essi vengono seguiti da un errore e quindi sono rappresentati con $\bar{x}$

$x \xrightarrow{\infty} f(x)$ $\qquad \infty$ = precisione infinita.

$\bar{x} \xrightarrow{\infty} f(\bar{x})$

Quindi solo i dati di ingresso vengono perturbati poiché i calcoli vengono eseguiti in entrambi i casi con precisione infinita senza commettere quindi errori e senza approssimare o arrotondare i calcoli.

Se accade che: $x \approx \bar{x}$ e $f(x) \approx f(\bar{x})$ allora il problema è ben condizionato. Ossia a piccole perturbazioni relative o assolute nei dati, corrispondono piccole perturbazioni nel risultato.

Se accade che: $x \approx \bar{x}$ e $f(x) \not\approx f(\bar{x})$, a piccole variazioni nei dati corrispondono grandi variazioni nel risultato, allora il problema si dice mal condizionato.

Quindi se: $\dfrac{|f(x) - f(\bar{x})|}{|f(x)|} \approx \dfrac{|x - \bar{x}|}{|x|}$, $x, f(x) \neq 0$, ossia gli errori associati a $\bar{x}$ e $f(\bar{x})$ hanno lo stesso ordine di grandezza, allora il problema si dice ben condizionato. Altrimenti mal condizionato.

NUMERO DI CONDIZIONAMENTO:

Per studiare il condizionamento di un problema occorre determinare relazioni del tipo:

$$\frac{|f(x) - f(\bar{x})|}{|f(x)|} \approx K(f,x)\frac{|x - \bar{x}|}{|x|}$$

$$\frac{|f(x) - f(\bar{x})|}{|f(x)|} \leq K(f,x)\frac{|x - \bar{x}|}{|x|}$$

K = numero di condizionamento.

Se $K \approx 1$ allora il problema è ben condizionato e viceversa.

RICHIAMI SU VETTORI E MATRICI:

Sia $x = (x_1, \dots x_n)^T$ (vettore colonna trasposto per questioni di spazio). Le norme di vettore più usate sono:

- $\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2} = \sqrt{x^T \cdot x}$ → NORMA EUCLIDEA — $\| \ \|$ = norma
- $\|x\|_\infty = \max_{1 \le i \le n} |x_i|$ ← valore assoluto → NORMA INFINITA
- $\|x\|_1 = \sum_{i=1}^{n} |x_i|$ ← valore assoluto → NORMA UNO

Le norme di matrice più usate sono:

- $\|A\|_2 = \sqrt{\rho(A^T A)}$ → NORMA SPETTRALE

con $\rho(B) = \max_i |\lambda_i|$, con λ_i autovalore di B, è detto raggio spettrale di B
(autovalore di massimo modulo)

- $\|A\|_\infty = \max_{1 \le i \le n} \sum_{j=1}^{n} |a_{ij}|$ ← valore assoluto

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & & & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}$$

Si sommano gli elementi in valore assoluto di ciascuna riga. Dopodiché il massimo tra i risultati ottenuti rappresenta la norma infinito ($|A|_\infty$).

- $\|A\|_1 = \max_{1 \le j \le n} \sum_{i=1}^{n} |a_{ij}|$ ← valore assoluto

Si sommano gli elementi di ciascuna colonna in valore assoluto. Dopodiché il massimo tra i risultati ottenuti rappresenta la norma 1 ($|A|_1$).

Per le norme 1, 2 e ∞ valgono le seguenti proprietà:

- $\|AB\| \le \|A\| \|B\|$
- $\|I\| = 1$ con I = matrice identità
- $\|Ax\| \le \|A\| \|x\|$

Una matrice si dice <u>triangolare superiore</u> quando gli elementi sotto la diagonale principale sono uguali a zero.
Si dice <u>triangolare inferiore</u> quando gli elementi sopra la diagonale principale sono uguali a zero.
Una matrice si dice <u>diagonale</u> quando gli elementi sopra e sotto la diagonale principale sono uguali a zero.

RISOLUZIONE NUMERICA DI SISTEMI LINEARI:

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

in forma matriciale:

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

in forma compatta: $Ax = b$

Assumeremo che questo sistema lineare abbia una e una sola soluzione.

Cosa succede se perturbiamo (di poco) i dati? cerchiamo col esaminare il condizionamento del sistema:

Consideriamo i dati A e b e i dati perturbati $\bar{A} = A + \delta A$ e $\bar{b} = b + \delta b$.

$\bar{x} = x + \delta x$ è la soluzione in aritmetica esatta del sistema perturbato $\bar{A}\bar{x} = \bar{b}$

Per esaminare il condizionamento confrontiamo:

- l'errore relativo associato alla soluzione $\longrightarrow \|\delta x\| / \|x\|$ con
- l'errore relativo associato ai dati $\longrightarrow \|\delta A\|/\|A\| \quad \|\delta b\|/\|b\|$

Se $A + \delta A$ è non singolare, quindi il sistema assume una sola soluzione, e $\|\delta A\|$ (perturbazione in A) risulti $< 1/\|A^{-1}\|$ (quindi piccola), si dimostra che:

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{K(A)}{1 + K(A)\frac{\|\delta A\|}{\|A\|}} \left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|b\|} \right)$$

dove $K(A) = \|A\|\|A^{-1}\| \geq \|A \cdot A^{-1}\| = \|I\| = 1$

Supponga inoltre: $\|\delta A\| < \frac{1}{2\|A^{-1}\|}$ si ha che:

$$\frac{\|\delta x\|}{\|x\|} \leq 2K(A) \left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|b\|} \right)$$

In questo caso se $2K(A)$ è piccolo, a piccole perturbazioni sui dati, corrisponde una piccola perturbazione sul risultato, ed il problema è ben condizionato.

$K(A)$ viene pertanto definito come numero di condizionamento.

Per le norme 1, 2 e ∞ abbiamo:

$$K(A) = \|A\|\|A^{-1}\| \geq \|AA^{-1}\| = \|I\| = 1$$

Lezione 3 — Metodi numerici per risoluzione sistemi lineari.

I metodi numerici per risolvere sistemi lineari sono essenzialmente di due tipi:
- diretti
- iterativi

METODI DIRETTI:

① Tecnica di sostituzione all'indietro per sistemi triangolari superiori.

Dato il sistema triangolare superiore:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ \vdots \\ a_{n-1,n-1}x_{n-1} + a_{n-1}x_n = b_{n-1} \\ a_{nn}x_n = b_n \end{cases}$$

con $a_{ii} \neq 0$ $\forall i$ questa ipotesi garantisce che $\det \neq 0 \rightarrow$ 1 soluzione

Risoluzione: sostituisco x_n nella penultima eq. e trovo x_{n-1} e così via.

$$\begin{cases} x_n = \dfrac{b_n}{a_{nn}} \\ x_{n-1} = \dfrac{b_{n-1} - a_{n-1,n}x_n}{a_{n-1,n-1}} \\ \vdots \\ x_1 = \dfrac{b_1 - \sum_{j=2}^{n} a_{1j}x_j}{a_{11}} \end{cases}$$

$$\begin{cases} x_n = \dfrac{b_n}{a_{nn}} \\ x_i = \dfrac{b_i - \sum_{j=i+1}^{n} a_{ij}x_j}{a_{ii}} \end{cases} \quad \text{con } i = n-1, \dots, 1$$

Costo computazionale: $\frac{n^2}{2}$ (in termini di operazioni aritmetiche $+, \cdot$)

Comandi matlab:

```
A = [...;...;...];
b = [.;.;.];
n = lenght(b);
x = zeros(n,1);
x(n) = b(n) / A(n,n);
for i = n-1 : -1 : 1
    s = 0;
    for j = i+1 : n
        s = s + A(i,j) * x(j)
    end
    x(i) = (b(i) - s) / A(i,i);
end
```

si sostituisce perché in questo modo si sfrutta delle funzioni di matlab e quindi è più veloce

Si può sostituire con: $s = A(i, i+1:n) * x(i+1:n)$

i da $n-1$ a 1 con passo -1

j da $i+1$ a n con passo $+1$

$$\sum_{j=i+1}^{n} a_{ij}x_j = a_{i,i+1}x_{i+1} + \dots + a_{in}x_n = \underbrace{(a_{i,i+1}, \dots, a_{in})}_{A(i, i+1:n)} \underbrace{(x_{i+1}, \dots, x_n)^T}_{x(i+1:n)}$$

③ Metodo dell'eliminazione di Gauss: Guardare (da Wiki) differenza Cramer Gauss

Ci consente di risolvere sistemi la cui matrice dei coefficienti è una matrice generica ossia non diagonale o triangolare ecc.

Dato il sistema lineare: $Ax = b$ di ordine n con 1 soluzione

Risoluzione: $Ax = b \overset{n-1}{\Longleftrightarrow} Ux = \overline{b}$

U: triangolare superiore

Quindi questo metodo trasforma iniziolmente il sistema assegnato in uno equivalente, ossia con la stessa soluzione, ma il sistema equivalente è associato ad una matrice triangolare superiore U. Una volta ricondotti al sistema equivalente possiamo utilizzare la tecnica ① e ricavare l'incognita x.
↳ sostituzione all'indietro

Trasformazioni:
- la soluzione rimane invariata se si scambiano tra loro due eq. del sistema.
- la soluzione rimane invariata se si sostituisce ad un eq. del sistema una combinazione lineare dell'eq. stessa con un altra.

Esempio: $n = 4$

$$Ax = b \Longleftrightarrow \begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 = b_3 \\ a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 = b_4 \end{cases}$$

- si elimina l'incognita x_1 dalla 2°, 3°, 4° eq.
- si elimina l'incognita x_2 dalla 3°, 4° eq.
- si elimina l'incognita x_3 dalla 4° eq..

3 passi $= n - 1 = 4 - 1 = 3$

Passo $K = 1$: si pone $a_{1j}^{(1)} := a_{1j}$ e $b_1^{(1)} := b_1$. Supponiamo $a_{1j}^{(1)} \neq 0$ altrimenti si scambiano la prima eq. con l'eq. k-esima tale che $a_{k1}^{(1)} \neq 0$.

$$a_{2j}^{(2)} = a_{2j}^{(1)} + \left(-\frac{a_{21}^{(1)}}{a_{11}^{(1)}}\right) a_{1j}^{(1)} \qquad b_2^{(2)} = b_2^{(1)} + \left(-\frac{a_{21}^{(1)}}{a_{11}^{(1)}}\right) b_1^{(1)}$$

Allo stesso modo elimino x_1 dalla 3° e 4° equazione

Proseguo allo stesso modo con il passo $K = 2$ e $K = 3$ ottenendo alla fine di $K = 3$ la matrice semplificata U.

Si risolve il sistema $Ux = \overline{b}$ con la tecnica di sostituzione all'indietro ①.
↳ Upper (matrice triangolare superiore)

Comandi Matlab:

```
A = [---;---;---];
b = [-;-;-];
n = lenght(b);
for K = 1:n-1
```

Si può sostituire con:

$A(K+1:n) = -A(K+1:n,K)/A(K,K)$

```
    for i = K+1:n
        A(i,K) = -A(i,K)/A(K,K);
    end
```

Si può sostituire con:

$A(i,K+1:n) = A(i,K+1:n) + A(i,K) \cdot A(K,i+1:n)$

```
    for j = K+1:n
        A(i,j) = A(i,j) + A(i,K)*A(K,j);
    end
    b(i) = b(i) + A(i,K)*b(K);
end
```

- Consideriamo il sistema di ordine $n = 18$:

$$Ax = b \qquad a_{ij} = \cos((j-1)\theta_i) \qquad \theta_i = \frac{2i-1}{2n}\pi$$

$$b_i = \sum_{j=1}^{n} a_{ij}$$

la cui soluzione esatta è: $x = (1, \dots, 1)^T$

Il condizionamento in norma infinito è pari a: $K_\infty(A) \approx 17$ (sistema lineare ben condizionato).

$\frac{\|x - \bar{x}\|_\infty}{\|x\|_\infty} \approx 10^{-3}$ ← errore relativo con $\bar{x}$ calcolato mediante il metodo di eliminazione di Gauss senza Pivoting.

$\frac{\|x - \bar{x}\|_\infty}{\|x\|_\infty} \approx 10^{-15}$ ← errore relativo con $\bar{x}$ calcolato mediante il metodo di eliminazione di Gauss con Pivoting parziale

Questo esempio mostra come l'errore relativo sia molto più grande se non si utilizza il pivoting parziale.

FATTORIZZAZIONE MATRICI:

- $P_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ scambio $2 \longleftrightarrow 3$ righe

$\longrightarrow P_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$ moltiplico per una matrice A

$$\longrightarrow P_2 A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{31} & a_{32} & a_{33} \\ a_{21} & a_{22} & a_{23} \end{pmatrix}$$

Quindi se voglio realizzare uno scambio tra le righe o le equazioni k e r del sistema lineare basta che io moltiplichi a sinistra per un opportuna matrice ottenuta scambiando le righe k e r dalla matrice identità.

$$P_k A^{(k)} x = P_k b^{(k)}$$

dove $k \longleftrightarrow r$

$$P_k = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{matrix} \\ k \\ \\ r \\ \\ \end{matrix}$$

Al passo k la sostituzione delle eq. $i = k+1, \dots, n$ con le eq. che si ottengono rispettivamente sommando alle eq. stesse la k-esima eq. moltiplicata per m_{ik}, si può realizzare nel seguente modo:

$$M_k P_k A^{(k)} x = M_k P_k b^{(k)}$$

Lezione 5 — Applicazioni fattorizzazione $PA=LU$, Metodi iterativi

- Calcolo del determinante di A: calcolo prima $PA = LU$ e poi

$$\det(A) = (-1)^s \prod_{i=1}^{n} u_{ii}$$

$$U = \begin{pmatrix} u_{11} & & \\ 0 & \ddots & \\ 0 & 0 & u_{nn} \end{pmatrix}$$

s = numero totale degli scambi di eq. effettuati

- Calcolo dell'inversa di A: $A^{-1} = ?$

$PA = LU \longrightarrow (PA)^{-1} = (LU)^{-1} \longrightarrow A^{-1}P^{-1} = U^{-1}L^{-1} \longrightarrow$

$A^{-1}P^{-1}P = U^{-1}L^{-1}P \longrightarrow A^{-1} = U^{-1}L^{-1}P$

Costo computazionale: n^3 (minimo costo per l'inversione)

- Risoluzione di p sistemi lineari aventi tutti la stessa matrice dei coefficienti:

$Ax_1 = b_1 \qquad Ax_2 = b_2 \qquad Ax_3 = b_3 \qquad p = 3$

risolvendo tramite eliminazione di Gauss: (con comandi Matlab)

$x_1 = A \backslash b_1$ con costo computazionale $n^3/3$

$x_2 = A \backslash b_2$ con costo computazionale $n^3/3$

$x_3 = A \backslash b_3$ con costo computazionale $n^3/3$

Costo computazionale totale: n^3

Possiamo però cercare di risolvere, con minore costo computazionale, questo problema:

determiniamo $PA = LU$

$Ax_i = b_i \longrightarrow \underbrace{PA}_{LU} x_i = Pb_i \longrightarrow LU x_i = Pb_i \longrightarrow$

$$\begin{cases} Ly_i = Pb_i \\ Ux_i = y_i \end{cases} \quad \text{per } i = 1,2,3$$

Costo computazionale: $\frac{n^3}{3}$ della fattorizzazione $PA = LU$ $+$

n^2 per risoluzione di ciascun sistema $=$

$$\frac{n^3}{3} + 3n^2$$

$3n^2$ è trascurabile per n sufficientemente grande e quindi il costo computazionale è: $\frac{n^3}{3}$

A SIMMETRICA DEFINITA POSITIVA: FATTORIZZAZIONE CHOLESKI

In questo caso il pivoting è superfluo poiché il metodo ③ è già stabile. Il pivoting non migliora ulteriormente la accuratezza della soluzione.
Quindi in questo caso sussiste questa decomposizione:

$$A = LU$$

È possibile riscrivere: $LU = L_1 \cdot L_1^T$
con L_1 matrice triangolare inferiore con elementi positivi sulla diagonale principale. Questa fattorizzazione è detta <u>fattorizzazione di Cholesky</u>.
$A = L_1 L_1^T$ si ottiene con costo computazionale $\frac{n^3}{6}$ ovvero la metà del costo computazionale del metodo di Gauss.

<u>Comandi Matlab:</u>

`R = chol(A)` calcola il fattore $R = L_1^T$ triangolare superiore della fattorizzazione di Choleski $A = R^T R = L_1 L_1^T$, della matrice simmetrica definita positiva A.

- Risoluzione del sistema lineare $Ax = b$:
  A definita simmetrica positiva.
  R = chol(A) — R = triangolare sup — R^T = triang. inf.
  $A = R^T \cdot R$
  $R^T \cdot \underbrace{R \cdot x}_{y} = b$ → $\begin{cases} R^T y = b \Rightarrow y \\ Rx = y \Rightarrow x \end{cases}$

  Costo computazionale: - se R è nota: n^2
  - altrimenti: $\mathcal{O}\left(\frac{n^3}{6} + n^2\right) = \mathcal{O}\left(\frac{n^3}{6}\right)$

- Calcolo inversa di A:
  A simmetrica definita positiva.
  $A = R^T \cdot R$ → $(A)^{-1} = (R^T R)^{-1}$ → $A^{-1} = R^{-1} \cdot (R^T)^{-1} = R^{-1}(R^{-1})^T$
  Nel primo caso avrei dovuto calcolare l'inversa di due matrici, mentre così solo di una, a guadagno di un minor costo computazionale.
  Costo computazionale: $\frac{2n^3}{3}$

Quindi: $x^{(0)} = \begin{pmatrix} x_1^{(0)} \\ \vdots \\ x_n^{(0)} \end{pmatrix} \xRightarrow{\delta} x^{(1)} = \begin{pmatrix} x_1^{(1)} \\ \vdots \\ x_n^{(1)} \end{pmatrix} \xRightarrow{\delta} x^{(2)} = \begin{pmatrix} x_1^{(2)} \\ \vdots \\ x_n^{(2)} \end{pmatrix}$ ecc...

② Metodo di Gauss - Seidel:

$$x_i^{(K+1)} = \frac{b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(K+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(K)}}{a_{ii}} \quad \text{con } K = 0, 1, \dots$$

In questo modo si ottiene una soluzione del problema più accurata.

- $\begin{cases} 3x_1 - x_2 = 2 \\ x_1 + 2x_2 = 3 \end{cases}$ $\quad \begin{cases} x_1 = \frac{2 + x_2}{3} \\ x_2 = \frac{3 - x_1}{2} \end{cases}$ $\quad x = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

- Metodo di Jacobi:

$$\begin{cases} x_1^{(K+1)} = \frac{2 + x_2^{(K)}}{3} \\ x_2^{(K+1)} = \frac{3 - x_1^{(K)}}{2} \end{cases}$$

	K=0	K=1	K=2	K=3	K=4	
$x^{(0)}$	$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$		
0	2/3	7/6	13/18	---		
0	3/2	7/6	11/12	---		

$$\frac{\|x - x^{(3)}\|_\infty}{\|x\|_\infty} \approx 0{,}8 \cdot 10^{-1}$$

Per avere un errore con una tolleranza specifica allora vado avanti con l'iterazione finché non la raggiungo.

- Metodo di Gauss - Seidel:

$$\begin{cases} x_1^{(K+1)} = \frac{2 + x_2^{(K)}}{3} \\ x_2^{(K+1)} = \frac{3 - x_1^{(K+1)}}{2} \end{cases}$$

	K=0	K=1	K=2	K=3	K=4	
$x^{(0)}$	$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$		
0	2/3	13/18	107/108	---		
0	7/6	35/36	217/216	---		

$$\frac{\|x - x^{(3)}\|_\infty}{\|x\|_\infty} \approx 0{,}8 \cdot 10^{-2}$$

L'errore con questo secondo metodo è minore.

CRITERI DI ARRESTO ITERAZIONI:

Quando si implementa un procedimento iterativo occorre definire dei criteri di arresto. Fissare un numero massimo (K_{max}) di iterazioni ne rappresenta uno.
Un altro criterio d'arresto, che riguarda la bontà di approssimazione $x^{(k)}$, consiste nel fissare una tolleranza relativa (tol_r ($\geq eps$)) o assoluta (tol_a) e nell'or- restare il processo all'iterazione $\bar{k}$ se $x^{(\bar{k})}$ soddisfa la seguente disuguaglianza:

$$\| x^{(\bar{k}+1)} - x^{(\bar{k})} \| \leq tol_r \| x^{(\bar{k}+1)} \|$$

oppure

$$\| x^{(\bar{k}+1)} - x^{(\bar{k})} \| \leq tol_a$$

COMANDI MATLAB: METODO JACOBI

```
A = [---; ---; ---];
b = [-; -; -];
Kmax = -;
toll = -;
n = length(b);
x0 = zeros(n,1);
D = diag(diag(A));
C = A - D;
for K = 1:Kmax
    x1 = D\(b - C*x0)
    if abs(x1 - x0) <= toll;
        break
    end
    x0 = x1;
end
```

METODO GAUSS - SEIDEL:

```
A = [---; ---; ---];
b = [-; -; -];
Kmax = -;
toll = -;
n = length(b);
x0 = zeros(n,1);
D = tril(A);
C = A - D;
for K = 1:Kmax
    x1 = D\(b - C*x0)
    if abs(x1 - x0) <= toll;
        break
    end
    x0 = x1;
end
```

CONVERGENZA DEI METODI DI JACOBI E GAUSS-SEIDEL:

- Si dimostra che se A è a diagonale dominante per righe allora: $\|I - D^{-1}A\|_\infty < 1$
  Se A è a diagonale dominante per colonne: $\|I - D^{-1}A\|_1 < 1$.
  Pertanto se A è a diagonale dominante per righe o colonne, allora i metodi di Jacobi e di Gauss-Seidel applicati al sistema lineare $Ax = b$, convergono.
- Si dimostra che se A è simmetrica definita positiva, allora il metodo di Gauss-Seidel converge.

Osserviamo inoltre che la convergenza del metodo di Gauss-Seidel non implica quella del metodo di Jacobi e viceversa.
Se entrambi convergono, in generale Gauss-Seidel converge più rapidamente.
La rapidità di convergenza dipende dal raggio spettrale: tanto più esso è piccolo tanto più rapidamente converge.

PROPRIETÀ' METODI ITERATIVI: $Ax = b$

- x viene determinata come limite di una successione di vettori convergente;
- x in aritmetica con precisione infinita di calcolo viene determinata in maniera approssimata; in aritmetica con precisione finita di calcolo può essere determinata con una precisione soddisfacente le richieste dell'utente;
- non modificano la matrice dei coefficienti A;
- sono efficienti per matrici sparse (con la maggior parte degli elementi uguali a zero) e di grandi dimensioni.

- <u>somme esponenziali di ordine n:</u>

$$\tilde{f}(x) = e_n(x) = \sum_{k=0}^{n} a_k e^{kx} \qquad \tilde{f}(x) = e_n(x) = \sum_{k=0}^{n} a_k e^{-kx}$$

E' utilizzato per l'approssimazione di funzioni con comportamento di tipo esponenziale male approssimabile con dei polinomi.

- <u>funzioni spline di ordine n:</u>

Sono funzioni polinomiali a tratti dotate di derivata continua fino all'ordine n-1.
Sono utilizzate per approssimare funzioni che generalmente non sono ben approssimabili da polinomi poiché per raggiungere un certo grado di approssimazione richiedono un grado elevato del polinomio e i polinomi di grado elevato oscillano fortemente.

Osserviamo che ciascun elemento dei suddetti sottospazi si esprime come combinazione lineare di funzioni, che definiscono una base per il sottospazio in questione, mediante i parametri a_k oppure a_k, b_k.

- le funzioni che definiscono una base per il sottospazio lineare dei polinomi algebrici di grado n, sono:

$$\varphi_0(x) = 1, \quad \varphi_1(x) = x, \quad \dots \quad \varphi_n(x) = x^n$$

Ogni polinomio si esprime quindi come combinazione di questi monomi.
Il suddetto sottospazio ha dunque dimensione finita n+1.

<u>INDIVIDUARE UNA PARTICOLARE FUNZIONE APPROSSIMANTE:</u>

Si adotterà uno dei seguenti criteri:

- <u>criterio dell'interpolazione:</u>
  E' utilizzato per l'approssimazione di una funzione.
- <u>criterio dei minimi quadrati:</u>
  E' utilizzato per l'approssimazione di un insieme di dati sperimentali.

Ora come funzione f_n fissiamo un polinomio e utilizziamo il criterio dell'interpolazione per determinare il polinomio f_n di grado minimo che passa per i n punti (x_i, y_i). Per determinare il polinomio di grado minimo a grado $n-1$:

$$p_{n-1}(x) = c_1 x^{n-1} + c_2 x^{n-2} + \dots + c_{n-1} x + c_n$$

definito dagli n coefficienti $c_1, \dots, c_n$ generici che vogliamo determinare in modo tale che:

$$p_{n-1}(x_i) = y_i \qquad i = 1, \dots, n$$

Si dimostra che se $x_i \neq x_j$ per $i \neq j$ allora esiste uno ed un solo polinomio soddisfacente le condizioni di interpolazione.

Imponiamo il passaggio per i punti x_i:

$$\begin{cases} c_1 x_1^{n-1} + \dots + c_{n-1} x_1 + c_n = y_1 \\ c_1 x_2^{n-1} + \dots + c_{n-1} x_2 + c_n = y_2 \\ \vdots \\ c_1 x_n^{n-1} + \dots + c_{n-1} x_n + c_n = y_n \end{cases}$$

Si ottiene così un sistema lineare di n equazioni e $c_1, \dots c_n$ incognite.

In forma matriciale:

$$\begin{pmatrix} x_1^{n-1} & \dots & x_1 & 1 \\ x_2^{n-1} & \dots & x_2 & 1 \\ \vdots & & \vdots & \vdots \\ x_n^{n-1} & \dots & x_n & 1 \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \iff Vc = y$$

La matrice dei coefficienti è la matrice di Van der monde, che è notoriamente mal condizionata quindi la soluzione non è accurata in quanto a piccole variazioni nei dati corrispondono non piccole variazioni nel risultato.

In caso $x_i \neq x_j \Rightarrow \det(V) = \prod_{i>j} (x_i - x_j) \neq 0 \Rightarrow$ la matrice V è non singolare. $\Rightarrow$ il sistema lineare ammette una sola soluzione:

$$\exists!\ c = (c_1 \dots c_n)^T : Vc = y.$$

Quindi esiste un solo polinomio interpolante i dati assegnati: $\exists!\ p_{n-1}$ passante per i punti (x_i, y_i).

- Calcoliamo i coefficienti del polinomio interpolante la funzione $f(x) = 1/(1+x^2)$ in 13 punti equispaziati nell'intervallo $[0,1]$

```
n = 13;
x = linspace(0,1,n);
f = inline('1./(1+x.^2)');   serve per definire l'espressione di una funzione.
y = f(x);
c = polyfit(x,y,n-1);
Warning: Polynomial is badly conditioned...
```

RAPPRESENTAZIONE DI LAGRANGE DEL POLINOMIO DI INTERPOLAZIONE:

Prima di definire questa rappresentazione dobbiamo definire dei particolari polinomi interpolanti:

si considerino i dati: $(x_1,0),\dots,(x_{j-1},0),(x_j,1),(x_{j+1},0),\dots,(x_n,0)$

supponiamo: $x_i \neq x_j \quad i \neq j$

Notiamo come le ordinate siano sempre 0 tranne quando ha ascissa x_j:

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

$l_1 \longrightarrow (x_1,1)\,(x_2,0)\,\dots\,(x_n,0)$

$l_2 \longrightarrow (x_1,0)\,(x_2,1)\,\dots\,(x_n,0)$ ecc.

Si può dimostrare che:

$$l_j(x_j) = \frac{(x_j-x_1)\dots(x_j-x_{j-1})(x_j-x_{j+1})\dots(x_j-x_n)}{(x_j-x_1)\dots(x_j-x_{j-1})(x_j-x_{j+1})\dots(x_j-x_n)} = 1 \qquad \text{per } j = 1,\dots,n$$

Questi polinomi $l_j(x)$ sono detti polinomi fondamentali di Lagrange.

A questo punto il polinomio di interpolazione lo si può riscrivere:

$$p_{n-1}(x) = \sum_{j=1}^{n} y_j\, l_j(x)$$

$$p_{n-1}(x_i) = \sum_{j=1}^{n} y_j\, l_j(x_i) = \sum_{j=1}^{n} y_j\, \delta_{ij} = y_i$$

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 41 di 82

Lezione 8 — Rappresentazione di Newton del polinomio di interpolazione.

RAPPRESENTAZIONE DI NEWTON DEL POLINOMIO DI INTERPOLAZIONE:

Dati n punti distinti, definiamo le seguenti quantità:

$$f[x_1, x_2] = \frac{f(x_2) - f(x_1)}{x_2 - x_1}$$ = differenza divisa

differenza divisa di ordine 1 di $f(x)$,

$$f[x_1, x_2, x_3] = \frac{f[x_2, x_3] - f[x_1, x_2]}{x_3 - x_1}$$

differenza divisa di ordine 2 di $f(x)$, e più in generale:

$$f[x_1, \dots x_n] = \frac{f[x_2, \dots, x_n] - f[x_1, \dots x_{n-1}]}{x_n - x_1}$$

differenza divisa di ordine $n-1$.

Le differenze divise sono invarianti rispetto a permutazioni degli argomenti cioè:
- $f[x_1, x_2] = f[x_2, x_1]$
- $f[x_1, x_2, x_3] = f[x_2, x_1, x_3] = f[x_3, x_2, x_1]$ ... ecc

Si definisce rappresentazione di Newton del polinomio interpolante i punti (x_i, y_i) con $y_i = f(x_i)$, la seguente espressione:

$$p_{n-1}(x) = f(x_1) + f[x_1, x_2](x - x_1) + f[x_1, x_2, x_3](x - x_1)(x - x_2) \pm \dots +$$
$$+ f[x_1, x_2, \dots, x_n](x - x_1) \dots (x - x_{n-1})$$

I polinomi $1, x - x_1, \dots, (x - x_1) \dots (x - x_{n-1})$ sono detti polinomi fondamentali di Newton.

Verifichiamo che però $p_{n-1}(x)$ verifichi la proprietà di interpolazione $p_{n-1}(x_i) = y_i$ per ogni $i = 1, \dots, n$.

Verifichiamo che: $p_{n-1}(x_1) = y_1 = f(x_1)$

$$p_{n-1}(x_1) = f(x_1) + f[x_1, x_2](x_1 - x_1) + f[x_1, x_2, x_3](x_1 - x_1)(x_1 - x_2) + \dots + \dots$$

$$p_{n-1}(x_1) = f(x_1) \quad \checkmark$$

Verifichiamo che: $p_{n-1}(x_2) = f(x_2)$

$$p_{n-1}(x_2) = f(x_1) + f[x_1, x_2](x_2 - x_1) + f[x_1, x_2, x_3](x_2 - x_1)(x_2 - x_2) + \dots + \dots =$$

- Scriviamo la rappresentazione di Newton del polinomio $p_3(x)$ interpolante i punti $(0,1), (1,-1), (2,1), (1/2, 2)$

x	y	$i=1$	$i=2$	$i=3$
0	1			
		$\frac{-1-1}{1-0} = -2$		
1	-1			
		$\frac{1-(-1)}{2-1} = 2$	$\frac{2-(-2)}{2-0} = 2$	
2	1			$\frac{\frac{16}{3}-2}{\frac{1}{2}-0} = \frac{20}{3}$
		$\frac{2-1}{\frac{1}{2}-2} = -\frac{2}{3}$	$\frac{-\frac{2}{3}-2}{\frac{1}{2}-1} = \frac{16}{3}$	
$\frac{1}{2}$	2			

$$p_3(x) = 1 + (-2)(x-0) + 2(x-0)(x-1) + \frac{20}{3}(x-0)(x-1)(x-2)$$

Comandi Matlab: no

```
function a = difdiv(x,y)
n = length(x);
for i = 1:n-1
    for j = n:-1:i+1
    y(j) = (y(j)-y(j-1))/(x(j)-x(j-i));
    end
end
a = y;
```

Descriviamo ora l'algoritmo che consente di valutare in un punto oppure in un vettore di punti il polinomio interpolante. A tal scopo consideriamo la rappresentazione di Newton del polinomio di grado 3 interpolante i punti $(x_1, f(x_1))$ $(x_2, f(x_2))$ $(x_3, f(x_3))$ $(x_4, f(x_4))$ e, indicando con a_i le differenze divise in esso presenti, scriviamo:

$$p_3(x) = a_1 + a_2(x-x_1) + a_3(x-x_1)(x-x_2) + a_4(x-x_1)(x-x_2)(x-x_3)$$

a_i = differenze divise.

	·	+
$p_3(z) = a_1$		
$p_3(z) = a_1 + a_2(z-x_1)$	1	2
$p_3(z) = p_3(z) + a_3(z-x_1)(z-x_2)$	2	2
$p_3(z) = p_3(z) + a_4(z-x_1)(z-x_2)(z-x_3)$	2	2

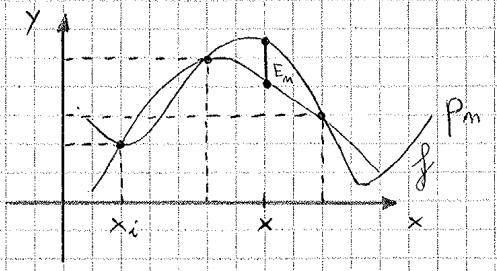
In questo modo abbiamo ad ogni passaggio quasi sempre 2 moltiplicazioni e 2 addizioni.

CONVERGENZA DEL POLINOMIO DI INTERPOLAZIONE:

Denotato con p_n il polinomio interpolante la funzione f nei punti $x_1, \dots x_{n+1}$, si definisce <u>errore di interpolazione</u> la funzione:

$$E_n(x) = f(x) - p_n(x)$$

Ci chiediamo se p_n è una buona approssimazione della funzione f e quanto piccolo è l'errore di interpolazione E_n



Osserviamo che $E_n(x) = 0$ per $x = x_i$ avendosi $E_n(x_i) = f(x_i) - p_n(x_i) = 0$ e $E_n(x) \equiv 0$ per ogni $x \in [a,b]$ e per $f = p$ con p polinomio di grado n avendosi, per unicità del polinomio interpolante, $p_n \equiv p$.

Per quale x_i e f : $\lim_{n \to \infty} E_n(x) = 0$?

La risposta a questa domanda non è sempre affermativa; infatti si dimostra che, fissata una matrice di interpolazione ovvero una matrice di nodi:

$$\begin{matrix} x_1 & & & \\ x_1 & x_2 & & \\ x_1 & x_2 & x_3 & \\ x_1 & x_2 & x_3 & x_4 \\ \dots & & & \end{matrix}$$

con $x_i \in [a,b]$ e $x_i \neq x_j$, $i \neq j$, è sempre possibile determinare una funzione $f \in [a,b]$ per la quale si abbia:

$$\lim_{n \to \infty} \| E_n \|_\infty \neq 0$$

Non è quindi detto che: $p_n(x) \longrightarrow f(x)$ per $n \to +\infty$ $\forall f$

Se i nodi di interpolazione sono scelti coincidenti con gli zeri dei polinomi di Chebichev, $z_i = -\cos\left(\left((2i-1)/(2n)\right)\pi\right) \in (-1,1)$, $i = 1, \dots, n$ la condizione $f \in C^1[-1,1]$ garantisce che:

$$\lim_{n\to\infty} \|E_n\|_\infty = 0 \quad \leftarrow \text{convergenza.}$$

- Se i nodi x_i sono scelti coincidenti con gli zeri dei polinomi di Chebichev traslati in $[-s, s]$, cioè' $x_i = \frac{b-a}{2} z_i + \frac{b+a}{2}$, $a = -s$ e $b = s$ e $f(x) = \frac{1}{1+x^2}$ (funzione di Runge), allora $\lim_{n\to\infty} \|E_n(x)\|_\infty = 0$

  $p_n \to f$ per $n \to \infty$

La convergenza dipende quindi da:
- scelta dei nodi di interpolazione — peggior caso: equispaziati — migliore: Chebichev
- regolarità funzione f

- $S_3(x_i) = y_i \quad i = 0, \dots, n$
  Sono le condizioni di interpolazione.

4n $\{a_i, b_i, c_i, d_i\} \quad i = 1, \dots, n$

$$\lim_{x \to x_i^-} [a_i + b_i x + c_i x^2 + d_i x^3] = \lim_{x \to x_i^+} a_{i+1} + b_{i+1} x + c_{i+1} x^2 + d_{i+1} x^3$$

$$a_i + b_i x_i + c_i x_i^2 + d_i x_i^3 = a_{i+1} + b_{i+1} x_i + c_{i+1} x_i^2 + d_{i+1} x_i^3 \qquad n-1 \text{ eq.}$$

Calcolo derivata prima: $b_i + 2c_i x_i + 3d_i x_i^2 = b_{i+1} + 2c_{i+1} x_i + 3d_{i+1} x_i^2 \qquad n-1$ eq.

Calcolo derivata seconda: $2c_i + 6d_i x_i = 2c_{i+1} + 6d_{i+1} x_i \qquad n-1$ eq.

Imponendo la terza condizione abbiamo: $n+1$ equazioni

In definitiva abbiamo: $4n-2$ eq. in $4n$ incognite

Ho quindi bisogno di altre due eq. per risolvere le $4n$ incognite $\{a_i, b_i, c_i, d_i\}$

Pertanto per definire una spline cubica occorre imporre altre due ulteriori condizioni.

Fra tutte le possibili condizioni aggiuntive, le tipiche sono:

- $S_3^{(2)}(x_0) = 0 \qquad S_3^{(2)}(x_n) = 0 \longrightarrow$ <u>spline cubiche naturali</u>
- $S_3^{(3)}(x_1^-) = S_3^{(3)}(x_1^+) \qquad S_3^{(3)}(x_{n-1}^-) = S_3^{(3)}(x_{n-1}^+) \longrightarrow$ <u>condizione not-a-Knot</u>
- $S_3^{(1)}(x_0) = f'(x_0) \qquad S_3^{(1)}(x_n) = f'(x_n)$

Imponendo una delle suddette condizioni, la spline corrispondente esiste ed è unica.

Per determinare la spline cubica S_3 soddisfacente le condizioni suddette e una delle condizioni aggiuntive, non si procede andando a risolvere il sistemone di $4n$ eq in $4n$ incognite, ma si procede assumendo come incognite non i parametri a_i, b_i, c_i, d_i ma le quantità $M_i = S_3^{(2)}(x_i)$, $i = 0, \dots, n$. In questo modo risolveremo un sistema lineare di ordine $n+1$ anziché $4n$ che gode di proprietà particolari se si impone una delle condizioni aggiuntive. Il sistema è simmetrico, a diagonale dominante e tridiagonale e si può risolvere con il metodo dell'eliminazione di Gauss.

Lezione 10 — Criterio dei minimi quadrati

Per individuare una unica funzione approssimante vi sono due criteri:
- l'interpolazione
- dei minimi quadrati

Abbiamo già notato che l'interpolazione polinomiale non garantisce al crescere del grado del polinomio una maggiore accuratezza nell'approssimazione di una funzione. Quindi non è detto che all'aumentare del numero dei punti di interpolazione, la convergenza sia garantita.

Questo può essere superato con l'interpolazione mediante funzioni polinomiali a tratti. Essa tuttavia mal si presta ad essere utilizzata per estrapolare informazioni dai dati noti, ovvero per generare valutazioni in punti che giacciono al di fuori dell'intervallo di interpolazione.
In quest'ultimo caso conviene utilizzare il <u>criterio dei minimi quadrati</u>.

Tale criterio viene adottato anche quando si vuole approssimare un insieme di dati (x_i, y_i), $i = 1, \dots, m$, con una funzione approssimante f_n che dipende da un numero $n \ll m$ di parametri.
Consideriamo, come funzione approssimante f_n, una combinazione lineare (a coefficienti costanti) delle funzioni di base $\varphi_k(x)$, $k = 1, \dots, n$:

$$f_n(x) = c_1 \varphi_1(x) + c_2 \varphi_2(x) + \dots + c_n \varphi_n(x)$$

Il criterio dei minimi quadrati consiste allora nel determinare i coefficienti c_k del modello, imponendo che:

$$\varepsilon^2(c_1, \dots, c_n) = \sum_{i=1}^{m} [y_i - f_n(x_i)]^2 = \sum_{i=1}^{m} \left[y_i - \sum_{k=1}^{n} c_k \varphi_k(x_i) \right]^2 = \text{volore minimo}$$

ε^2 è denominato come <u>residuo</u> e noi vogliamo, imponendo che questo residuo risulti minimo, che la funzione passi il più vicino possibile ai punti.

Questa funzione f_n che minimizza il residuo prende il nome di migliore approssimazione dei dati secondo il criterio dei minimi quadrati.

Osserviamo che per $n = m - 1$, la migliore approssimazione f_n dei dati (x_i, y_i), $i = 1, \dots, m$, coincide con la funzione interpolante i punti (x_i, y_i).
Infatti per essa si ha $\varepsilon^2 \equiv 0$.

Questa matrice $B = A^T \cdot A$, si dimostra una matrice simmetrica e semidefinita positiva, cioè $x^T B x \geq 0 \quad \forall x \neq 0$ (vettore nullo).
B è definita positiva se e solo se le colonne di A sono linearmente indipendenti.

Pertanto, se le colonne di A sono linearmente indipendenti, il sistema delle eq. normali ammette una ed una sola soluzione e questa rende minima ϵ^2.
In tal caso, per la risoluzione del sistema $Bc = d$ si potrebbe utilizzare la decomposizione di Choleski $B = L_1 L_1^T$, ma risulta più efficiente procedere altrimenti verso la decomposizione $A = Q \cdot R$

Se le colonne di A sono invece linearmente dipendenti, il sistema ammette infinite soluzioni; si dimostra che solo una di queste ha norma euclidea minima.
Pertanto in questo caso assumiamo come soluzione del nostro problema ai minimi quadrati, il vettore c^* così definito:

$$\sum_{i=1}^{m} \left[y_i - \left(c_1^* \varphi_1(x_i) + c_2^* \varphi_2(x_i) + \dots + c_n^* \varphi_n(x_i) \right) \right]^2 = \text{minimo}$$

$$\| c^* \|_2^2 = \text{minimo}$$

- Assegnati i dati (x_i, y_i), $i = 1, \dots, m$, determiniamo il polinomio di grado 1, $p_1(x) = c_1 x + c_2$, che meglio approssima i suddetti dati secondo il criterio dei minimi quadrati.
  I coefficienti c_1 e c_2 si ottengono come soluzione del sistema delle eq. normali:

$$\begin{pmatrix} \sum_{i=1}^{m} x_i^2 & \sum_{i=1}^{m} x_i \\ \sum_{i=1}^{m} x_i & m \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{m} y_i x_i \\ \sum_{i=1}^{m} y_i \end{pmatrix}$$

e definiscono la cosiddetta retta di regressione lineare.

$$g_m = p_1 = c_1 x + c_2$$

$$\begin{pmatrix} \sum_{i=1}^{m} \varphi_1(x_i) \varphi_1(x_i) & \sum_{i=1}^{m} \varphi_2(x_i) \varphi_1(x_i) \\ \sum_{i=1}^{m} \varphi_1(x_i) \varphi_2(x_i) & \sum_{i=1}^{m} \varphi_2(x_i) \varphi_2(x_i) \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{m} y_i \varphi_1(x_i) \\ \sum_{i=1}^{m} y_i \varphi_2(x_i) \end{pmatrix}$$

- Data la funzione: $S(x) = \begin{cases} 3 + x - 3x^3 & 0 \le x \le 1 \\ a + b(x-1) + c(x-1)^2 + d(x-1)^3 & 1 \le x \le 2 \end{cases}$

- Determinare a, b, c in modo che $S(x)$ sia una spline cubica in $[0,2]$.
- Determinare d, se esiste, che rende $S(x)$ una spline cubica naturale.

- $S_3(1^-) = S_3(1^+)$ $\quad 3 + 1 - 3 = a \quad \Rightarrow \quad a = -5$

$S_3^{(1)}(1^-) = S_3^{(1)}(1^+)$ $\quad 1 - 27x^2\big|_{x=1} = b + 2c(x-1) + 3d(x-1)^2\big|_{x=1}$

$1 - 27 = b \quad \Rightarrow \quad b = -26$

$S_3^{(2)}(1^-) = S_3^{(2)}(1^+)$ $\quad -54x\big|_{x=1} = 2c + 6d(x-1)\big|_{x=1}$

$-54 = 2c \quad \Rightarrow \quad c = -27$

$S(x) \in C^2$

- $S''(0) = 0 \quad S''(2) = 0$. Se sono soddisfatte queste due condizioni la funzione è una spline cubica naturale.

$S''_{[0,1]} = 54x \quad \Rightarrow \quad S''(0) = 0$

$S''_{[1,2]} = 2c + 6d(x-1) \quad \Rightarrow \quad 2c + 6d = 0 \quad \Rightarrow \quad -54 + 6d = 0$

$\Rightarrow \quad d = 9$

$S''(2) = 0$ per $d = 9$

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 57 di 82

Per stimare la rapidità di convergenza di un metodo, si introduce la seguente definizione:

si definisce ordine di convergenza di una successione $\{x_n\}$ convergente ad α, il numero reale $p \geq 1$ tale che:

$$\lim_{n \to \infty} \frac{|\alpha - x_{n+1}|}{|\alpha - x_n|^p} = C \neq 0 \neq +\infty$$

α rappresenta il limite della successione x_n.
Se questa condizione di limite è verificata, allora la successione $\{x_n\}$ ha ordine di successione p

- Se $p = 1 \Rightarrow C < 1 \Rightarrow$ convergenza lineare
- Se $1 < p < 2 \Rightarrow$ convergenza superlineare
- Se $p = 2 \Rightarrow$ convergenza quadratica
- Se $p = 3 \Rightarrow$ convergenza cubica.

Se la successione $\{x_n\}$ è definita da un metodo iterativo e ha ordine p, allora si dice che il metodo iterativo ha ordine p di convergenza.

Supponiamo che un metodo abbia ordine di convergenza p allora vale la definizione di limite vista precedentemente, ossia per n sufficientemente grande

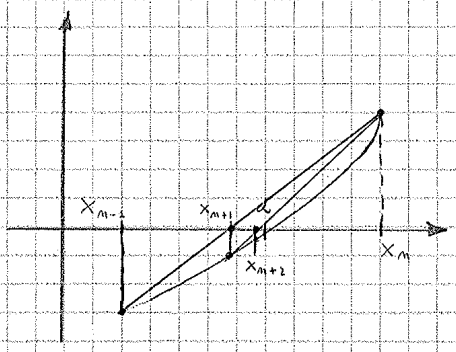
$$\lim_n \frac{|\alpha - x_{n+1}|}{|\alpha - x_n|^p} = C \Rightarrow |\alpha - x_{n+1}| \approx C |\alpha - x_n|^p \Rightarrow$$

$$\Rightarrow \text{se } |\alpha - x_n| \leq 10^{-k} \text{ allora } |\alpha - x_{n+1}| \approx C \, 10^{-kp}$$

Dunque per n sufficientemente grande se x_n ha k decimali corretti, il numero di decimali corretti presenti in x_{n+1} è dell'ordine di kp.

METODO DELLE SECANTI:

Graficamente può essere rappresentato così:



Nel metodo delle secanti si parte da due valori a e b che chiamerò con x_0 e x_1 e si definisce x_{n+1} con $n = 1, 2, \dots$ come l'intersezione con l'asse delle ascisse della retta passante per i punti $(x_{n-1}, f(x_{n-1}))$ e $(x_n, f(x_n))$:

$$\begin{cases} y = f(x_n) + \dfrac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}(x - x_n) \\ y = 0 \end{cases}$$

$$x_{n+1} \equiv x = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} \qquad n = 1, 2 \dots$$

Se la funzione $f(x)$ è tale per cui tutte le successive approssimazioni appartengono all'intervallo di partenza $[a, b]$ e se α è una radice semplice di $f(x) = 0$, si dimostra che se $f \in C^2([a, b])$ e $|\alpha - x_0|$, $|\alpha - x_1|$ sono sufficientemente piccoli, allora:

$$\lim_{n \to \infty} x_n = \alpha$$

e la convergenza è garantita.
L'ordine di convergenza del metodo è $p = \dfrac{1 + \sqrt{5}}{2} \approx 1{,}618$ ed è una convergenza di ordine superlineare $p = 2$

Comandi Matlab:

```
f = inline('...');
x0 = ...;                x1 = ...;
nmax = ...;              toll = ...;
for n = 1:nmax
    x2 = x1 - f(x1)*(x1 - x0)/(f(x1) - f(x0));
    if abs(x2 - x1) <= toll
        break
    end
    x0 = x1;             x1 = x2;
end                      x2
```

Comandi Matlab:

```
f = inline('___');
fd = inline('___');
x0 = _;
nmax = _;
toll = _;
for n = 1: nmax
    x1 = x0 - f(x0) / fd(x0);
    if abs(x1 - x0) <= toll;
        break
    end
    x0 = x1;
end
x1
```

Teorema:

Sia $I := [a, b]$. Se:

① $g(x) \in I$ per ogni $x \in I$ $(g(I) \subseteq I)$

② $g \in C^1(I)$

③ $|g'(x)| \leq m < 1 \quad \forall x \in I$

Se ①, ② e ③ sono soddisfatte allora esiste uno e uno solo punto fisso α di g in I e per $x_{n+1} = g(x_n)$ si ha:

$$\lim_{n \to +\infty} x_n = \alpha \qquad \forall x_0 \in I$$

quindi la convergenza di x_n in α.

Inoltre, se $\exists I_\alpha : |g'(x)| > 1 \quad \forall x \in I_\alpha$, allora:

$$\lim_{n \to +\infty} x_n \neq \alpha$$

quindi x_n non converge ad α.

- Determiniamo la radice $\sqrt{2} \approx 1{,}4142135623$ dell'equazione $f(x) = x^4 - 4 = 0$. Consideriamo:

  i) $g(x) = x + f(x) \quad \Rightarrow \quad x_{n+1} = x_n + (x_n^4 - 4)$

  ii) $g(x) = x - \dfrac{f(x)}{11} \quad \Rightarrow \quad x_{n+1} = x_n - \dfrac{(x_n^4 - 4)}{11}$ con $K = -11$

  A partire da $x_0 = 1$, si ha:

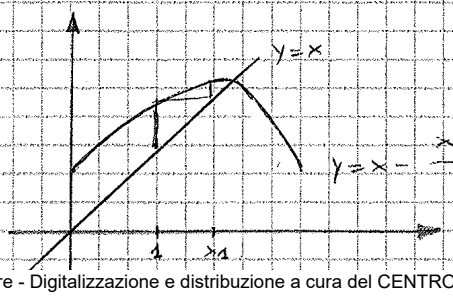
n	i	ii
1	-2	$1{,}272727\ldots$
2	10	$1{,}337830\ldots$
3	10^4	$1{,}414330\ldots$
4	10^{16}	$1{,}414208\ldots$
5	10^{64}	$1{,}414213\ldots \approx \sqrt{2}$

Il diverso comportamento dei metodi i e ii dipende dal fatto che risulta:

i) $g(x) = x + (x^4 - 4) \quad \Rightarrow \quad |g'(x)| = |1 + 4x^3| > 1$ per ogni $x > 0$

ii) $g(x) = x - \dfrac{(x^4 - 4)}{11} \quad \Rightarrow \quad |g'(x)| = \left|1 - \dfrac{4x^3}{11}\right| < 1$ per ogni $x \in \left(0, \sqrt[3]{\dfrac{11}{2}}\right)$.

Pertanto, posto $I = \left(0, \sqrt[3]{\dfrac{11}{2}}\right)$ il teorema precedente garantisce la convergenza a partire da qualsiasi $x_0 \in I$. Graficamente si ha:



$$f(x) = e^x - 2x^2$$

$$x_{n+1} = x_n - \frac{e^{x_n} - 2x_n^2}{e^{x_n} - 4x_n}$$ da applicare a partire da $x_0 = 0$

$x_0 = 0$, $x_1 = -1$, $x_2 = -0{,}6263\ldots$, $x_3 = -0{,}5041\ldots \approx \alpha_1$

- Proporre un metodo iterativo convergente del tipo $x_{n+1} = g(x_n)$ per approssimare la radice negativa dell'eq. $e^x - 2x^2 = 0$.
Calcolare le prime tre iterate a partire da $x_0 = 0$.

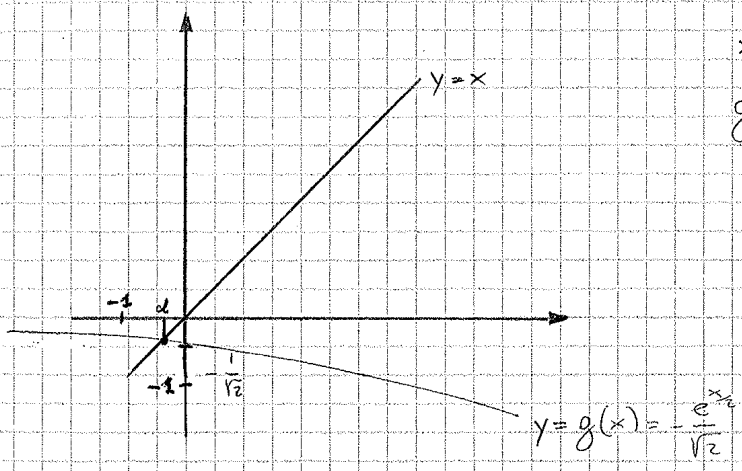
$f(x) = e^x - 2x^2$ $\qquad f(x) \equiv 0$

Usando il metodo del punto fisso:

$2x^2 = e^x \quad \Rightarrow \quad x = \pm\sqrt{\frac{e^x}{2}}$

$g(x) = -\sqrt{\frac{e^x}{2}} = -\frac{e^{x/2}}{\sqrt{2}}$

$x_{n+1} = g(x_n) \quad \rightarrow \quad x_{n+1} = -\frac{e^{x_n/2}}{\sqrt{2}}$



$x = g(x)$

$g(x) = -\frac{e^{+x/2}}{\sqrt{2}}$

$I = (-\infty, 0]$

i) $g(I) \subseteq I \qquad g(I) = \left[-\frac{1}{\sqrt{2}}, 0\right) \subseteq (-\infty, 0)$ ✓

ii) $g(I) \in C^1(I) \qquad g = -\frac{e^{x/2}}{\sqrt{2}}$ ✓

iii) $|g'(x)| \leq m < 1 \quad \forall x \in I$

$|g'(x)| = \left|-\frac{e^{x/2}}{2\sqrt{2}}\right| = \frac{e^{x/2}}{2\sqrt{2}} \leq \frac{1}{2\sqrt{2}} < 1 \qquad m = \frac{1}{2\sqrt{2}}$

$\frac{x}{2} < 0 \qquad \exp\left(\frac{x}{2}\right) \leq \exp(0) = 1$ ✓

Allora il teorema mi dice che:

$\exists!\, \alpha : g(\alpha) = \alpha \qquad x_n : x_{n+1} = g(x_n)$

$x_n \rightarrow \alpha \qquad \forall x_0 \in I$

FORMULE DI QUADRATURA INTERPOLATORIE:

Consideriamo il seguente problema: il calcolo dell'integrale definito su un intervallo chiuso e limitato:

$$\int_a^b f(x)\,dx$$

Ricordiamo che non è sempre possibile calcolare analiticamente un integrale, per esempio $\int_0^1 e^{-x^2}dx$ non è risolvibile mediante il calcolo della funzione primitiva con funzioni elementari.
Quando anche ciò fosse possibile potrebbe risultare complicato calcolarlo, per esempio l'integrale coincide con la somma di una serie oppure è definito mediante funzioni elementari e non, approssimate.
In queste situazioni e quando f è nota solo per punti oppure è valutabile per ogni valore di x mediante un algoritmo, si rende allora necessario l'utilizzo di un metodo numerico in grado di restituire un valore approssimato dell'integrale, indipendentemente dalla complessità della funzione integranda.
Un metodo numerico per il calcolo di un integrale definito assume la seguente espressione:

$$\int_a^b f(x)\,dx \approx \sum_{j=1}^{m} w_j f(x_j)$$

che è definito come <u>formula di quadratura</u>.
I punti x_j sono detti <u>nodi di quadratura</u> e i numeri w_j <u>coefficienti o pesi</u> della <u>formula di quadratura</u>.

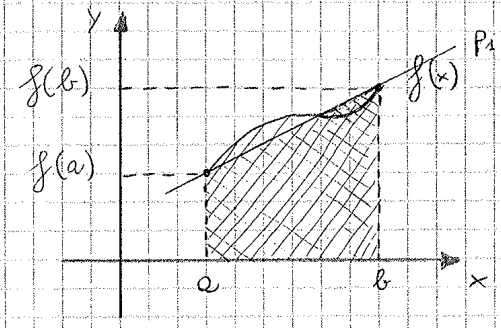
Fissati n nodi distinti $x_1, \dots, x_n$ nell'intervallo $[a,b]$, si definiscono <u>interpolatorie</u> le formule di quadratura che si ottengono approssimando l'integrale assegnato con l'integrale del polinomio $p_{n-1}(f;x)$ che interpola la funzione integranda f nei suddetti nodi, cioè:

$$\int_a^b f(x)\,dx \approx \int_a^b p_{n-1}(f;x)\,dx \qquad f \approx p_{n-1}(f;x)$$

Il vantaggio sta nel fatto che l'integrale di un polinomio lo si può calcolare analiticamente. Quindi:

$$\int_a^b p_{n-1}(f;x)\,dx = \int_a^b \underbrace{\sum_{j=1}^{n} \ell_j(x) f(x_j)}_{\text{rappresentazione di Lagrange del polinomio interpolante}}\,dx =$$

- per $n=2$, $x_1=a$ e $x_2=b$ si ha la <u>formula del trapezio</u>:



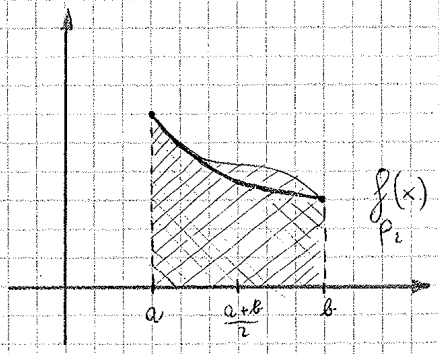
$$\int_a^b f(x)\,dx \approx \int_a^b p_1(f;x)\,dx =$$

$$= \int_a^b \left[\underbrace{\frac{x-b}{a-b}}_{\ell_1(x)} f(a) + \underbrace{\frac{x-a}{b-a}}_{\ell_2(x)} f(b) \right] dx =$$

$$= \frac{b-a}{2}\left[f(a)+f(b)\right]$$

Si approssima l'integrale $f(x)$ con l'area del trapezio di base maggiore più base minore $[f(a)+f(b)]$ per l'altezza $b-a$ diviso 2.

- per $n=3$, $x_1=a$, $x_2=\frac{b+a}{2}$ e $x_3=b$ si ha la <u>formula di Simpson</u>:



$$\int_a^b f(x)\,dx \approx \int_a^b p_2(f;x)\,dx =$$

$$= \int_a^b \left[\frac{\left(x-\frac{a+b}{2}\right)(x-b)}{\left(a-\frac{a+b}{2}\right)(a-b)} f(a) + \frac{(x-a)(x-b)}{\left(\frac{a+b}{2}-a\right)\left(\frac{a+b}{2}-b\right)} f\left(\frac{a+b}{2}\right) + \frac{(x-a)\left(x-\frac{a+b}{2}\right)}{(b-a)\left(b-\frac{a+b}{2}\right)} f(b) \right] dx =$$

$$= \frac{b-a}{2}\left[f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right]$$

Si approssima $f(x)$ con l'arco di parabola passante per a, $(a+b)/2$ e b

Supponiamo che la funzione integranda $f(x)$ sia una funzione integrabile, ma non abbastanza regolare (per esempio $f(x)$ o una delle sue prime derivate presenta delle singolarità o punti di discontinuità in $[a,b]$.)
È evidente che approssimare una funzione irregolare con un polinomio che è una funzione regolare non è vantaggioso.
Supponiamo inoltre che $f(x)$ sia fattorizzabile nella forma:

$$f(x) = w(x)\,g(x)$$

dove $w(x)$ è una funzione di forma semplice contenente le singolarità di $f(x)$, mentre $g(x)$ rappresenta la parte regolare della funzione integranda $f(x)$

Ricordiamo inoltre che, per ogni n, P_n possiede esattamente n zeri, tutti reali, distinti e appartenenti ad (a,b)

Questo vuol dire che questi zeri del polinomio P_n ortogonale in (a,b) rispetto alla funzione peso $w(x)$, possono essere scelti come nodi di quadratura, poiché reali, distinti e appartenenti ad (a,b)

Se io scelgo questi come nodi x_i su cui interpolare e vado a costruire le formule di quadratura pesate interpolatorie associate a questi nodi, vado a costruire delle formule dal nome di <u>formule Gaussiane</u>.

<u>FORMULE GAUSSIANE:</u>

Sono delle formule interpolatorie pesate che utilizzano come nodi x_j gli zeri dei polinomi ortogonali in $[a,b]$ rispetto alla funzione peso $w(x)$.

- per $w(x)=1$, $[a,b]=[-1,1]$ e x_j^{GL} coincidenti con gli zeri del polinomio (di Legendre) di grado n ortogonale in $[-1,1]$ rispetto a $w(x)$, si ha la <u>formula di Gauss-Legendre</u>:

$$\int_{-1}^{1} g(x)\,dx \approx \sum_{j=1}^{n} \overline{w}_j^{GL}\, g(x_j^{GL})$$

$$\text{con } \overline{w}_j^{GL} = \int_{-1}^{1} l_j(x)\,dx \qquad j=1,\dots,n$$

- per $w(x)=\dfrac{1}{\sqrt{1-x^2}}$, $[a,b]=[-1,1]$ e x_j^{GC} coincidenti con gli zeri del polinomio (di Chebicheb) di grado n ortogonale in $[-1,1]$ rispetto a $w(x)$, si ha la <u>formula di Gauss-Chebicheb</u>:

$$\int_{-1}^{1} \frac{1}{\sqrt{1-x^2}}\, g(x)\,dx \approx \sum_{j=1}^{n} \overline{w}_j^{GC}\, g(x_j^{GC})$$

$$\text{con } \overline{w}_j^{GC} = \frac{\pi}{n} \qquad x_j^{GC} = -\cos\left(\frac{2j-1}{2n}\pi\right) \qquad j=1,\dots,n$$

- per $w(x)=(1-x)^{\alpha}(1+x)^{\beta}$, $\alpha,\beta>-1$, $[a,b]=[-1,1]$ e x_j^{GJ} coincidenti con gli zeri del polinomio (di Jacobi) di grado n ortogonale in $[-1,1]$ rispetto a $w(x)$, si ha la <u>formula di Gauss-Jacobi</u>:

$$\int_{-1}^{1} (1-x)^{\alpha}(1+x)^{\beta} g(x)\,dx \approx \sum_{j=1}^{n} \overline{w}_j^{GJ}\, g(x_j^{GJ})$$

$$\overline{w}_j^{GJ} = \int_{-1}^{1} (1-x)^{\alpha}(1+x)^{\beta} l_j(x)\,dx \qquad j=1,\dots,n$$

CONVERGENZA:

Si definisce <u>errore di quadratura</u> della formula:

$$\int_a^b w(x)\, g(x)\, dx \approx \sum_{j=1}^{n} \overline{w}_j\, g(x_j)$$

la quantità:

$$R_n(g) = \int_a^b w(x)\, g(x)\, dx - \sum_{j=1}^{n} \overline{w}_j\, g(x_j)$$

Per le formule interpolatorie pesate:

$$R_n(g) = \int_a^b w(x)\, E_n(g;x)\, dx$$

dove $E_n(g;x) = g(x) - p_{n-1}(g;x)$ definisce l'<u>errore di interpolazione</u>

Osserviamo che fissato n, numero dei nodi di interpolatura, le formule interpolatorie sono esatte, cioè l'errore $R_n \equiv 0$, ogni qualvolta $g(x)$ è un polinomio di grado $n-1$. Infatti, in tal caso, per l'unicità del polinomio interpolante, g e il polinomio $p_{n-1}(g)$ (interpolante g in n nodi distinti) coincidono e, di conseguenza $E_n(g) \equiv 0$.

Pertanto, mediante le formule interpolatorie pesate definite a partire da n nodi distinti, possiamo calcolare esattamente gli integrali del tipo $\int_a^b w(x) g(x)\, dx$, in cui g è un qualsiasi polinomio di grado $n-1$.
Quindi:

$$\int_a^b w(x)\, g(x)\, dx = \sum_{j=1}^{n} \overline{w}_j\, g(x_j) \quad \text{(e non più } \approx\text{)}$$

Si dimostra che le formule di Newton-Cotes a n nodi e chiuse (ovvero quelle che includono tra i loro nodi anche gli estremi di integrazione) sono esatte (cioè $R_n(g) \equiv 0$) ogni qual volta g è un polinomio di grado $n-1$ se n è pari, di grado n se n è dispari.

- la formula del trapezio ($n=2$) è esatta per tutti i polinomi di grado 1.
- la formula di Simpson ($n=3$) è esatta per tutti i polinomi di grado 3

Si dimostra che le formule gaussiane a n nodi sono esatte ogni qualvolta g è un polinomio di grado $2n-1$.

Una formula di quadratura si dice convergente se:

$$\lim_{n\to\infty} R_n(g) = 0$$

Teorema:
Se: - $g \in C[a,b]$ con $\int_a^b w(x)g(x)\,dx \approx \sum_{j=1}^{n} \bar{w}_j g(x_j)$
- $\sum_{j=1}^{n} |\bar{w}_j| \le M$ con M indipendenti da n

allora si può dimostrare: $\lim_{n\to\infty} R_n(g) = 0$

Inoltre se: - $g \in C^k[a,b]$
allora:

$$|R_n(g)| = O\left(\frac{1}{n^k}\right), \quad n \to \infty$$

Questo ci dice che, più è regolare g e più è rapida la convergenza.
(più k è grande e più è rapida la convergenza).

Questo teorema garantisce la convergenza delle formule gaussiane per ogni $g \in C[a,b]$. Infatti si ha:

$$0 < \int_a^b w(x)\,\ell_i^2(x)\,dx = \sum_{j=1}^{n} \bar{w}_j \ell_i^2(x_j) = \bar{w}_i$$

Pertanto, i pesi di una Gaussiana sono tutti positivi e inoltre:

$$\sum_{j=1}^{n} |\bar{w}_j| = \sum_{j=1}^{n} \bar{w}_j = \int_a^b w(x)\,dx = M$$

- Calcoliamo l'integrale: $I = \int_0^1 x^3 e^x\,dx = -2e^1 + 6 \approx 0,5634436\ldots$
mediante formula di Gauss Legendre I_n^{GL} con n nodi di quadratura.
Cambiamento di variabile: $x = \frac{b+a}{2}t + \frac{b+a}{2} = \frac{1}{2}t + \frac{1}{2}$

n	I_n^{GL}	$\lvert I - I_n^{GL}\rvert$
2	0,5655...	1,73e-02
3	0,5632343...	1,41e-04
4	0,5634359...	4,00e-07
5	0,5634363 4...	5,66e-10
6	0,5634363430...	4,76e-13
7	0,5634363430830...	6,66e-16

- Scriviamo la versione comporta della formula del trapezio, detta formula dei trapezi, utilizzando una partizione uniforme (ovvero costante) dell'intervallo di integrazione. A tal scopo poniamo $h = \frac{b-a}{m}$, consideriamo i punti:

$$a \equiv x_1 < x_2 < \dots < x_m < x_{m+1} \equiv b$$

$$x_i = a + (i-1)h \qquad i = 1, \dots, m+1$$

ed approssimiamo, mediante formula del trapezio, l'integrale esteso all'intervallo di integrazione $[x_i, x_{i+1}]$ $i = 1 \dots m$:

$$\int_a^b f(x)\,dx = \sum_{i=1}^{m} \int_{x_i}^{x_{i+1}} f(x)\,dx \approx \sum_{i=1}^{m} \frac{x_{i+1} - x_i}{2} \left[f(x_i) + f(x_{i+1}) \right] =$$

$$= \frac{h}{2} \left[f(x_1) + f(x_2) + f(x_2) + f(x_3) + \dots + f(x_m) + f(x_{m+1}) \right] =$$

$$= \frac{h}{2} \left[f(x_1 + 2 \sum_{i=2}^{m} f(x_i) + f(x_{m+1}) \right]$$

Comandi Matlab:

```
function t = trapezi(f,a,b,m)
%
x = linespace(a,b,m+1);
y = f(x);
t = (b-a)/(2*m)*(y(1) + 2*sum(y(2:m)) + y(m+1));
```

- Calcoliamo l'integrale:

$$I = \int_0^1 x^3 e^x\,dx = -2e^1 + 6 \approx 0,563436343081808 3$$

mediante la formula dei trapezi I_m^T con $m = 2^n + 1$ e $n = 2, \dots, 15$

m	I_m^T	$\lvert I - I_m^T \rvert$
5,0000 e + 00	0,6136008380528142	5,62 e - 02
3,0000 e + 00	0,5775647355773843 3	1,41 e - 02
9,7000 e + 01	0,5663733416442370	3,54 e - 03
3,3000 e + 01	0,5643 2108598739768	8,85 e - 04
6,5000 e + 01	0,5636575502682232	2,21 e - 04
1,2300 e + 02	0,5634316462201426	5,53 e - 05
2,5700 e + 02	0,5634501688503072	1,38 e - 05
5,1300 e + 02	0,5634397395542683	3,46 e - 06
⋮	⋮	⋮
3,2763 e + 04	0,5634363432577 20	8,44 e - 10

Si dimostra che:

- le formule composte convergono per $m \to \infty$, ovvero per $h \to 0$, per $f \in C[a,b]$
- per la formula dei trapezi si ha:

$$R_m(f) = -\frac{(b-a)}{12} h^2 f''(\xi) \qquad \text{con } \xi \in (a,b)$$

e per $f \in C^2[a,b]$; se f è periodica e il periodo coincide con $b-a$ (a è un sottomultiplo di $b-a$), allora l'ordine di convergenza è superiore a $\mathcal{O}(h^2)$. Infatti:

$$|R_m(f)| = \mathcal{O}(h^{2K+1})$$

per $f \in C^{2K+1}[a,b]$ e $f^{(2j-1)}(a) = f^{(2j-1)}(b) \quad j = 1 \dots K$

- per la formula composta di Simpson si ha:

$$|R_m(f)| = \mathcal{O}(h^4) \qquad \text{per } f \in C^4[a,b]$$

Si nota come le formule composte convergono certamente perché f sia continua, però l'ordine di convergenza non dipende dalla regolarità della funzione f, è quindi fisso: $\mathcal{O}(h^2)$ per la formula dei trapezi e $\mathcal{O}(h^4)$ per la formula di Simpson.

Per funzioni poco regolari conviene considerare non più le partizioni uniformi ma quelle di tipo adattativo, per le quali si ha un maggiore addensamento di nodi in prossimità delle singolarità.
Dunque là dove la funzione è poco regolare consideriamo una partizione più densa di nodi e viceversa là dove la funzione è regolare.

Comandi Matlab:

`[q, N] = quad(f, a, b, toll)` Calcola, mediante la formula composta di Simpson ed una partizione di tipo adattativo, il valore approssimato q dell'integrale $\int_a^b f(x)\,dx$ con una tolleranza assoluta toll ed utilizzando N valutazioni di funzione (toll = 10^{-6} di default)