Fondamenti di Calcolo Numerico

Claudio Tessa - 2024/2025

Ci saranno 3 progetti pratici da consegnare. All'esame scritto si puo' prendere massimo 24, i progetti danno in totale al massimo 9 punti aggiuntivi. Si puo' richiedere l'orale.

1 Introduzione

Il **calcolo numerico** è una branca della matematica che si occupa di **trovare soluzioni approssimate a problemi matematici usando metodi numerici**, cioè algoritmi che possono essere implementati su un computer.

In particolare, si parte da un problema reale/fisico x_f e lo modelliamo matematicamente in un problema matematico $P(x_m, d) = 0$, dove x_m e' la **soluzione al problema matematico** e d sono i **dati**. Si passa poi al problema numerico $P(x_n, d) = 0$ **approssimando**, e si risolve con un algoritmo che ci fornira' la soluzione computazionale x_c.

Quando si risolve un problema matematico al computer, bisogna tenere in mente la **precisione macchina**. Si definisce la precisione macchina come il piu' piccolo numero eps tale che $1 + eps \neq 1$ (questo e' possibile in quanto i numeri vengono rappresentati in binario con un numero finito di cifre).

Tutte le operazioni e funzioni complesse nel calcolatore sono ricondotte a **operazioni elementari** (ad esempio, $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$). Ogni risultato quindi, e' soggetto ad **approssimazione**. Lo scopo della matematica numerica e' misurare e controllare tale approssimazione a seconda dell'obiettivo di interesse.

1.1 Problemi matematici

I problemi matematici che verranno affrontati saranno, in ordine,

1. **Sistemi lineari**:
   $A\boldsymbol{x} = \boldsymbol{b}$, con A matrice $n \times n$, $\boldsymbol{x}$ vettore $n \times 1$ e b vettore $n \times 1$,
   $x_m = \boldsymbol{x}, \quad d = \{b_i, a_{ij}, 1 \leq i, j \leq n\}$.
2. **Interpolazione e approssimazione di funzioni e dati**:
   Dato $d = \{(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)\}$, vogliamo trovare il polinomio $p(t)$ che passa per questi dati,
   $p(t) = \sum_{k=0}^{n} a_k t^k, \quad x_n = \{a_0, \dots, a_n\}$.
3. **Integrazione**:
   $x(m) = \int_a^b f(t)dt, \quad d = \{a, b, f\}$.
4. **Radici (zeri) di equazioni non lineari**:
   $f(t) = \sum_{k=0}^{n} a_k t^k, \quad x_m = \{\alpha_1, \dots, \alpha_n\}, \quad d = \{a_0, \dots, a_n\}$.

5. **Equazioni differenziali ordinarie**:
   $\begin{cases} y'(t) = f(t, y(t)) \quad t_0 \leq t \\ y(t_0) = y_0 \end{cases}$
   $x_m = y : (t_0, \infty) \to \mathbb{R}, \quad \{t_0, y_0, f\}.$

1.2 Approssimazione

Vogliamo **approssimare la derivata**

$$y'(t) = f(t, y(t))$$

Discretizziamo l'intervallo di definizione di $y'(t)$ suddividendolo in N_h sottointervalli di larghezza h. Possiamo quindi riscrivere la derivata come

$$y'(t_k) = f(t_k, y(t_k))$$

Definiamo il **rapporto incrementale**

$$y'(t_k) \simeq \frac{y(t_{k+1}) - y(t_k)}{t_{k+1} - t_k}$$

dove $t_{k+1} - t_k = h$.

Chiamiamo u_k un'approssimazione di $y(t_k)$. Allora approssimiamo $y'(t_k)$ con

$$\begin{cases} \dfrac{u_{k+1} - u_k}{h} = f(t_k, u_k) \quad k \geq 0 \\ u_0 = y_0 \end{cases}$$

si ottengono $\{u_0, u_1, \ldots, u_n\}$ soluzioni numeriche.

Dimostreremo che:

- $|u_k| \leq C(t_k)|y_0|$ (**stabilita'**).
- $|u_k - y(t_k)| \to 0 \; \forall k$, per $h \to 0$ (**convergenza**).

2 Metodi diretti per sistemi lineari

Consideriamo il sistema lineare $A\boldsymbol{x} = \boldsymbol{b}$, dove:

- $A \in \mathbb{R}^{n \times n}$ di componenti a_{ij}
- $\boldsymbol{b} \in \mathbb{R}^n$ e' dato
- $\boldsymbol{x} \in \mathbb{R}^n$ e' il vettore delle incognite

Allora la soluzione del sistema esiste ed e' unica se e solo se $\det(A) \neq 0$. Possiamo utilizzare la *formula di Cramer* $x_j = \frac{\det(A_j)}{\det A}$, con $A_j = [\boldsymbol{a}_1 \quad \ldots \quad \boldsymbol{a}_{j-1} \quad \boldsymbol{b} \quad \boldsymbol{a}_{j+1} \quad \ldots \quad \boldsymbol{a}_n]$ e $\boldsymbol{a}_i$ la colonna i-esima di A. Tuttavia questa formula e' **inutilizzabile**, infatti il calcolo di un determinante richiede $n!$ operazioni. Servono quindi dei **metodi numerici** che si traducano in **algoritmi efficienti**.

Partiamo da un caso semplice in cui la matrice e' **triangolare inferiore**. Allora $\ell_{ij} = 0$ per $i < j$. La prima riga di $L\boldsymbol{x} = \boldsymbol{b}$ e' $\ell_{11}x_1 = b_1 \implies x_1 = \frac{b_1}{\ell_{11}}$. La seconda riga e' $\ell_{21}x_1 + \ell_{22}x_2 = b_2 \implies x_2 = \frac{b_2 - \ell_{22}x_1}{\ell_{22}}$. In generale quindi si ha:

$$x_i = \frac{b_i - \sum_{j=1}^{i-1} \ell_{ij}x_j}{\ell_{ii}}$$

Questo e' il **metodo di sostituzioni in avanti**. Notiamo che il numero di operazioni per tale metodo e' n^2.

In maniera analoga si puo' risolvere un sistema $U\boldsymbol{x} = \boldsymbol{b}$, con U matrice **triangolare superiore**, $u_{ij} = 0$ per $i > j$ (in questo caso si parla di **sostituzioni all'indietro**):

$$x_i = \frac{b_i - \sum_{j=i+1}^{n} u_{ij}x_j}{u_{ii}}$$

2.1 Fattorizzazione LU

Definiamo come $A_{p,i} \in \mathbb{R}^{i \times i}$, con $i = 1, \dots, n$, le **sottomatrici principali** di A ottenute considerando le prime i righe e colonne.

Teorema 1

Data $A \in \mathbb{R}^{n \times n}$, se $\det(A_{p,i}) \neq 0 \; \forall i = 1, \dots, n-1$, allora esistono un'unica matrice triangolare inferiore L, con $\ell_{ii} = 1, \; i = 1, \dots, n$, e un'unica matrice triangolare superiore U tali che:

$$A = LU$$

2.1.1 Metodo dell'Eliminazione Gaussiana (MEG)

L'idea è **trasformare la matrice dei coefficienti** A in una **matrice triangolare superiore** U, tramite una serie di operazioni elementari sulle righe (che mantengono il sistema equivalente), facilitando così la risoluzione del sistema tramite **sostituzione all'indietro** (l'eliminazione gaussiana, infatti, e' alla base del calcolo della fattorizzazione LU).

Si introducono:

- Matrici $A^{(k)}$, che rappresentano la matrice dei coefficienti dopo il passo k.
- Vettori $\boldsymbol{b}^{(k)}$, che rappresentano il termine noto dopo il passo k.

Siano $A^{(1)} = A$, e $a_{ij}^{(k)} = (A^{(k)})_{ij}$. Ad ogni passo si modificano A e $\boldsymbol{b}$ fino ad ottenere un sistema $U\boldsymbol{x} = \boldsymbol{y}$, molto piu' facile da risolvere:

$$A^{(1)} = A \to A^{(2)} \to \cdots \to A^{(k)} \to A^{(k+1)} \to \cdots \to A^{(n)} = U$$

$$\boldsymbol{b}^{(1)} = \boldsymbol{b} \to \boldsymbol{b}^{(2)} \to \cdots \to \boldsymbol{b}^{(k)} \to \boldsymbol{b}^{(k+1)} \to \cdots \to \boldsymbol{b}^{(n)} = \boldsymbol{y}$$

L'idea e' quella di annullare gli elementi $a_{ij}^{(k)}$ con:

- $i \geq k$ e $k < k$, per formare gli zeri sotto la diagonale principale, oppure
- $2 \leq i \leq k-1$ e $j < i$

Per fare cio' si sottrae un multiplo della riga k alle successive in modo da annullare gli elementi desiderati. Applicando le stesse operazioni sulle righe si ottiene anche y. Al termine del MEG abbiamo $A^{(n)} = U, \quad \ell_{ij} = L, \quad \boldsymbol{b}^{(n)} = \boldsymbol{y}$.

Algorithm Metodo dell'Eliminazione Gaussiana

---

for $k = 1, \ldots, n-1$ **do**
  for $i = k+1, \ldots, n$ **do**
    $\ell_{ij} = \dfrac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$
    for $j = k+1, \ldots, n$ **do**
      $a_{ij}^{(k+1)} = a_{ij}^{(k)} - \ell_{ik} a_{kj}^{*k}$
    $\boldsymbol{b}_i^{(k+1)} = \boldsymbol{b}_i^{(k)} - \ell_{ik} b_k^{(k)}$

Notiamo che il costo computazionale e' dell'ordine di $\frac{2}{3}n^3$.

La verifica a priori delle ipotesi di applicabilita' del *Teorema 1* e' troppo costosa per essere utile in generale, richiede il calcolo di $n-1$ determinanti. Esistono **condizioni sufficienti** facili da verificare:

1. Matrici a dominanza diagonale stretta per righe o per colonne:

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}|, \;\; i = 1, \ldots, n \qquad\qquad |a_{ii}| > \sum_{j \neq i} |a_{ji}|, \;\; i = 1, \ldots, n$$

2. Matrici simmetriche definite positive:

$$\forall \boldsymbol{z} \in \mathbb{R}^n, \; z \neq 0 \implies \boldsymbol{z}^T A \boldsymbol{z} > 0$$

2.1.2 Fattorizzazione di Cholesky

Per le matrici simmetriche definite positive vale la seguente fattorizzazione di Cholesky (che e' unica):

$$A = R^T R$$

con R triangolare superiore.

Poniamo $r_{11} = \sqrt{a_{11}}$. L'algoritmo e' il seguente.

Algorithm Fattorizzazione di Cholesky

---

for $j = 2, ..., n$ **do**
  for $i = 1, ..., j-1$ **do**

$$r_{ij} = \frac{1}{r_{ii}} \left(a_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj} \right)$$
$$r_{jj} = \sqrt{a_{jj} - \sum_{k=1}^{j-1} r_{kj}^2}$$

In questo caso il costo computazionale e' pari a $\frac{1}{3}n^3$. Il guadagno sostanziale rispetto al MEG e' in termini di memoria, in quanto utilizziamo una sola matrice triangolare.

2.1.3 Pivoting

Se al passo k del MEG si ha $a_{kk}^{(k)} = 0$ (cioe' se $\det(A_{p,k}) = 0$), allora scambio la riga k con la riga $i > k$ tale che $a_{ik}^{(k)} \neq 0$.

Pivoting puo' essere interpretato come una pre-moltiplicazione di A e $\boldsymbol{b}$ per una matrice di permutazione P:

$$PA = LU$$
$$PA\boldsymbol{x} = P\boldsymbol{b} \implies LU\boldsymbol{x} = P\boldsymbol{b} \implies \begin{cases} L\boldsymbol{y} = P\boldsymbol{b} \\ U\boldsymbol{x} = \boldsymbol{y} \end{cases}$$

2.2 Propagazione degli errori di arrotondamento

Il MEG non e' affetto da errore numerico, restituisce la soluzione esatta. Questo pero' e' vero in **aritmetica esatta**, cioe' svolgendo i conti "a mano". Il calcolatore, nel memorizzare una variabile z, introduce un **errore di arrotondamento** δz. Il valore effettivo memorizzato sara' quindi

$$\hat{z} = z + \delta z$$

Questo perche' il calcolatore e' dotato di memoria finito e non puo' quindi memorizzare esattamente i numeri.

In un algoritmo dove un numero enorme di operazioni elementari viene eseguito, gli errori di arrotondamento (seppur piccolissimi) si propagano, **amplificandosi**, rendendo inaccurato l'output.

Nel nostro caso stiamo di fatto risolvendo il sistema lineare perturbato

$$(A + \delta A)\hat{\boldsymbol{x}} = \boldsymbol{b} + \delta\boldsymbol{b}$$

e introducendo un **errore computazionale** $\delta\boldsymbol{x} = \hat{\boldsymbol{x}} - \boldsymbol{x}$.

Vorremmo quindi stimare $\delta\boldsymbol{x}$. Introduciamo prima alcuni elementi fondamentali per poterlo fare:

- **Norma**

$$\|A\|_2 = \sup_{\boldsymbol{v} \in \mathbb{R}^n \setminus \{0\}} \frac{\|A\boldsymbol{v}\|_2}{\|\boldsymbol{v}\|_2}$$

dove

$$\|\boldsymbol{v}\|_2 = \sqrt{\sum_{i=1}^{n} \boldsymbol{v}_i^2}$$

- **Condizionamento di matrice**

$$K(A) = \|A\|_2 \cdot \|A^{-1}\|_2$$

inoltre, se A e' simmetrica definita positiva, allora

$$\|A\|_2 = \lambda_{max}(A) \qquad \|A^{-1}\|_2 = \frac{1}{\lambda_{min}(A)}$$

$$\implies K(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}$$

dove $\lambda_{max}(A)$ e $\lambda_{min}(A)$ rappresentano rispettivamente l'autovalore maggiore e minore di A.

Possiamo ora stimare $\delta\boldsymbol{x}$.

$$(A + \delta A)(\boldsymbol{x} + \delta\boldsymbol{x}) = \boldsymbol{b} + \delta\boldsymbol{b}$$

$$\cancel{A\boldsymbol{x}} + \delta A\boldsymbol{x} + (A + \delta A) = \cancel{\boldsymbol{b}} + \delta\boldsymbol{b}$$

$$\underbrace{(A + \delta A)}_{\text{raccogliendo } A}\, \delta\boldsymbol{x} = \delta\boldsymbol{b} - \delta A\boldsymbol{x}$$

$$A(I + A^{-1}\delta A)\delta\boldsymbol{x} = \delta\boldsymbol{b} - \delta A\boldsymbol{x}$$

moltiplicando per A^{-1}

$$\delta\boldsymbol{x} = (I + A^{-1}\delta A)^{-1} A^{-1}(\delta\boldsymbol{b} - \delta A\boldsymbol{x})$$

$$\|\delta\boldsymbol{x}\|_2 \leq \|(I + A^{-1}\delta A)^{-1}\|_2 \cdot \|A^{-1}\|_2 \cdot (\|\delta\boldsymbol{b}\|_2 + \|\delta A\boldsymbol{x}\|_2)$$

moltiplicando per $\dfrac{\|A\|_2}{\|A\|_2}$

$$\|\delta\boldsymbol{x}\|_2 \leq K(A) \cdot \|(I + A^{-1}\delta A)^{-1}\|_2 \cdot \left(\frac{\|\delta\boldsymbol{b}\|_2}{\|A\|_2} + \frac{\|\delta A\boldsymbol{x}\|_2}{\|A\|_2} \right)$$

notiamo che se $\|A^{-1}\|_2 \cdot \|\delta A\|_2 < 1 \implies I + A^{-1}\delta A$ e' invertibile, allora

$$\|(I + A^{-1}\delta A)^{-1}\|_2 \leq \frac{1}{1 - \|A^{-1}\|_2 \cdot \|\delta A\|_2}$$

$$\implies \|\delta \boldsymbol{x}\|_2 \leq \frac{K(A)}{1-K(A)\frac{\|\delta A\|_2}{\|A\|_2}}\left(\frac{\|\delta \boldsymbol{b}\|_2}{\|A\|_2}+\frac{\|\delta A \boldsymbol{x}\|_2}{\|A\|_2}\right)$$

moltiplichiamo tutto per $\frac{1}{\|\boldsymbol{x}\|_2}$ per trovare l'erroe relativo $\frac{\|\delta \boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2}$. Notiamo inoltre che $\|\delta A \boldsymbol{x}\|_2 \leq \|\delta A\|_2 \cdot \|\boldsymbol{x}\|_2$. Quindi

$$\frac{\|\delta \boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} \leq \frac{K(A)}{1-K(A)\frac{\|\delta A\|_2}{\|A\|_2}}\left(\frac{\|\delta \boldsymbol{b}\|_2}{\|A\|_2 \cdot \|\boldsymbol{x}\|_2}+\frac{\|\delta A\|_2 \cancel{\|\boldsymbol{x}\|_2}}{\|A\|_2 \cdot \cancel{\|\boldsymbol{x}\|_2}}\right)$$

notiamo che $\|\boldsymbol{b}\|_2 \leq \|A\|_2 \cdot \|\boldsymbol{x}\|_2 \implies \frac{1}{\|A\|_2 \cdot \|\boldsymbol{x}\|_2} \leq \frac{1}{\|\boldsymbol{b}\|_2}$ troviamo infine che

$$\frac{\|\delta \boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} \leq \frac{K(A)}{1-K(A)\frac{\|\delta A\|_2}{\|A\|_2}}\left(\frac{\|\delta \boldsymbol{b}\|_2}{\|\boldsymbol{b}\|_2}+\frac{\|\delta A\|_2}{\|A\|_2}\right)$$

Ci accorgiamo che l'errore computazionale dovuto alla propagazione dell'errore di arrotondamento e' quindi grande per **matrici malcondizionate**, cioe' con numero di condizionamento grande.

2.3 Stabilita' della soluzione di un sistema lineare

In generale possiamo introdurre il **residuo** $\boldsymbol{r} = \boldsymbol{b} - A\hat{\boldsymbol{x}} = A(\boldsymbol{x} - \hat{\boldsymbol{x}})$. Essendo una quantita' calcolabile (a differenza dell'errore), ci chiediamo come sia legato all'errore.

$$\|\delta \boldsymbol{x}\|_2 = \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2 = \|A^{-1}\boldsymbol{r}\|_2 \leq \|A^{-1}\|_2 \cdot \|r\|_2$$

ricordando che $\|\boldsymbol{b}\|_2 \leq \|A\|_2 \cdot |\boldsymbol{x}|_2 \implies \frac{1}{\|\boldsymbol{x}\|_2} \leq \|A\|_2 \frac{1}{\|\boldsymbol{b}\|_2}$, otteniamo

$$\frac{\|\delta \boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} \leq K(A)\frac{\|\boldsymbol{r}\|_2}{\|\boldsymbol{b}\|_2}$$

quindi il residuo e' un buono stimatore dell'errore se $K(A)$ e' piccolo.

2.4 Pivoting (II)

L'operazione che maggiormente amplifica l'errore di arrotondamento e' la **sottrazione**. Riprendendo l'algoritmo del MEG, vorremmo che $a_{kk}^{(k)}$ fosse il piu' piccolo possibile in modo da rendere piccolo il sottraendo nella sottrazione. Per questo motivo, **pivoting si fa sempre** (anche quando $a_{kk}^{(k)} \neq 0$), scambiando la riga k con la riga $\bar{r}$:

$$|a_{\bar{r}k}^{(k)}| = \max_{i>k}|a_{ik}^{(k)}|$$

Nel caso piu' generale, il pivoting viene effettuato anche scambiando le colonne nella ricerca del massimo. Cio' equivale ad introdurre un'altra matrice di permutazione Q:

$$PAQ = LU$$

$$A\boldsymbol{x} = \boldsymbol{b} \Leftrightarrow \underbrace{PAQ}_{LU}\underbrace{Q^{-1}\boldsymbol{x}}_{\boldsymbol{x}^*} = P\boldsymbol{b} \implies L\boldsymbol{y} = P\boldsymbol{b} \qquad U\boldsymbol{x}^* = \boldsymbol{y} \qquad \boldsymbol{x} = Q\boldsymbol{x}^*$$

Il pivoting totale riduce al minimo l'errore computazionale. **Non** puo' pero' fare niente per matrici malcondizionate, perche' in generale anche $K(PAQ)$ e' grande.

2.5 Fill-in

Il pattern di una matrice ci permette di identificare facilmente gli elementi diversi da zero. Si ottiene ponendo un punto su ogni posizione con elementi non nulli.

Una matrice e' detta **sparsa** quando il numero di elementi nulli e' molto maggiore di quello degli elementi non nulli.

Il fill-in consiste nel fatto che **i fattori L e U di una matrice sparsa non sono in generale sparsi**. In questo caso la fattorizzazione LU non e' adatta dal punto di vista dell'uccupazione di memoria per matrici sparse.

Per ridurre il fill-in puo' essere utile eseguire una tecnica di **riordinamento**, che consiste nel numerare diversamente le righe di A.

3 Metodi Iterativi per Sistemi Lineari

3.1 Definizione di metodo iterativo

Consideriamo sempre il **sistema lineare** $A\boldsymbol{x} = \boldsymbol{b}$, con $A \in \mathbb{R}^{n \times n}$, $\boldsymbol{b} \in \mathbb{R}^n$, $\boldsymbol{x} \in \mathbb{R}^n$, e $\det(A) \neq 0$.

Ricordiamo che la fattorizzazione LU **non** e' idonea per:

- Sistemi di **grandi dimensioni**, visto il costo di $\sim n^3$ operazioni.
- Sistemi con **matrici sparse** a causa del fenomeno del *fill-in*.

Si introduce una successione $\boldsymbol{x}^{(k)}$ di vettori determinata da una legge ricorsiva che identifica il metodo iterativo.

Al fine di innescare il processo iterativo, e' necessario fornire un punto di partenza $\boldsymbol{x}^{(0)}$:

$$\boldsymbol{x}^{(0)} \to \boldsymbol{x}^{(1)} \to \cdots \to \boldsymbol{x}^{(k)} \to \dots$$

Affinché il metodo abbia senso, deve soddisfare la proprieta' di **convergenza**:

$$\lim_{k \to \infty} \boldsymbol{x}^{(k)} = \boldsymbol{x}$$

La convergenza, inoltre, non deve dipendere dalla scelta di $\boldsymbol{x}^{(0)}$.

Poiche' la convergenza e' garantita solo dopo infinite iterazioni, dal punto di vista pratico dovremo **arrestare il processo iterativo dopo un numero finito di iterazioni**, quando riterremo di essere arrivati "sufficientemente vicini" alla soluzione. Possiamo quindi concludere che, anche in aritmetica esatta, un metodo iterativo (a differenza del MEG) sara' inevitabilmente affetto da un **errore numerico**.

3.2 Metodo di Jacobi

Dopo aver scelto il vettore iniziale $\boldsymbol{x}^{(0)}$:

$$a_{11}x_1^{(1)} = b_1 - a_{12}x_2^{(0)} - a_{13}x_3^{(0)} - \cdots - a_{1n}x_n^{(0)}$$

$$\vdots$$

$$a_{n1}x_n^{(1)} = b_n - a_{n2}x_2^{(0)} - a_{n3}x_3^{(0)} - \cdots - a_{n,n-1}x_{n-1}^{(0)}$$

e, in generale, la soluzione $\forall k > 0,\ \forall i = 1, \ldots, n$ e':

$$x_i^{(k+1)} = \frac{b_i - \sum_{j \neq i} a_{ij} x_j^{(k)}}{a_{ii}}$$

Ogni iterazione costa $\sim n^2$ operazioni, quindi Jacobi e' competitivo con MEG se il numero di iterazioni e' inferiore a n. Per **matrici sparse** invece, il costo e' solo $\sim n$ operazioni per iterazione!

Notare che il metodo di Jacobi e' totalmente **parallelo**, ovvero tutti gli $x_i^{(k+1)}$ possono essere calcolati in modo indipendente gli uni dagli altri, con grandi vantaggi per sistemi di grandi dimensioni

3.3 Metodo di Gauss-Seidel

Partendo dal metodo di Jacobi $x_i^{(k+1)} = \frac{b_i - \sum_{j \neq i} a_{ij} x_j^{(k)}}{a_{ii}}$, prendiamo $j < i$. Al passo $k+1$ conosciamo gia' $x_j^{(k+1)}$ perche' e' gia' stata calcolata. Possiamo quindi pensare di usare, al passo $k+1$, le quantita' gia' calcolate al passo precedente k:

$$a_{11}x_1^{(1)} = b_1 - a_{12}x_2^{(0)} - a_{13}x_3^{(0)} - \cdots - a_{1n}x_n^{(0)} \qquad (\text{come Jacobi})$$

$$a_{22}x_2^{(1)} = b_2 - a_{21}x_1^{(1)} - a_{23}x_3^{(0)} - \cdots - a_{2n}x_n^{(0)}$$

$$\vdots$$

$$a_{nn}x_n^{(1)} = b_n - a_{n1}x_1^{(1)} - a_{n2}x_2^{(1)} - \cdots - a_{n,n-1}x_{n-1}^{(1)}$$

e, in generale, la formula $\forall k > 0,\ \forall i = 1, \ldots, n$ e':

$$x_i^{(k+1)} = \frac{b_i - \sum_{j < i} a_{ij} x_j^{(k+1)} - \sum_{j > i} a_{ij} x_j^{(k)}}{a_{ii}}$$

I costi computazionali sono paragonabili a quelli del metodo di Jacobi. A differenza di Jacobi, con Gauss-Seidel **non** e' possibile calcolare le soluzioni in parallelo.

3.4 Metodi iterativi lineari

In generale consideriamo metodi iterativi lineari della seguente forma:

$$\boldsymbol{x}^{(k+1)} = B\boldsymbol{x}^{(k)} + \boldsymbol{f}$$

dove $B \in \mathbb{R}^{n \times n}$ (**matrice di iterazione**) e $\boldsymbol{f} \in \mathbb{R}^n$ identificano il metodo.

Come bisogna prendere B ed $\boldsymbol{f}$?

1. **Consistenza** - Il metodo se applicato alla soluzione esatta $\boldsymbol{x}$ deve restituire $\boldsymbol{x}$ (ovvero, continuando ad iterare, viene restituito sempre $\boldsymbol{x}$ se siamo arrivati alla soluzione esatta $\boldsymbol{x}$):

   $$\boldsymbol{x} = B\boldsymbol{x} + \boldsymbol{f} \implies \boldsymbol{f} = (I - B)\boldsymbol{x} = (I - B)A^{-1}\boldsymbol{b}$$

   La precedente equazione ci da una relazione tra B ed $\boldsymbol{f}$ in funzione dei dati, e fornisce una condizione necessaria non sufficiente per la convergenza.
2. **Convergenza** - Introduciamo l'errore $\boldsymbol{e}^{(k)} = \boldsymbol{x} - \boldsymbol{x}^{(k)}$ e una norma vettoriale $\|\cdot\|$, ad esempio la norma euclidea. Abbiamo:

   $$\begin{aligned} \|\boldsymbol{e}^{(k+1)}\| &= \|\boldsymbol{x} - \boldsymbol{x}^{(k+1)}\| = \|\boldsymbol{x} - B\boldsymbol{x}^{(k)} - \boldsymbol{f}\| \\ (\text{consistenza}) \ &= \|\boldsymbol{x} - B\boldsymbol{x}^{(k)} - (I - B)\boldsymbol{x}\| \\ &= \|B\boldsymbol{e}^{(k)}\| \leq \|B\| \, \|\boldsymbol{e}^{(k)}\| \end{aligned}$$

   Applicando ricorsivamente questa disuguaglianza, otteniamo

   $$\|\boldsymbol{e}^{(k+1)}\| \leq \|B\| \, \|B\| \, \|\boldsymbol{e}^{(k-1)}\| \leq \|B\| \, \|B\| \, \|B\| \, \|\boldsymbol{e}^{(k-2)}\| \leq \cdots \leq \|B\|^{k+1} \, \|\boldsymbol{e}^{0}\|$$

   Percio':

   $$\lim_{k \to \infty} \|\boldsymbol{e}^{(k+1)}\| \leq \left(\lim_{k \to \infty} \|B\|^{k+1} \right) \|\boldsymbol{e}^{(0)}\|$$

   Notare che $\|B\| < 1 \implies \lim_{k \to \infty} \|\boldsymbol{e}^{(k+1)}\| = 0$. Pertanto $\|B\| < 1$ e' la **condizione sufficiente di convergenza**.

   Introduciamo $\rho(B)$ **raggio spettrale** di B tale che $\rho(B) = \max_j |\lambda_j(B)|$, dove $\lambda_j(B)$ sono gli autovalori di B. Abbiamo che se B e' SDP $\implies \rho(B) = \|B\|_2$.

In generale, valgono le seguenti proprieta'.

> ***Proprieta' 1***
>
> $\rho(B) \leq \|B\|$, per qualunque tipo di norma.

> ***Proprieta' 2*** (condizione necessaria e sufficiente di convergenza)
>
> Il metodo iterativo con matrice di iterazione B converge $\iff \rho(B) < 1$.

3.5 Convergenza dei metodi di Jacobi e Gauss-Seidel

Consideriamo la matrice $A = D - E - F$, dove:

- D e' la diagonale di A
- $-E$ e' la parte triangolare inferiore (escluso la diagonale principale) di A
- $-F$ e' la parte triangolare superiore (escluso la diagonale principale) di A

$$A = \begin{bmatrix} \ddots & & -F \\ & D & \\ -E & & \ddots \end{bmatrix}$$

Il metodo di **Jacobi** puo' essere scritto come

$$D\boldsymbol{x}^{(k+1)} = (E + F)\boldsymbol{x}^{(k)} + \boldsymbol{b}$$

e la matrice di iterazione e' data da

$$B_J = D^{-1}(E + F) = D^{-1}(D - A) = I - D^{-1}A$$

Per il metodo di **Gauss-Seidel** invece si ha

$$(D - E)\boldsymbol{x}^{(k+1)} = F\boldsymbol{x}^{(k)} + \boldsymbol{b}$$

e la matrice di iterazione e' data da

$$B_{GS} = (D - E)^{-1}F$$

Valgono i seguenti risultati di **convergenza**:

- Se A e' **strettamente dominante diagonale per righe**, ovvero

  $$|a_{ii}| > \sum_{j \neq i} |a_{ij}|, \quad i = 1, \ldots, n$$

  allora J e GS convergono.
- Se A e' **simmetrica e definita positiva** (SDP), allora GS converge.
- Se A e' **tridiagonale**, allora J e GS o convergono entrambi, o non convergono entrambi. Se convergono, allora GS e' piu' veloce.

3.6 Il metodo di Richardson stazionario

E' basato sulla seguente legge di aggiornamento, assegnati $\boldsymbol{x}^{(0)}$ e $\alpha \in \mathbb{R}$,

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{k} + \alpha(\boldsymbol{b} - A\boldsymbol{x}^{(k)})$$

dove $\boldsymbol{b} - A\boldsymbol{x}^{(k)}$ e' il residuo $\boldsymbol{r}^{(k)}$ al passo k.

L'idea e' di aggiornare la soluzione numerica aggiungendo una quantita' proporzionale al residuo. Infatti, ci si aspetta che se il residuo e' grande bisogna correggere di molto la

soluzione al passo k, e viceversa.

Dalla definizione otteniamo

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha A\boldsymbol{x}^{(k)} + \alpha\boldsymbol{b} = \underbrace{(I - \alpha A)}_{B_\alpha}\boldsymbol{x}^{(k)} + \underbrace{\alpha\boldsymbol{b}}_{\boldsymbol{f}}$$

segue che il metodo di Richardson stazionario e' caratterizzato dalla matrice di iterazione $B_\alpha = I - \alpha A$ e da $\boldsymbol{f} = \alpha\boldsymbol{b}$.

E' un metodo consistente, infatti $\boldsymbol{x} = \boldsymbol{x} + \cancel{\alpha(\boldsymbol{b} - A\boldsymbol{x})}$.

3.6.1 Stabilita' e convergenza del metodo di Richardson stazionario

Concentriamoci sul caso di matrice A simmetrica e definita positiva. Introduciamo gli autovalori di A che sono reali e positivi:

$$0 < \lambda_{min}(A) = \lambda_1(A) \leq \lambda_2(A) \leq \cdots \leq \lambda_n(A) = \lambda_{max}(A)$$

Abbiamo che il metodo di Richardson stazionario con matrice A simmetrica e definita positiva e' convergente se e solo se

$$0 < \alpha < \frac{2}{\lambda_{max}(A)}$$

3.6.2 Scelta del parametro ottimale

Ci chiediamo ora quale sia il valore del parametro α, fra quelli che garantiscono la convergenza, che **massimizzi la velocita' di convergenza**. Ovvero, vogliamo cercare $0 < \alpha_{opt} < \frac{2}{\lambda_{max}(A)}$ tale che $\rho(B_\alpha)$ sia minimo:

$$\alpha_{opt} = \underset{0<\alpha<2/\lambda_{max}(A)}{\operatorname{argmin}} \left\{ \max_i |1 - \alpha\lambda_i(A)| \right\}$$

Abbiamo che

$$|1 - \alpha_{opt}\lambda_{min}(A)| = |1 - \alpha_{opt}\lambda_{max}(A)|$$

$$\alpha_{opt} = \frac{2}{\lambda_{min}(A) + \lambda_{max}(A)}$$

3.6.3 Massima velocita' di convergenza

Calcoliamo il valore del raggio spettrale in corrispondenza del valore del parametro ottimale:

$$\rho_{opt} = \rho(B_{\alpha_{opt}}) = -1 + \alpha_{opt}\lambda_{max}(A) = |1 - \alpha_{opt}\lambda_{min}(A)| = \frac{\lambda_{max}(A) - \lambda_{min}(A)}{\lambda_{max}(A) + \lambda_{min}(A)}$$

Poiche' A e' simmetrica e definita positiva (SDP), abbiamo $||A|| = \lambda_{max}(A)$.

Inoltre vale $\lambda_i(A^{-1}) = \dfrac{1}{\lambda_i(A)}\ \forall i$.

Poiche' anche A^{-1} e' SDP, abbiamo

$$\|A^{-1}\| = \frac{1}{\lambda_{min}(A)} \implies K(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)} \implies \rho opt = \frac{K(A)-1}{K(A)+1}$$

3.7 Tecnica di Precondizionamento

Il valore ottimale $\rho_{opt} = \dfrac{K(A)-1}{K(A)+1}$ esprime la **massima velocita' di convergenza** possibile ottenibile per una matrice A con il metodo di Richardson stazionario.

Tuttavia, matrici malcondizionate (ovvero con $K(A)$ grande) sono caratterizzate da una velocita' di convergenza del metodo molto bassa. Per migliorare la velocita' di convergenza, introduciamo una matrice P SDP e invertibile. Allora, il sistema lineare di partenza e' equivalente al seguente **sistema precondizionato**:

$$P^{-1}A\boldsymbol{x} = P^{-1}\boldsymbol{b}$$

3.7.1 Metodo di Richardson stazionario precondizionato

Applicando il metodo di Richardson stazionario al precedente sistema:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha P^{-1}(\boldsymbol{b} - A\boldsymbol{x}^{(k)}) = \boldsymbol{x}^{(k)} + \alpha P^{-1}\boldsymbol{r}$$

otteniamo gli stessi risultati di convergenza del caso non precondizionato, a patto di sostituire A con $P^{-1}A$ (anch'essa di autovalori reali e positivi):

- Convergenza:
  $0 < \alpha < \dfrac{1}{\lambda_{max}(P^{-1}A)}$
- Valori ottimali:
  $\alpha_{opt} = \dfrac{1}{\lambda_{min}(P^{-1}A) + \lambda_{max}(P^{-1}A)}$
  $\rho_{opt} = \dfrac{K(P^{-1}A)-1}{K(P^{-1}A)+1}$

quindi, se $K(P^{-1}A) \ll K(A)$ otteniamo una **velocita' di convergenza maggiore** rispetto al caso non precondizionato.

Tuttavia, tale metodo precondizionato **non e'** a prescindere piu' veloce del caso non precondizionato. Infatti, ricordando che $\boldsymbol{x}^{(k-1)} = \boldsymbol{x}^{(k)} + \alpha P^{-1}\boldsymbol{r}^{(k)}$, introduciamo il **residuo precondizionato** $z^{(k)} = P^{-1}\boldsymbol{r}^{(k)}$. Pertanto, l'algoritmo e' dato dai seguenti passi ad ogni k:

$\alpha_{opt} = \dfrac{2}{\lambda_{min}(P^{-1}A) + \lambda_{max}(P^{-1}A)}$
$\boldsymbol{r}^{(k)} = \boldsymbol{b} - A\boldsymbol{x}^{(k)}$
$Pz^{(k)} = \boldsymbol{r}^{(k)}$

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_{opt} z^{(k)}$$

---

al terzo passo dobbiamo **risolvere un sistema lineare in** P, assente nel caso precondizionato.

Pertanto, sebbene con $K(P^{-1}A) \ll K(A)$ il metodo permette di ridurre il numero di iterazioni necessarie per la convergenza, il sistema lineare in P deve essere facilmente risolvibile. A tal fine, P deve avere una struttura particolare, ad esempio essere **diagonale**, **triangolare**, o data d un prodotto di quest'ultime.

3.7.2 Fattorizzazione LU inesatta

Utile per **matrici sparse**. E' data da $P_{ILU} = \tilde{L}\tilde{U}$, con $\tilde{L}$ e $\tilde{U}$ ottenuti effettuando il MEG, in cui pero' si fissano a priori $\tilde{\ell}_{ij} = 0$ e $\tilde{u}_{ij} = 0$ se $a_{ij} = 0$.

In questo modo, i due fattori $\tilde{L}$ e $\tilde{U}$ hanno la stessa sparsita' di A e si puo' quindi usare la sua occupazione di memoria per memorizzarli (ovvero non c'e' fill-in).

Ovviamente $\tilde{L}\tilde{U} \neq A$. L'idea e' quindi di usarlo come precondizionatore perche' contiene informazioni su A ed e' di facile risoluzione (2 sistemi triangolari + MEG incompleto $\sim n^2$ operazioni).

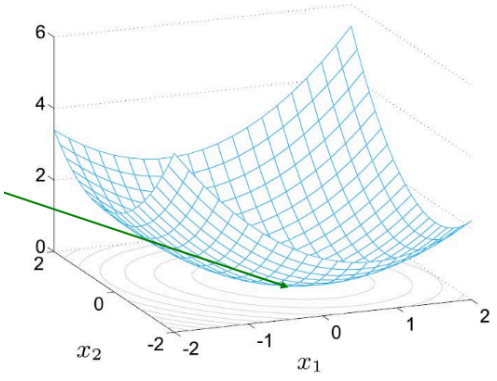
3.8 Metodo del Gradiente

Il metodo del gradiente ha un principio di funzionamento simile a Richardson stazionario, ma invece di essere "stazionario", il parametro α viene aggiornato ad ogni iterazione (dinamico).

Introduciamo la seguente funzione energia Φ:

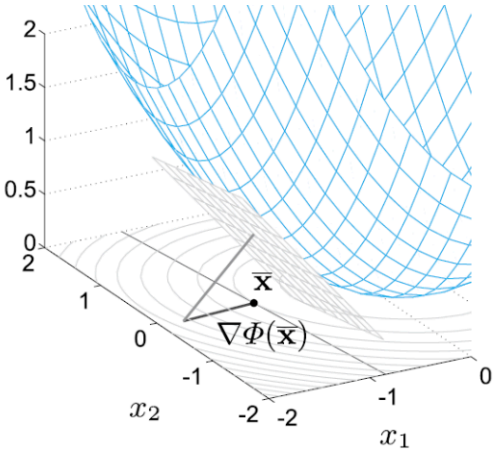
$$\Phi(\boldsymbol{y}) = \frac{1}{2}\boldsymbol{y}^T A \boldsymbol{y} - \boldsymbol{y}^T \boldsymbol{b}, \qquad \Phi : \mathbb{R}^n \to \mathbb{R}$$

Se A e' SDP, l'energia e' una funzione convessa che ammette un unico punto di minimo:



Poiche' $\nabla\Phi(\boldsymbol{y}) = A\boldsymbol{y} - \boldsymbol{b}$, abbiamo che il punto di minimo (gradiente nullo) coincide con la soluzione di $A\boldsymbol{x} = \boldsymbol{b}$.

Dato un punto $\bar{\boldsymbol{x}}$, il vettore $\nabla\Phi(\bar{\boldsymbol{x}})$ individua la direzione di massima decrescita dell'energia:



Il metodo del gradiente assume come direzione di aggiornamento quella di massima decrescita dell'energia nel punto $\boldsymbol{x}^{(k)}$:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \nabla\Phi\left(\boldsymbol{x}^{(k)}\right)$$

dove α_k e' un parametro dinamico.

Ricordando che $\nabla\Phi(\boldsymbol{y}) = A\boldsymbol{y} - \boldsymbol{b}$, si ha $-\nabla\Phi\left(\boldsymbol{x}^{(k)}\right) = -A\boldsymbol{x}^{(k)} + \boldsymbol{b} = \boldsymbol{r}^{(k)}$. Ovvero

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \alpha_k \boldsymbol{r}^{(k)}$$

Il vantaggio di questo metodo e' che il parametro α_k si puo' scegliere in maniera ottimale ad ogni iterazione e, soprattutto, non richiede piu' di conoscere gli autovalori di A (che spesso vanno approssimati essi stessi per via numerica).

3.8.1 Scelta del parametro dinamico

Ad ogni iterazione, il parametro dinamico α_k si ottiene con (dimostrazione non richiesta):

$$\alpha_k = \frac{\left(\boldsymbol{r}^{(k)}\right)^T \boldsymbol{r}^{(k)}}{\left(\boldsymbol{r}^{(k)}\right)^T A\boldsymbol{r}^{(k)}}$$

Notiamo che

$$\boldsymbol{r}^{(k)} = \boldsymbol{b} - A\left(\boldsymbol{x}^{(k-1)} + \alpha_{k-1}\boldsymbol{r}^{(k-1)}\right) = (I - \alpha_{k-1}A)\boldsymbol{r}^{(k-1)}$$

pertanto il prodotto matrice-vettore nel calcolo di α_k viene poi usato nel calcolo del residuo (che quindi ha un costo computazionale pari alla somma di due vettori).

Un algoritmo efficiente quindi e'

Dato $\boldsymbol{x}^{(0)}$, calcolo il residuo iniziale $\boldsymbol{r}^0 = \boldsymbol{b} - A\boldsymbol{x}$, e per ogni $k \geq 0$ determino:

$$\alpha_k = \frac{\left(\boldsymbol{r}^{(k)}\right)^T \boldsymbol{r}^{(k)}}{\left(\boldsymbol{r}^{(k)}\right)^T A\boldsymbol{r}^{(k)}} \qquad \text{Costo} \sim n^2 \text{ operazioni, o} \sim n \text{ operazioni se } A \text{ e' sparsa}$$

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{r}^{(k)} \qquad \text{Costo} \sim n \text{ operazioni}$$

$$\boldsymbol{r}^{(k+1)} = (I - \alpha_k A)\boldsymbol{r}^{(k)} \qquad \text{Costo} \sim n \text{ operazioni perche } A\boldsymbol{r}^{(k)} \text{ e' noto}$$

3.9 Metodo del Gradiente Coniugato

La velocita' di convergenza del metodo del gradiente e' la stessa di Richardson stazionario ottimale:

$$\|\boldsymbol{e}^{(k+1)}\|_A \leq \frac{K(A)-1}{K(A)+1}\|\boldsymbol{e}^{(k)}\|_A \implies \|\boldsymbol{e}^{(k+1)}\|_A \leq \left(\frac{K(A)-1}{K(A)+1}\right)^k \|\boldsymbol{e}^{(0)}\|_A$$

dove, dato un vettore $\boldsymbol{z}$, la norma A di $\boldsymbol{z}$ e' data da

$$\|\boldsymbol{z}\|_A = \sqrt{(\boldsymbol{z},\boldsymbol{z})_A} = \sqrt{\boldsymbol{z}^T A \boldsymbol{z}}$$

poiche' si e' introdotto il prodotto scalare in A: $(\boldsymbol{v},\boldsymbol{z})_A = \boldsymbol{v}^T A \boldsymbol{z}$.

La direzione di aggiornamento data dal residuo e' **ortogonale** alla precedente, per costruzione. Infatti il gradiente e' sempre ortogonale alla direzione precedente, cioe'

$$0 = \left(\nabla \Phi\left(\boldsymbol{x}^{(k+1)}\right), \boldsymbol{r}^{(k)}\right)_2 = -\left(\boldsymbol{r}^{(k+1)}, \boldsymbol{r}^{(k)}\right)_2$$

La scelta della direzione di massima decrescita e' quindi ottimale fra un'iterazione e l'alta, ma potrebbe non esserlo a medio/lungo termine. Infatti, in generale si ha

$$\left(\boldsymbol{r}^{(k+1)}, \boldsymbol{r}^{(j)}\right) \neq 0 \qquad j < k$$

Il carattere esplorativo di $\mathbb{R}^n$ da parte del gradiente non e' quindi elevato. Per superare tale limite, si introducono nuove direzioni di aggiornamento $\boldsymbol{d}^{(k)}$ tutte ortogonali (rispetto al prodotto scalare in A) fra di loro:

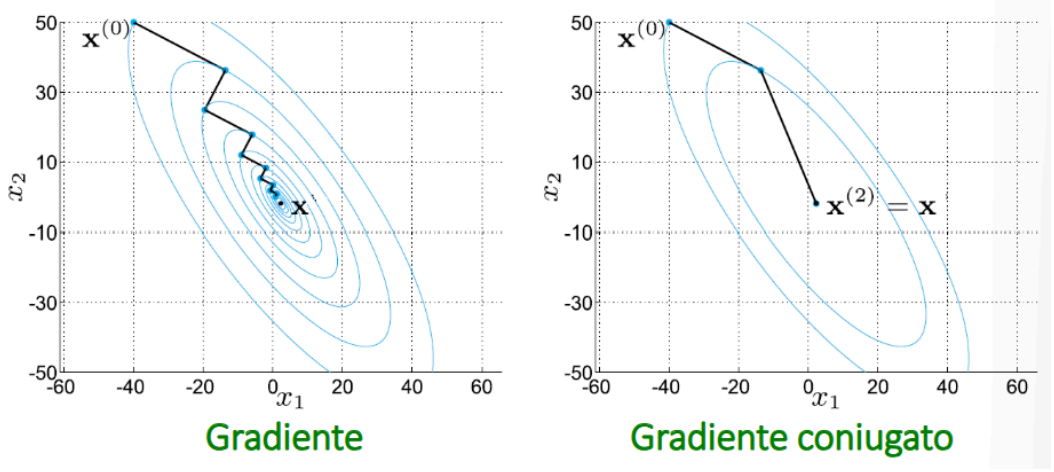
$$\left(\boldsymbol{d}^{(k)}, A\boldsymbol{d}^{(j)}\right) = \left(\boldsymbol{d}^{(k)}, \boldsymbol{d}^{(j)}\right)_A = 0 \qquad \forall j < k$$

I costi computazionali della singola iterazione sono paragonabili a Richardson e Gradiente. E' efficace per matrici sparse.

Teorema

In aritmetica esatta il metodo del gradiente coniugato (GC) converge alla soluzione esatta in al piu' n iterazioni.

Tuttavia, a causa della propagazione degli errori di arrotondamento, cio' di fatto non avviene, ma indica la alta velocita' di convergenza del metodo.



Se A e' SDP, l'errore fra l'iterazione k e quella di partenza scala nel seguente modo:

$$\|\boldsymbol{e}^{(k)}\|_A \leq \frac{1c^k}{1+c^{2k}}\|\boldsymbol{e}^{(0)}\|_A \qquad c = \frac{\sqrt{K(A)}-1}{\sqrt{K(A)}+1}$$

Per matrici SDP, il metodo del gradiente coniugato converge piu' velocemente del metodo del gradiente.

3.9.1 Gradiente coniugato precondizionato

In questo caso, l'errore si comporta come quello del caso non precondizionato, a patto di sostituire A con $P^{-1}A$. Ad esempio, per il gradiente coniugato precondizionato, si ha:

$$\|\boldsymbol{e}^{(k)}\|_A \leq \frac{1c^k}{1+c^{2k}}\|\boldsymbol{e}^{(0)}\|_A \qquad c = \frac{\sqrt{K(P^{-1}A)}-1}{\sqrt{K(P^{-1}A)}+1}$$

Se il precondizionatore e' scelto bene, si ha:

$$\frac{\sqrt{K(P^{-1}A)}-1}{\sqrt{K(P^{-1}A)}+1} < \frac{\sqrt{K(A)}-1}{\sqrt{K(A)}+1}$$

e quindi l'errore nel caso precondizionato si riduce piu' velocemente rispetto al caso non condizionato.

3.10 Criteri di arresto

Come discusso, servono dei criteri per arrestare il processo iterativo. Cio' deve avvenire quando stimiamo che l'errore $\boldsymbol{e}^{(k)} = \boldsymbol{x} - \boldsymbol{x}^{(k)}$ (che non possiamo calcolare non conoscendo $\boldsymbol{x}$) sia sufficientemente piccolo.

Possiamo utilizzare 2 possibili criteri:

1. **Criterio sul residuo** - Abbiamo gia' visto

$$\frac{\|\boldsymbol{x}-\boldsymbol{x}^{(k)}\|}{\|\boldsymbol{x}\|} \leq K(A)\frac{\|\boldsymbol{r}^{(k)}\|}{\|\boldsymbol{b}\|} \implies \frac{\|\boldsymbol{r}^{(k)}\|}{\|\boldsymbol{b}\|} \leq \varepsilon$$

dove ε e' la **tolleranza assegnata**.

Questo e' un buon criterio quando il condizionamento e' basso. Viene quindi utilizzato spesso per metodi precondizionati (che quindi hanno $K(P^{-1}A)$ piccolo):

$$\frac{\|\boldsymbol{x}-\boldsymbol{x}^{(k)}\|}{\|\boldsymbol{x}\|} \leq K(P^{-1}A)\frac{\|\boldsymbol{z}^{(k)}\|}{\|\boldsymbol{b}\|} \implies \frac{\|\boldsymbol{z}^{(k)}\|}{\|\boldsymbol{b}\|} \leq \varepsilon \qquad \boldsymbol{z}^{(k)} = P^{-1}\boldsymbol{r}^{(k)}$$

2. **Criterio sull'incremento** - Definiamo $\boldsymbol{\delta}^{(k)} = \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}$, abbiamo che

   $$\|\boldsymbol{\delta}^{(k)}\| \leq \varepsilon$$

   Vale la seguente relazione fra errore e incremento:

   $$\|\boldsymbol{e}^{(k)}\| \leq \frac{1}{1-\rho(B)}\|\boldsymbol{\delta}^{(k)}\|$$

   infatti si ha

   $$\|\boldsymbol{e}^{(k)}\| = \|\boldsymbol{x}-\boldsymbol{x}^{(k)}\| = \|\boldsymbol{x}-\boldsymbol{x}^{(k+1)}+\boldsymbol{x}^{(k+1)}-\boldsymbol{x}^{(k)}\|$$

   $$= \|\boldsymbol{e}^{(k+1)}+\boldsymbol{\delta}^{(k)}\| \leq \rho(B)\|\boldsymbol{e}^{(k)}\| + \|\boldsymbol{\delta}^{(k)}\|$$

   Quindi l'incremento e' un buon criterio se $\rho(B)$ e' piccolo. Non e' invece affidabile se $\rho(B)$ e' vicino a 1.

4 Radici di funzioni non lineari

Data $f : [a,b] \to \mathbb{R}$, vogliamo trovare $\alpha \in (a,b)$ tale che $f(\alpha) = 0$, dove α e' detta **zero** o **radice** dell'equazione $f(\alpha) = 0$.

4.1 Metodi iterativi locali

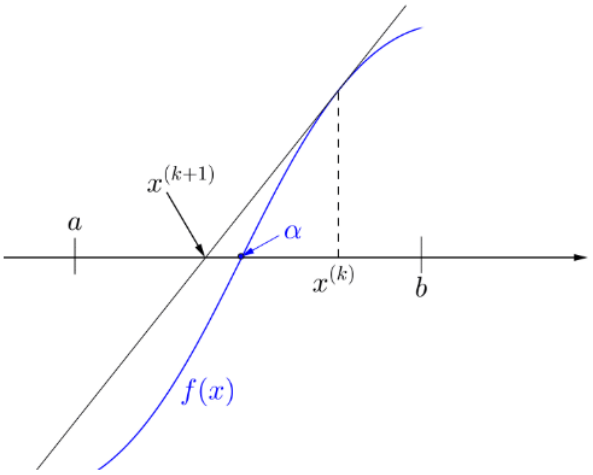
Ricordiamo l'idea alla base dei metodi iterativi:

1. Scelgo $x^{(0)}$
2. Genero una successione $x^{(1)}, x^{(2)}, \ldots, x^{(k)}, \ldots$
3. Voglio avere la convergenza $\lim_{k\to\infty} x^{(k)} = \alpha$

Abbiamo tre metodi iterativi locali:

- **Metodo di Newton** - Considero la tangente ad f in $x^{(k)}$,
  $y(x) = f(x^{(k)}) - f'(x^{(k)})(x - x^{(k)})$, e scelgo $x^{(k+1)}$ tale che $y\left(x^{(k+1)}\right) = 0$. Si ottiene:

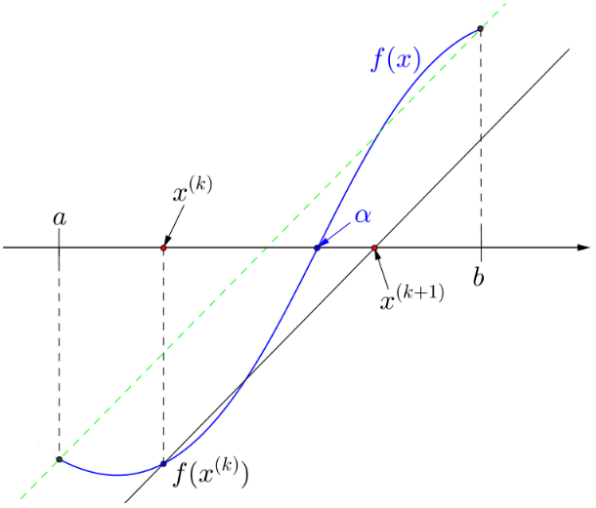
  $$x^{(k+1)} = x^{(k)} - \frac{f\left(x^{(k)}\right)}{f'\left(x^{(k)}\right)}, \qquad k > 0$$



Come si puo' notare, abbiamo bisogno della derivata prima. Possiamo evitare con i metodi successivi.

- **Metodo delle corde** - Sostituisce $f'\left(x^{(k)}\right)$ con q

$$x^{(k+1)} = x^{(k)} - \frac{f\left(x^{(k)}\right)}{q}, \qquad q = \frac{f(b) - f(a)}{b - a} \neq 0$$



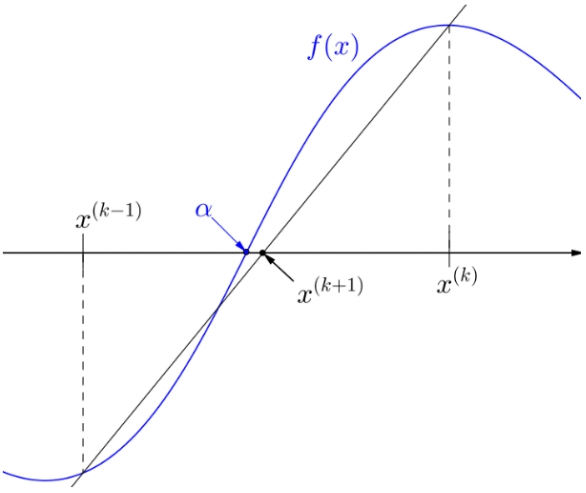
- **Metodo delle secanti** - Sostituisce $f'(x)$ con il *rapporto incrementale*

$$\frac{f\left(x^{(k)}\right) - f\left(x^{(k-1)}\right)}{x^{(k)} - x^{(k-1)}}$$

ottenendo:

$$x^{(k+1)} = x^{(k)} - \left(x^{(k)} - x^{(k+1)}\right) \frac{f\left(x^{(k)}\right)}{f\left(x^{(k)}\right) - f\left(x^{(k-1)}\right)}, \qquad k \geq 0$$

Bisogna pero' fornire anche $x^{(-1)}$ come dato di partenza.



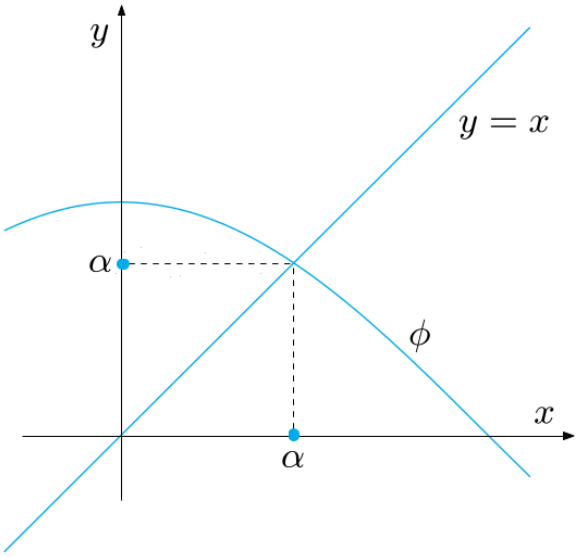
Questi 3 metodi sono **metodi locali**, cioe' la convergenza e' garantita solo se $x^{(0)}$ e' preso sufficientemente vicino ad α, ossia se $\exists \delta > 0 : |\alpha - x^{(0)}| < \delta$.

Questa condizione e' difficile da applicare a priori (δ e' difficilmente stimabile). Si possono pero' usare considerazioni fisiche in base al problema di partenza.

4.2 Iterazioni di punto fisso

Supponiamo di voler trovare lo zero di $f(x) = x - \cos(x)$. Osservo che se α e' uno zero di f, cioe' $f(\alpha) = \alpha - \cos(\alpha)$, allora α soddisfa $\alpha = \cos(\alpha)$.

Detta $\phi(x)$ la funzione di cui si cercano i punti fissi, cioe' le soluzioni dell'equazione $x = \phi(x)$, riconosciamo che α e' punto fisso per la funzione $\cos(x)$. Geometricamente significa



In altre parole, possiamo riscrivere il problema come:

$$\text{Trovare } \alpha \text{ tale che } f(\alpha) = 0 \iff \text{Cercare } \alpha \text{ tale che } \begin{cases} \alpha = \phi(\alpha) \\ \phi(x) = x - f(x) \end{cases}$$

Un metodo per la ricerca dei punti fissi di una funzione $\phi : [a, b] \to \mathbb{R}$ lo si puo' trovare considerando il seguente metodo iterativo (**metodo delle iterazioni di punto fisso**):

$$\begin{cases} \text{Dato } x^{(0)} \\ x^{(k+1)} = \phi\left(x^{(k)}\right), \quad k \geq 0 \end{cases}$$

ϕ e' detta **funzione di iterazione di punto fisso**.

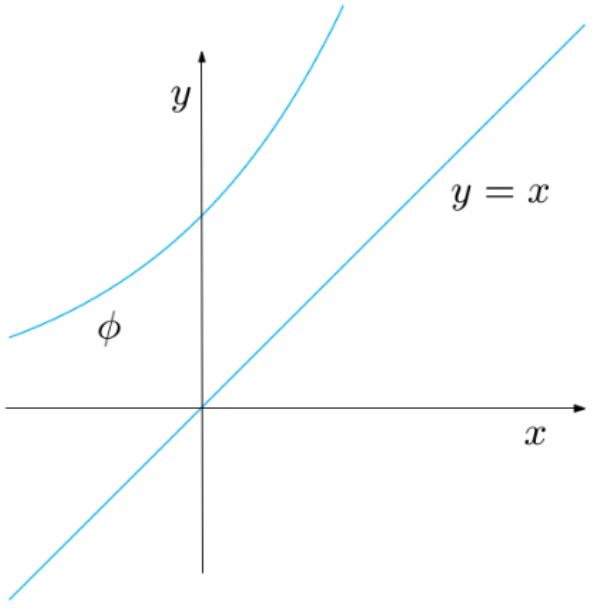
Esempio

Nel nostro caso cerchiamo il funto fisso della funzione $\cos$. La successione generata quindi sara' la seguente:

$$\begin{aligned} x^{(0)} &= 1 \\ x^{(1)} &= \cos\left(x^{(0)}\right) = \cos(1) = 0.54030230586814 \\ x^{(2)} &= \cos\left(x^{(1)}\right) = 0.85755321584639 \\ &\vdots \\ x^{(20)} &= \cos\left(x^{(19)}\right) = 0.73918439977149 \end{aligned}$$

Che tende al valore di $\alpha = 0.73908513$.

In generale non e' vero che un'iterazione di punto fisso e' sempre convergente. Ad esempio, se cerco i punti fissi della funzione esponenziale $\phi(x) = e^x$. In questo caso non ci sono punti fissi, le iterazioni di punto fisso $x^{(k+1)} = \phi\left(x^{(k)}\right)$, con $k \geq 0$ non convergono.



4.2.1 Teorema: $\exists!$ punto fisso

Sia $\phi : [a, b] \to \mathbb{R}$ continua, e sia $x^{(0)}$ in $[a, b]$ assegnato. Consideriamo le iterazioni di punto fisso

$$x^{(k+1)} = \phi\left(x^{(k)}\right), \qquad k \geq 0$$

1. Se $\forall x \in [a,b]$ si ha $\phi(x) \in [a,b]$, allora $\exists$ almeno un punto fisso $\alpha \in [a,b]$
2. Se inoltre
   $\exists L > 1$ tale che $|\phi(x_1) - \phi(x_2)| \leq L|x_1 - x_2|,\ \forall x_1, x_2 \in [a,b]$, allora
   1. $\exists! \alpha \in [a,b]$ tale che $\phi(\alpha) = \alpha$
   2. $\forall x^{(0)} \in [a,b],\ \lim_{k\to\infty} x^{(k)} = \alpha$

(dimostrazione non richiesta)

4.3 Convergenza locale

Sia $\phi : [a,b] \to \mathbb{R}$ funzione di iterazione, e supponiamo che α sia un suo punto fisso. Sia inoltre $\phi \in C^1(I_\alpha)$, dove I_α e' un opportuno intorno di α. Allora:

1. Se $|\phi'(\alpha)| < 1$, allora $\exists \delta > 0$ tale che $\forall x^{(0)} : |x^{(0)} - \alpha| < \delta$, la successione $x^{(k)} \to \alpha$. Inoltre:

$$\lim_{k\to\infty} \frac{x^{(k+1)} - \alpha}{(x^{(k)} - \alpha)^1} = \phi'(\alpha) \qquad \text{ordine 1}$$

2. Se inoltre $\phi \in C^2(I_\alpha)$ e $\phi'(\alpha) = 0, \phi''(\alpha) \neq 0$, allora la successione $x^{(k)} \to \alpha$. Inoltre:

$$\lim_{k\to\infty} \frac{x^{(k+1)} - \alpha}{(x^{(k)} - \alpha)^2} = \frac{\phi''(\alpha)}{2} \qquad \text{ordine 2}$$

   La convergenza e' piu' veloce nel caso di ordine 2.

Le precedenti relazioni valgono per$|k \to \infty$. Per un generico k possiamo scrivere le seguenti disuguaglianze:

1. Se $|\phi'(\alpha)| < 1$, allora

$$\left|x^{(k+1)} - \alpha\right| \leq \underbrace{|\phi'(\alpha)|}_{<1} \left|x^{(k)} - \alpha\right|$$

   cioe' l'errore al passo $k+1$ scala linearmente rispetto all'errore al passo k (ordine 1).
2. Se inoltre, $\phi \in C^2(I_\alpha)$ e $\phi'(\alpha) = 0, \phi''(\alpha) \neq 0$, allora

$$\left|x^{(k+1)} - \alpha\right| \leq \frac{|\phi''(\alpha)|}{2} \left|x^{(k)} - \alpha\right|^2$$

   cioe' l'errore al passo k scala quadraticamente rispetto all'errore al passo k (ordine 2).

4.3.1 Metodo delle corde

Se f e' regolare, anche ϕ_C lo e', e vale

$$\phi_C(x) = x - \frac{1}{q} f(x), \qquad \phi_C'(x) = 1 - \frac{1}{q} f'(x), \qquad q = \frac{f(b) - f(a)}{b - a}$$

Considero α punto fisso (zero di f):

$$\phi'(\alpha) = 1 - \frac{1}{q} f'(\alpha)$$

se $f'(\alpha) = 0$ non so dire nulla, invece se $f'(\alpha) \neq 0$ allora il metodo converge se e solo se

$$\left| 1 - \frac{1}{q} f'(\alpha) \right| < 1 \implies -1 < 1 - \frac{1}{q} f'(\alpha) < 1$$

$$\implies \begin{cases} (i) & \frac{1}{q} f'(\alpha) > 0 \implies q \text{ ed } f'(\alpha) \text{ devono avere lo stesso segno} \\ (ii) & \frac{1}{q} f'(\alpha) < 2 \implies \frac{b-a}{f(b)-f(a)} f'(\alpha) < 2 \implies b - a < \frac{2}{f'(\alpha)} [f(b) - f(a)] \end{cases}$$

se valgono entrambe (i) e (ii), allora il metodo delle corde ha convergenza lineare.

4.3.2 Metodo di Newton

$$\Phi_N(x) = x - \frac{f(x)}{f'(x)}, \qquad \phi_N'(x) = \frac{f(x) f''(x)}{[f'(x)]^2}$$

infatti

$$\phi_N'(x) = 1 - \frac{[f'(x)]^2 - f(x) f''(x)}{[f'(x)]^2} = 1 - 1 + \frac{f(x) f''(x)}{[f'(x)]^2} = \frac{f(x) f''(x)}{[f'(x)]^2}$$

inoltre, si ha

$$\phi_N''(x) = \frac{[f(x) f''(x)]' [f'(x)]^2 - f(x) f''(x) 2 f'(x) f''(x)}{[f'(x)]^4}$$

$$= \frac{f'(x) f''(x) + f(x) f'''(x)}{[f(x)]^2} - 2 \frac{f(x) [f''(x)]^2}{[f(x)]^3}$$

se α e' una radice semplice si ha $f(\alpha) = 0$ e $f'(\alpha) \neq 0$, quindi:

$$\phi_N'(\alpha) = \frac{f(\alpha) f''(\alpha)}{[f'(\alpha)]^2} = 0, \qquad \phi_N''(\alpha) = \frac{f''(\alpha)}{f'(\alpha)} \neq 0$$

Quindi, applicando il teorema, il metodo di Newton converge localmente ed e' di ordine 2.

4.3.3 Metodo di Newton modificato

Se α ha molteplicita' $m > 1$ (cioe' se le derivate $f^{(j-1)}(\alpha) = 0, j \leq m$), allora Newton **converge** ancora, ma non e' piu' del secondo ordine. Infatti, in generale si puo' dimostrare che:

$$\phi_N'(\alpha) = 1 - \frac{1}{m}$$

Quindi, per **ripristinare l'ordine di convergenza**, si modifica il metodo di Newton nel seguente modo:

$$x^{(k+1)} = x^{(k)} - m\frac{f\left(x^{(k)}\right)}{f'\left(x^{(k)}\right)}, \qquad \phi_{Nmod}(x) - x - m\frac{f(x)}{f'(x)}$$

Il valore di m e' spesso incognito a priori, ci vogliono stime numeriche.

4.4 Criteri di arresto

Supponiamo che $\{x^{(k)}\}_k$ sia una successione di approssimazione di uno zero di $f(x)$ generata da uno dei metodi precedenti. Supponiamo inoltre che $f \in C^{-1}(I_\alpha), \forall k$, e indichiamo con $e^{(k)} = \alpha - x^{(k)}$ l'errore al passo k.

Vogliamo sapere quando arrestare il metodo iterativo. Esistono due criteri: *criterio sul residuo*, e *criterio sull'incremento*.

4.4.1 Criterio sul residuo

Fissato $\varepsilon > 0$, si arresta il metodo se $\left|f\left(x^{(k)}\right)\right| < \varepsilon$. Inoltre, si puo' dimostrare che:

- Se $|f'(\alpha)| \simeq 1 \implies \left|e^{(k)}\right| \simeq \varepsilon$ e il criterio di arresto e' soddisfacente.
- Se $|f'(\alpha)| \ll 1 \implies$ test inaffidabile perche' potrebbe valere $\left|e^{(k)}\right| \gg \varepsilon$.
- Se $|f'(\alpha)| \gg 1 \implies$ test restrittivo perche' $\left|e^{(k)}\right| \ll \varepsilon$, ma comunque il criterio e' valido (puo' comportare ulteriori iterazioni "inutili").

4.4.2 Criterio sull'incremento

Fissato $\varepsilon > 0$, si arresta il metodo se $\left|x^{(k+1)} - x^{(k)}\right| < \varepsilon$. Inoltre, si puo' dimostrare che:

- Se $-1 < \phi'(\alpha) < 0 \implies$ test soddisfacente.
- Per i metodi del II ordine, vale $\phi'(\alpha) = 0 \implies$ test che va bene $\implies$ ok per Newton.
- Se $\phi'(\alpha) \simeq 1 \implies$ test insoddisfacente.

5 Metodi numerici per sistemi non-lineari

Abbiamo il seguente problema:

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0} \iff \begin{cases} f_1(x_1, \dots, x_j, \dots, x_n) = 0 \\ \vdots \\ f_i(x_1, \dots, x_j, \dots, x_n) = 0 \\ \vdots \\ f_n(x_1, \dots, x_j, \dots, x_n) = 0 \end{cases}$$

dove:

- $\boldsymbol{x} \in \mathbb{R}^n$ e' il vettore delle incognite x_j

- $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ con f_i che esprimono delle relazioni non-lineari tra le incognite
- Indichiamo con $\boldsymbol{\alpha}$ le radici di $\boldsymbol{f}$.

Dal punto di vista numerico, abbiamo 2 difficolta: e' un sistema, e quindi ho equazioni accoppiate; e' non-lineare, e quindi va linearizzato.

5.1 Metodo di Newton

Riprendiamo il metodo di Newton per il caso scalare, cioe' per determinare radici di una funzione non-lineare:

$$x^{(k+1)} = x^{(k)} - \frac{f\left(x^{(k)}\right)}{f'\left(x^{(k)}\right)}$$

possiamo riscriverlo in maniera equivalente come:

$$f'\left(x^{(k)}\right)\delta x^{(k)} = -f\left(x^{(k)}\right)$$

$$x^{(k+1)} = x^{(k)} + \delta x^{(k)}$$

Dato un punto $\boldsymbol{z} \in \mathbb{R}^n$, introduciamo la matrice Jacobiana relativa a $\boldsymbol{f}$

$$J(\boldsymbol{z}) = \nabla \boldsymbol{f}(\boldsymbol{z}), \qquad j_{i\ell} = \frac{\partial f_i(\boldsymbol{z})}{\partial x_\ell}, \qquad i, \ell = 1, \dots, n$$

Per ogni $\boldsymbol{z} \in \mathbb{R}^n$ si ha quindi $J(\boldsymbol{z}) \in \mathbb{R}^{n \times n}$.

Introduciamo il metodo di Newton per la risoluzione numerica del sistema non lineare $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}$, cioe' per la determinazione delle sue radici $\boldsymbol{\alpha}$. Dato $\boldsymbol{x}^{(0)}$, se $J\left(\boldsymbol{x}^{(k)}\right)$ e' una matrice non singolare, per ogni k risolvo:

$$J\left(\boldsymbol{x}^{(k)}\right)\delta \boldsymbol{x}^{(k)} = -\boldsymbol{f}\left(\boldsymbol{x}^{(k)}\right)$$

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \delta \boldsymbol{x}^{(k)}$$

Ad ogni iterazione k, il primo passo del metodo di Newton consiste nella risoluzione di un sistema lineare di dimensione n:

1. $J\left(\boldsymbol{x}^{(k)}\right)\delta \boldsymbol{x}^{(k)} = -\boldsymbol{f}\left(\boldsymbol{x}^{(k)}\right)$ sistema lineare $n \times n$.
   Infatti $J\left(\boldsymbol{x}^{(k)}\right)$ e' una matrice non singolare di **coefficienti noti** $\dfrac{\partial f_i\left(\boldsymbol{x}^{(k)}\right)}{\partial x_\ell}$ e $-\boldsymbol{f}\left(\boldsymbol{x}^{(k)}\right)$ e' il termine noto. Una volta risolto il sistema lineare, e ottenuto quindi $\delta \boldsymbol{x}^{(k)}$, aggiorno la $\boldsymbol{x}^{(k+1)}$ mediante il secondo passo del metodo.
2. $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \delta \boldsymbol{x}^{(k)}$ aggiornamento.

Ripetendo il tutto ad ogni iterazione k.

Analogamente al caso scalare, il metodo di Newton per sistemi e'

- Un **metodo locale**: se $\exists \delta > 0 : \left\| \boldsymbol{\alpha} - \boldsymbol{x}^{(0)} \right\| < \delta$, allora si ha convergenza:

$$\lim_{k \to \infty} \left\| \boldsymbol{\alpha} - \boldsymbol{x}^{(k)} \right\| = 0$$

- Un **metodo del secondo ordine**: se converge e J e' derivabile, allora si ha:

$$\frac{\left\| \boldsymbol{\alpha} - \boldsymbol{x}^{(k+1)} \right\|}{\left\| \boldsymbol{\alpha} - \boldsymbol{x}^{(k)} \right\|^2} \leq C$$

5.2 Criteri di arresto

Essendo un metodo iterativo, bisogna introdurre opportuni criteri di arresto. Analogamente al caso scalare, si introducono:

- Criterio sul **residuo**: scelgo la tolleranza ε, arresto le iterazioni quando

$$\left\| \boldsymbol{f}\left(\boldsymbol{x}^{(k)}\right) \right\| < \varepsilon$$

- Criterio sull'**incremento**: scelgo la tolleranza ε, arresto le iterazioni quando

$$\left\| \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)} \right\| < \varepsilon$$

5.3 Costi computazionali e varianti

Il metodo di newton richiede la soluzione di un sistema lineare ad ogni iterazione. Notiamo che $J\left(\boldsymbol{x}^{(k)}\right)$ cambia ad ogni iterazione k e quindi va ricostruita ogni volta.

Per velocizzare il metodo di Newton si possono introdurre le seguenti varianti.

5.3.1 Aggiornamento Jacobiana ogni p iterazioni

Dato un numero intero $p \geq 2$, l'idea e' di aggiornare la matrice Jacobiana $J\left(\boldsymbol{x}^{(k)}\right)$ ogni p volte. Il tempo di costruzione e' quindi ridotto, dovendosi effettuare ogni p iterazioni.

Se si utilizza la fattorizzazione LU per risolvere il sistema lineare, allora il costo della fattorizzazione di $J\left(\boldsymbol{x}^{(k)}\right)$ (ovvero $\frac{2}{3}n^3$) e' ripetuto su p iterazioni. Di fatto ad ogni iterazione mediamente ho un costo di $\frac{2n^3}{3p}$.

Il tutto al prezzo di un numero di iterazioni maggiori (ordine <2). Piu' p e' grande, piu' aumenta il numero di iterazioni (p non deve essere pero' troppo grande altrimenti si perde la convergenza). La speranza e' che il costo complessivo sia ridotto.

5.3.2 Inexact Newton

L'idea e' di sostituire ad ogni iterazione k la matrice Jacobiana $J\left(\boldsymbol{x}^{(k)}\right)$ con una sua approssimazione $\tilde{J}^k$ che sia in generale facilmente costruibile, e tale che il sistema lineare

associato sia facilmente risolvibile. Il tutto ancora al prezzo di un numero di iterazioni maggiori (ordine < 2) di quelle che si hanno con Newton.

Notare come nonostante il nome, questo e' un **metodo esatto** (inexact si riferisce alla Jacobiana): a convergenza avremo comunque la soluzione $\boldsymbol{\alpha}$. Quello che cambia e' la velocita' di convergenza.

6 Approssimazione di Dati e Funzioni

Supponiamo di conoscere $n+1$ coppie di dati $(x_i, y_i), i = 0, \dots, n$. In generale, queste possono costituire:

1. **Misure** y_i di una certa quantita' fisica effettuate sperimentalmente in corrispondenza dei punti x_i.
2. **Valori di una funzione** $f(x_i)$ in corrispondenza dei punti x_i.

Il problema dell'approssimazione dei dati o di una funzione consiste nel determinare una funzione $\tilde{f}(x)$ che abbia delle *buone* proprieta' di approssimazione dei dati $(x_i, y_i), i = 0, \dots, n$ che permetta di avere carattere predittivo anche non in corrispondenza dei nodi.

6.1 Interpolazione Lagrangiana

Date $n+1$ coppie di dati $(x_i, y_i), i = 0, \dots, n$, diciamo che un'approssimante $\tilde{f}(x)$ e' di tipo **interpolatorio** se vale la seguente relazione:

$$\tilde{f}(x_i) = y_i, \qquad \forall i = 0, \dots, n$$

Un interpolatore e' quindi una funzione che assume il valore dei dati in corrispondenza dei nodi x_i.

Si introducono di seguito le seguenti $n+1$ funzioni, associate agli $n+1$ dati $(x_i, y_i), i = 0, \dots, n$, che prendono il nome di **polinomi caratteristici di Lagrange**:

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

cioe'

$$\varphi_k(x) = \frac{(x - x_0)(x - x_1) \dots (x - x_{k-1})(x - x_{k+1}) \dots (x - x_n)}{(x_k - x_0)(x_k - x_1) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_k)}$$

Essi sono dati dal prodotto di n termini di primo grado, percio' sono dei **polinomi di grado** n .

Notiamo che il denominatore e' sempre diverso da 0 per costruzione. Se $i \neq k$, allora uno dei termini a numeratore e' nullo, e quindi $\phi_k(x_i) = 0$. Se $i = k$, allora il numeratore e' uguale

al denominatore. Percio' abbiamo

$$\varphi_k(x_i) = \delta_{ik} = \begin{cases} 1 & \text{se } i = k \\ 0 & \text{se } i \neq k \end{cases}$$

Possiamo quindi introdurre l'interpolatore Lagrangiano degli $n+1$ dati $(x_i, y_i), i = 0, \dots, n$. Esso e' dato dalla seguente espressione:

$$\Pi_n(x) = \sum_{j=0}^{n} y_j \varphi_j(x)$$

Quando stiamo approssimando i valori di una funzione $(x_i, f(x_i))$, si e' soliti indicare l'interpolatore Lagrangiano con $\Pi_n f(x)$.

6.1.1 Proprieta' dell'interpolatore Lagrangiano

Valgono le seguenti proprieta':

1. $\Pi_n(x)$ e' un **interpolatore**. Infatti, valutandolo per un generico nodo x_i si ottiene

$$\Pi_n(x_i) = \sum_{j=0}^{n} y_j \varphi_j(x_i) = \sum_{j=0}^{n} y_i \delta_{ij} = y_i$$

2. $\Pi_n(x)$ e' un **polinomio di grado** n. Infatti, esso e' dato dalla somma dei polinomi di grado n $y_j \varphi_j(x)$.
3. $\Pi_n(x)$ e' l'**unico polinomio di grado** n **interpolante gli** $n+1$ **dati** $(x_i, y_i), i = 0, \dots, n$. Infatti, supponiamo che esista un altro interpolatore polinomiale di grado n $\Psi_n(x)$ che interpola i dati $(x_i, y_i), i = 0, \dots, n$. Introduciamo il seguente polinomio di grado n:

   $$D(x) = \Pi_n(x) - \Psi_n(x)$$

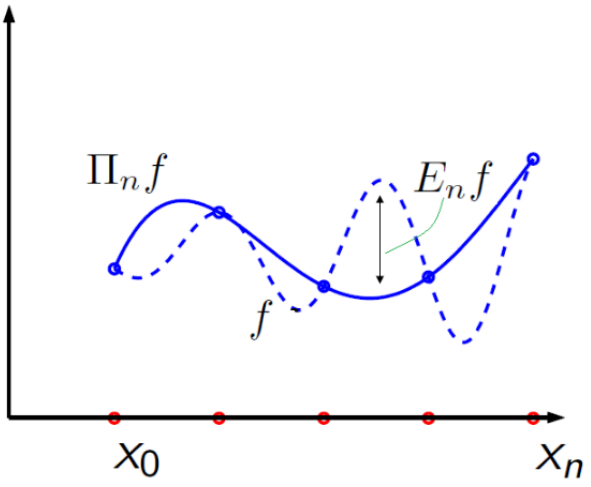
   Vale che $D(x_i) = \Pi_n(x_i) - \Psi_n(x_i) = 0, \forall i = 0, \dots, n$. Allora, $D(x)$ ha $n+1$ zeri. Ma, essendo un polinomio di grado n, si deve avere $D(x) \equiv 0$, da cui segue l'unicita'.

6.1.2 Accuratezza dell'interpolatore Lagrangiano nel caso di approssimazione di funzioni

Si consideri il caso di approssimazione dei valori di una funzione $f(x)$. Si ha il seguente risultato:

> ***Proposizione*** - 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 continuita' fino all'ordine $n+1$ in i. Allora $\forall x \in I,\ \exists \xi_x \in I$ tale che
>
> $$E_n f(x) = f(x) - \Pi_n f(x) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \prod_{i=0}^{n} (x - x_i)$$

Notare come la funzione $E_n f(x)$ si annulli in corrispondenza dei nodi x_i. Il suo valore e' invece, in generale, diverso da zero lontano dai nodi, cioe' per $x \neq x_i$.



Per ottenere un'informazione piu' compatta sull'errore, troviamo una stima del suo valore massimo. Nel caso di nodi x_i equispaziati con $x_{j+1} - x_j = h,\ \forall j = 0, \dots, n-1$ vale la seguente disuguaglianza:

$$\left| \prod_{i=0}^{n} (x - x_i) \right| \leq n! \frac{h^{n+1}}{4}$$

Infatti, si ha che la funzione $\left| \prod_{i=0}^{n} (x - x_i) \right|$ ammette massimo in prossimita' degli estremi. Supponiamo, senza perdita di generalita', che il massimo sia in $x \in (x_{n-1}, x_n)$. Allora, per tale x vale

$$\begin{aligned} \left| \prod_{i=0}^{n} (x - x_i) \right| &= |x - x_0|\ |x - x_1| \dots |x - x_{n-2}|\ |(x - x_{n-1})(x - x_n)| \\ &\leq nh \cdot (n-1)h \cdot \dots \cdot 2h \cdot \frac{h^2}{4} \\ &\leq n! \frac{h^{n+1}}{4} \end{aligned}$$

Dall'espressione dell'errore nella proposizione, abbiamo:

$$\max_{x \in I} |E_n f(x)| = \max_{x \in I} \left| \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \right| \left| \prod_{i=0}^{n} (x - x_i) \right|$$

Introducendo la stima $\left| \prod_{i=0}^{n} (x - x_i) \right| \leq n! \frac{h^{n+1}}{4}$ si ottiene che nel caso di nodi equispaziati vale la seguente **stima dell'errore massimo**:

$$\max_{x \in I} |E_n f(x)| \leq \frac{|\max_{x \in I} f^{(n+1)}(x)|}{4(n+1)} h^{n+1}$$

6.1.3 Convergenza dell'interpolatore Lagrangiano

Vorremmo che, aumentando le informazioni a disposizione (cioe' $n+1$), l'accuratezza di una funzione approssimante migliori. Nello specifico, per l'interpolatore Lagrangiano, vorremmo quindi che

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

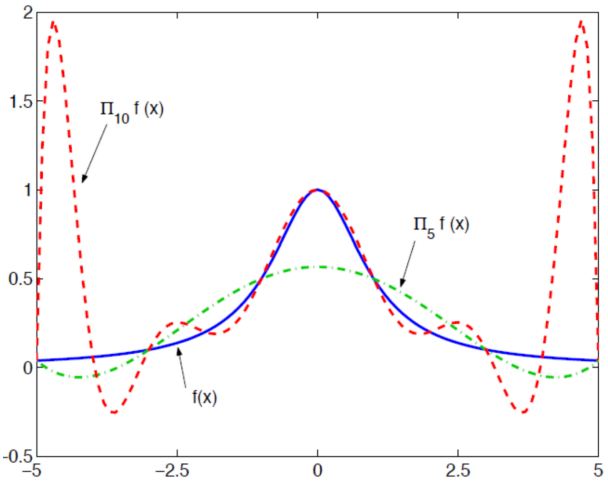
Tornando alla stima dell'errore massimo, abbiamo incertezza solo sul numeratore, non sappiamo a priori se sia limitato per $n \to \infty$, cosa che garantirebbe la convergenza. Esistono infatti dei casi per cui

$$\lim_{n\to\infty} \left| \max_{x\in I} f^{(n+1)}(x) \right| = \infty$$

che portano a $\lim_{n\to\infty} \max_{x\in I} |E_n f(x)| = \infty$, ovvero alla non convergenza dell'interpolatore Lagrangiano.

6.1.4 Fenomeno di Runge

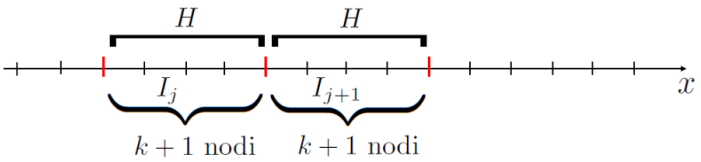
La possibile mancata convergenza dell'interpolatore Lagrangiano prende il nome di fenomeno di Runge. In particolare, in presenza di questo fenomeno, la funzione errore E_n presenta delle oscillazioni ai nodi estremi, che crescono con il crescere di n.



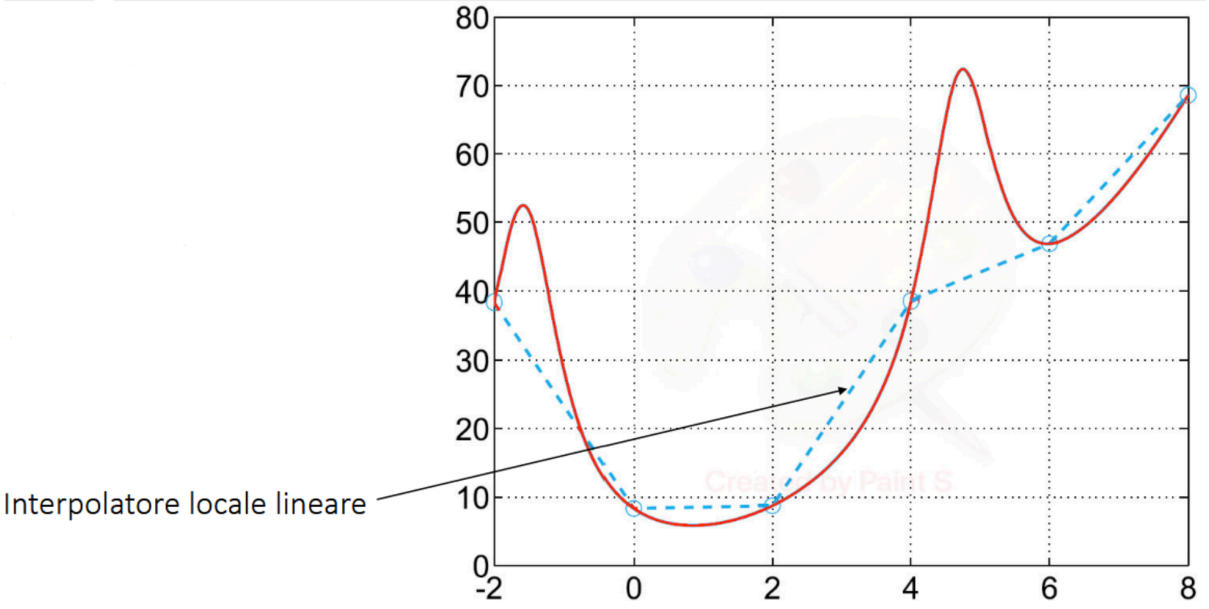
6.2 Interpolazione Lagrangiana Composita

Per superare i limiti dell'interpolazione Lagrangiana, una possibile strategia consiste nell'introdurre un'interpolatore continuo, dato dall'unione di tanti interpolatori Lagrangiani di basso ordine, diciamo $k \ll n$.

Tali interpolatori locali sono costruiti sugli intervalli disgiunti I_j, ognuno composto da $k+1$ nodi e di lunghezza $H = kh$



Tale interpolatore globale viene indicato con $\Pi_k^H(x)$. Quando esso e' stato costruito per approssimare una funzione f, si utilizza la notazione $\Pi_k^H f(x)$.



6.2.1 Accuratezza dell'interpolatore Lagrangiano composito

In questo caso ci aspettiamo che le cose funzionino bene, cioe' che quando aumentiamo il numero di informazioni $n+1$ che abbiamo a disposizione, l'accuratezza dell'interpolatore Lagrangiano composito migliori sempre.

Infatti, questa volta, quando aumenta n faremo aumentare il numero degli intervalli I_j su cui costruire gli interpolatori locali, **senza variare il grado locale k che rimane costante**. Se percio' si lavora con k piccolo (di solito 1, 2 o 3), non si incorrera' piu' nel fenomeno di Runge quando n cresce.

Introducendo l'errore puntuale $E_k^H f(x) = f(x) - \Pi_k^H f(x)$ e notando che esso e' dato dal massimo degli errori degli interpolatori Lagrangiani di grado k su ogni I_j otteniamo, nel caso di nodi equispaziati,

$$\max_{x\in I} |E_k^H f(x)| \leq \max_j \frac{\max_{x\in I_j} \left|f^{(k+1)}(x)\right|}{4(k+1)} h^{k+1} = \frac{\max_{x\in I} \left|f^{(k+1)}(x)\right|}{4(k+1)} h^{k+1}$$

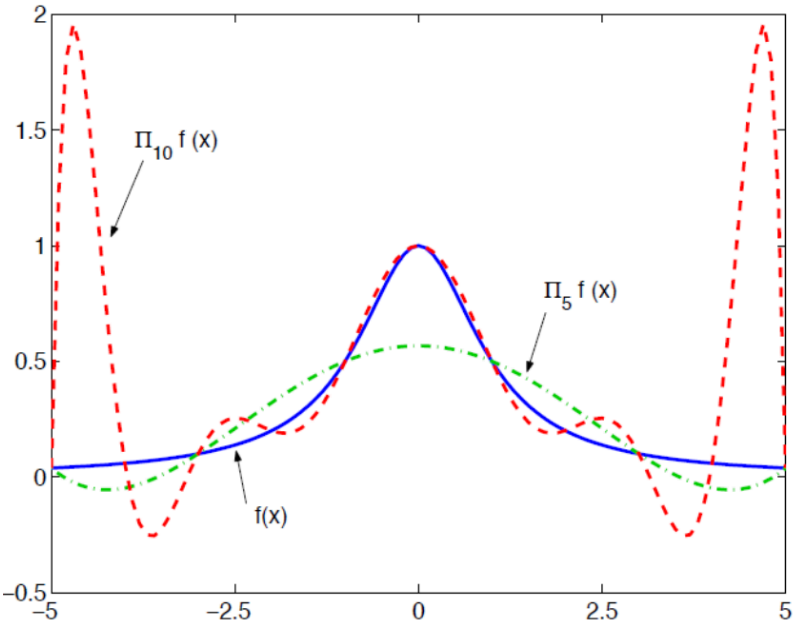
$$(\text{usando } h = H/k) \; = \frac{\max_{x\in I} \left|f^{(k+1)}(x)\right|}{4(k+1)k^k + 1} H^{k+1} \overset{H\to 0}{\to} 0$$

Questo dimostra la **convergenza dell'interpolatore Lagrangiano composito**.

6.3 Interpolazione con ubicazione specifica dei nodi

Per ovviare ai problemi di stabilita' dell'interpolatore Lagrangiano all'aumentare del numero di informazioni disponibili $n+1$, si puo' considerare di ubicare i nodi in precise posizioni che garantiscono la stabilita' al crescere di n. Questo permette di usare sempre con successo il polinomio interpolatore Lagrangiano di grado n.

In generale, queste scelte particolari prevedono l'addensamento dei nodi in prossimita' degli estremi del dominio di analisi, dove, come visto, insorgono le oscillazioni che portano a instabilita'.

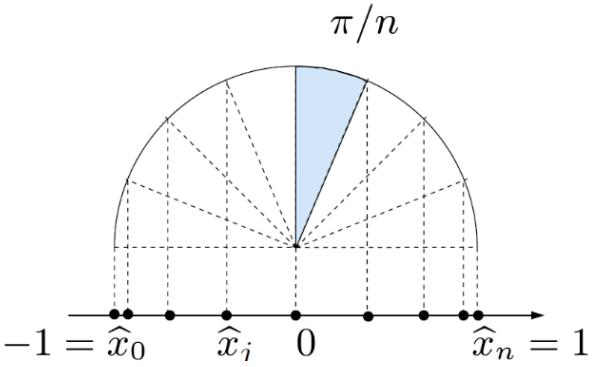


6.3.1 Nodi di Chebyshev

Consideriamo prima il caso in cui il dominio di interesse e' dato da $[-1, 1]$. Al solito, abbiamo $n + 1$ dati, ubicati in corrispondenza delle seguenti ascisse:

$$\hat{x}_j = -\cos\left(\frac{\pi j}{n}\right), \qquad j = 0, \ldots, n$$

Questi nodi sono ottenuti come proiezioni sull'asse x di punti sulla circonferenza unitaria, individuati da settori circolari ottenuti con lo stesso angolo $\frac{\pi}{n}$. Notare come essi siano effettivamente piu' addensati verso le estremita'.



Consideriamo ora l'interpolatore Lagrangiano di grado n costruito sui nodi di Chebyshev. Questo viene costruito esattamente come fatto in precedenza, a patto di prendere $\hat{x}_j$ come nodi su cui costruire gli n polinomi caratteristici di Lagrange:

$$\hat{\varphi}_k(x) = \prod_{\substack{j=0 \\ j \neq k}}^{n} \frac{x - \hat{x}_j}{\hat{x}_k - \hat{x}_j} \qquad \Pi_n^C(x) = \sum_{j=0}^{n} y_j \hat{\varphi}_j(x)$$

Esso viene indicato con $\Pi_n^C(x)$. Indicheremo l'interpolatore con $\Pi_n^C f(x)$ quando applicato a dati ottenuti dalla valutazione di una funzione f, ovvero $y_j = f(x_j)$.

6.3.2 Convergenza dell'interpolazione sui nodi di Chebyshev

Supponiamo che la funzione $f(x)$ ammetta derivata continua fino all'ordine $s+1$ compreso, ovvero $f \in C^{s+1}([-1,1])$. Allora si ha il seguente risultato di convergenza:

$$\max_{x\in[-1,1]} |f(x) - \Pi_n^C f(x)| \leq \tilde{C}\frac{1}{n^s}$$

per un'opportuna costante C:

$$\max_{x\in[-1,1]} |f(x) - \Pi_n^C f(x)| \leq C\frac{1}{n^s}$$

il precedente risultato ci dice che:

1. Se $s \geq 1$, allora si ha sempre convergenza

$$\lim_{n\to\infty} \max_{x\in[-1,1]} |f(x) - \Pi_n^C f(x)| = 0$$

2. La velocita' di convergenza e' tanto piu' alta tanto piu' e' alto s
3. Se l'ipotesi vale per ogni $s > 0$, tale velocita' e' **esponenziale**, che e' la stima migliore che si puo' ottenere, essendo piu' veloce di qualsiasi $\frac{1}{n^s}$

$$\max_{x\in[-1,1]} |f(x) - \Pi_n^C f(x)| \leq \tilde{C}e^{-n}$$

6.3.3 Nodi di Chebyshev su un intervallo generale

Nel caso in cui il dominio di interesse sia $[a,b]$, e' sufficiente mappare i nodi $\hat{x}_j$ da $[-1,1]$ ad $[a,b]$. A questo fine introduciamo la seguente mappa:

$$x = \psi(\hat{x}) = \frac{a+b}{2} + \frac{b-a}{2}\hat{x}$$

che agisce dal dominio $[-1,1]$ nella variabile $\hat{x}$ al dominio corrente $[a,b]$.

I nodi di Chebyshev su un intervallo generico $[a,b]$ sono quindi dati da

$$x_j^C = \psi(\hat{x}_j) = \frac{a+b}{2} + \frac{b-a}{2}\hat{x}_j, \qquad j = 0,\dots,n$$

I polinomi caratteristici di Lagrange seguono al solito modo:

$$\varphi_k^C(x) = \prod_{\substack{j=0 \\ j\neq k}}^{n} \frac{x - x_j^C}{x_k^C - x_j^C}$$

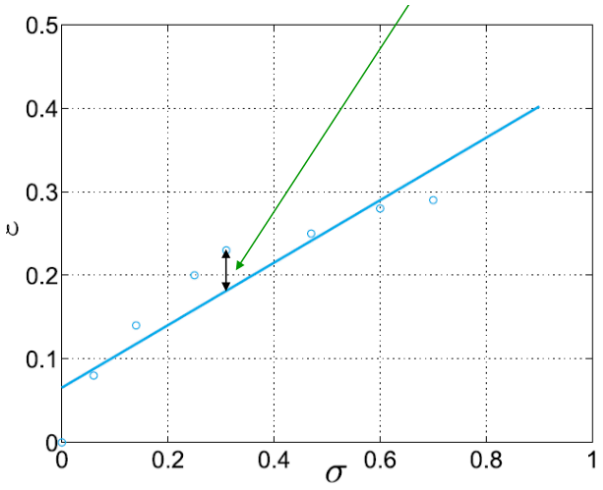
6.4 Metodo dei Minimi Quadrati

Supponiamo al solito di conoscere $n+1$ coppie di dati $(x_i, y_i), i = 0,\dots,n$, e di cercare un approssimante di basso grado $m \ll n$, pero' globale, cioe' su tutto l'intervallo $[a,b]$ che contiene i nodi x_i.

Questo e', ad esempio, il caso in cui si voglia fare un'**estrapolazione dei dati** a disposizione, cioe' una "previsione" al di fuori dell'intervallo $[a, b]$. Nessuna delle strategie viste in precedenza va bene. Infatti:

- L'interpolazione Lagrangiana potrebbe andare incontro al fenomeno di Runge, essendo i dati in numero elevato.
- L'interpolazione Lagrangiana composita terrebbe conto solo degli ultimi $k + 1$ dati, quindi non utilizzerebbe tutta la storia a disposizione.
- L'interpolazione su nodi di Chebyshev non e' in generale possibile, perche' i dati a disposizione sono equispaziati.

Si cerca quindi un polinomio globale di basso grado $m \ll n$ (in modo che sia stabile) che verra' poi valutato all'interno o al di fuori dell'intervallo $[a, b]$. Per fare questo devo pero' **abbandonare il vincolo di interpolazione**.



Dobbiamo introdurre il concetto di **approssimazione**. In particolare, si cerca il polinomio di grado m che mediamente minimizza lo **scarto quadratico medio**, ovvero la somma dei quadrati degli errori nei nodi. Questa approssimazione e' chiamata **ai minimi quadrati**.

Introduciamo lo spazio dei polinomi di grado m:

$$\mathbb{P}_m = \{p_m : \mathbb{R} \to \mathbb{R} : p_m(x) = b_0 + b_1 x + \cdots + b_m x_m\}$$

Allora, il problema ai minimi quadrati consiste nel cercare il polinomio q tale che

$$\sum_{i=0}^{n} [y_i - q(x_i)]^2 \leq \sum_{i=0}^{n} [y_i - p_m(x_i)]^2, \qquad \forall p_m \in \mathbb{P}_m$$

Dobbiamo quindi determinare $m + 1$ coefficienti b_j per individuare il polinomio di miglior approssimazione nel senso dei minimi quadrati. Indicheremo con a_j i valori di b_j che ci danno il minimo del precedente problema:

$$q(x) = a_0 + a_1 x + \ldots a_m x_m$$

6.4.1 Retta di regressione

Nel caso in cui si cerchi la retta di miglior approssimazione nel senso dei minimi quadrati, si parla di retta di regressione (ovvero $m = 1$). Il problema di determinare b_0 e b_1 si imposta nel seguente modo. Introduciamo la funzione:

$$\Phi(b_0, b_1) = \sum_{i=0}^{n} [y_i - p_m(x_i)]^2 = \sum_{i=0}^{n} [y_i - (b_0 + b_1 x_i)]^2$$

Il minimo di questa funzione al variare di b_0 e b_1 mi dara' proprio la retta di regressione:

$$\Phi(a_0, a_i) = \min_{(b_0, b_1)} \Phi(b_0, b_1)$$

$$q(x) = a_0 + a_1 x$$

Il punto di minimo e' quindi dato imponendo:

$$\frac{\partial \Phi}{\partial b_0}(a_0, a_1) = 0, \qquad \frac{\partial \Phi}{\partial b_1}(a_0, a_1) = 0$$

Le precedenti portano al seguente sistema in 2 equazioni e incognite:

$$\begin{cases} \sum_{i=0}^{n} [a_0 + a_1 x_i - y_i] = 0 \\ \sum_{i=0}^{n} [a_0 x_i + a_1 x_i^2 - x_i y_i] = 0 \end{cases}$$

Tale sistema ha per soluzione:

$$a_0 = \frac{1}{D} \left[\sum_{i=0}^{n} y_i \sum_{j=0}^{n} x_j^2 - \sum_{j=0}^{n} x_j \sum_{i=0}^{n} x_i y_i \right]$$

$$a_1 = \frac{1}{D} \left[(n+1) \sum_{i=0}^{n} x_i y_i - \sum_{j=0}^{n} x_j \sum_{i=0}^{n} y_i \right]$$

con $D = (n+1) \sum_{i=0}^{n} x_i^2 - \left(\sum_{i=0}^{n} x_i \right)$.

6.4.2 Caso generale

In generale, abbiamo la seguente funzione da minimizzare:

$$\Phi(b_0, b_1, \ldots, b_m) = \sum_{i=0}^{n} \left[y_i - (b_0 + b_1 x + \cdots + b_m x_i^m) \right]^2$$

Per trovare gli $m + 1$ coefficienti a_j che determinano il polinomio di miglior approssimazione (nel senso dei minimi quadrati):

$$q(x) = a_0 + a_1 x + \cdots + a_m x^m$$

Cio' porta al seguente sistema lineare in $m + 1$ equazioni e $m + 1$ incognite:

$$\frac{\partial \Phi}{\partial b_j} = 0, \qquad j = 0, \ldots, m$$

7 Integrazione Numerica - Formule di Quadratura

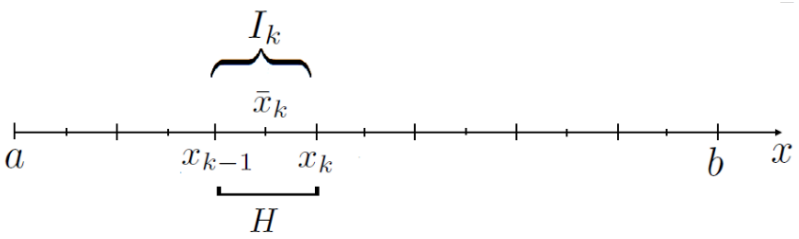
Consideriamo l'integrale

$$I(f) = \int_a^b f(x)dx$$

Suddividiamo l'intervallo $[a, b]$ in M intervalli I_k di ampiezza costante H:

$$[x_{k-1}, x_k],\ k = 1, \dots, M \qquad x_k = a + kH,\ k = 0, \dots, M$$

Introduciamo inoltre su ogni intervallo i punti medi $\bar{x} = \frac{x_{k-1}+x_k}{2}$



Consideriamo una formula di quadratura per il calcolo approssimato di $I(f)$, e sia $I_H(f)$ il valore approssimato ottenuto. Si dice che la formula di quadratura e' di **ordine** p se l'errore soddisfa la seguente stima:

$$E_H = |I(f) - I_H(f)| \leq CH^p$$

Si dice inoltre che la formula di quadratura ha **grado di esattezza pari ad** r se essa risulta esatta quando applicata ai **polinomi di grado minore o uguale ad** r, cioe' se

$$E_H = |I(f) - I_H(f)| = 0, \qquad \forall f \in \mathbb{P}^r(a, b)$$

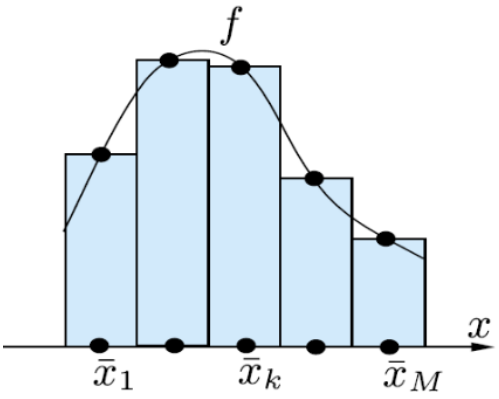
7.1 Formula del punto medio composita

Consideriamo la proprieta' di additivita' dell'integrale, cioe'

$$I(f) = \sum_{k=1}^{M} \int_{I_k} f(x)dx$$

Partendo da questa formula esatta, possiamo approssimare il valore dell'integrale su ogni intervallo, e poi sommare i contributi.

L'idea della formula del **punto medio composita** si basa sull'approssimare il valore di $\int_{I_k} f(x)dx$ con l'area del rettangolo che ha per altezza $f(\bar{x}_k)$.



Poiche' la base di questi rettangoli e' sempre pari ad H, avremo quindi la seguente approssimazione per $I(f)$:

$$I_{pm}(f) = H \sum_{k=1}^{M} f(\bar{x}_k)$$

Abbiamo la seguente stima dell'errore:

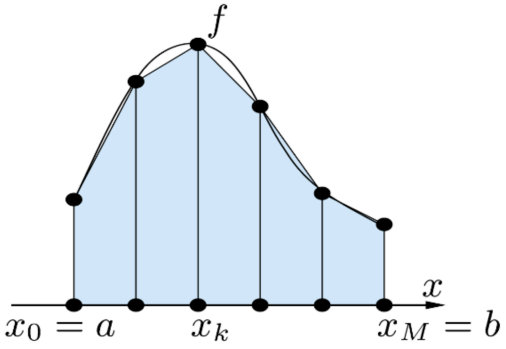
$$|I(f) - I_{pm}(f)| \leq \max_x |f''(x)| \frac{b-a}{24} H^2$$

Da questa stima si evince che:

1. La formula del punto medio composita e' di **ordine 2**.
2. La formula del punto medio composita ha **grado di esattezza pari a 1**. Infatti, ricordando che stiamo assumendo f con derivata seconda continua, si ha che $f'' = 0$ in ogni x se f e' una retta, cioe' per $r = 1$.

7.2 Formula dei trapezi composita

Partiamo sempre dalla proprieta' di additivita' dell'integrale. Questa volta approssimiamo l'aria sottesa da f nell'intervallo I_k considerando il trapezio costruito sui punti x_{k-1} e x_k e sulle corrispondenti ordinate.



Si ha quindi la formula dei **trapezi composita**:

$$I_{tr}(f) = \frac{H}{2} \sum_{k=1}^{M} (f(x_{k-1}) + f(x_k))$$

per comodita' di implementazione si scrive:

$$I_{tr}(f) = \frac{H}{2}(f(a) + f(b)) + H\sum_{k=1}^{M-1} f(x_k)$$

Notiamo che la formula dei trapezi composita equivale all'integrale esatto dell'interpolatore Lagrangiano composito di grado 1:

$$I_{tr}(f) = \int_a^b \Pi_1^H f(x)dx$$

Su ogni intervallo I_k possiamo quindi considerare L'errore di interpolazione Lagrangiana, da cui si ottiene la seguente stima dell'errore per la formula dei trapezi composita:

$$|I(f) - I_{tr}(f)| \leq \frac{1}{12}\max_x |f''(x)|(b-a)H^2$$

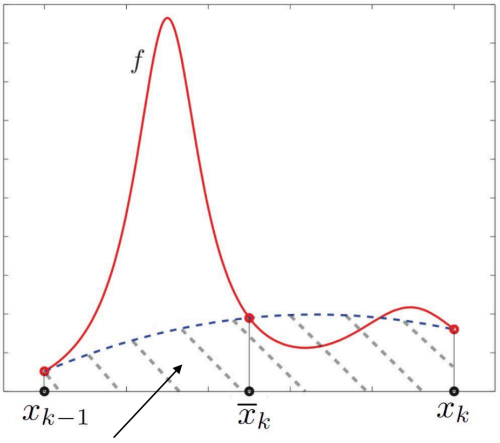
Da questa stima si evince che:

1. La formula dei trapezi composita e' di **ordine 2**.
2. La formula dei trapezi composita ha **grado di esattezza pari a 1**.

Quindi, ordine e grado di esattezza sono uguali a quelli del punto medio. La scelta di uno o dell'altro metodo risiede principalmente sull'avere a disposizione le coordinate dei punti medi o dei punti estremi degli intervalli.

Formula di Simpson composita

Partiamo sempre dalla proprieta' di additivita' dell'integrale. Questa volta approssimiamo l'area sottesa da f nell'intervallo I_k considerando l'area sottesa dalla parabola che interpola x_{k-1}, $\bar{x}_k$ e x_k.



Integrando esattamente tali parabole su ogni intervallo I_k, si ottiene la formula di **Simpson composita:**

$$I_{sim}(f) = \frac{H}{6}\sum_{k=1}^{M}(f(x_{k-1}) + 4f(\bar{x}_k) + f(x_k))$$

Notiamo che, per costruzione, la formula di Simpson composita equivale all'**integrale esatto dell'interpolatore Lagrangiano composito di grado 2**:

$$I_{sim}(f) = \int_a^b \Pi_2^H f(x)dx$$

Si puo' mostrare che in questo caso vale il seguente risultato per l'errore, assumendo che f abbia derivata quarta continua su $[a, b]$:

$$|I(f) - I_{sim}(f)| \leq \frac{b-a}{2800} \max_x |f^{(iv)}(x)| H^4$$

Da questa stima si evince che:

1. La formula di Simpson composita e' di **ordine 4**.
2. La formula di Simpson composita ha **grado di esattezza pari a 3**.

Tale formula, quindi, non integra esattamente soltanto i polinomi che sono globalmente di grado $r = 1$ (rette), e $r = 2$ (parabole), ma anche $r = 3$ (cubiche).

Approssimazione Numerica di Equazioni Differenziali Ordinarie

Problema di Cauchy

Il problema di Cauchy e' un problema di prim'ordine, cioe' solo con derivata prima della incognita.

Sia $I = [t_0, T]$ l'intervallo temporale di interesse. Trovare la funzione vettoriale $\boldsymbol{y} : I \to \mathbb{R}^n$ che soddisfa

$$\begin{cases} \boldsymbol{y}'(t) = \boldsymbol{f}(t, \boldsymbol{y}(t)) \\ \boldsymbol{y}(t_0) = \boldsymbol{y}_0 \end{cases}$$

con $\boldsymbol{f} : I \times \mathbb{R}^n \to \mathbb{R}^n$ funzione vettoriale assegnata (in generale non lineare in y)

Problema di Cauchy Scalare

Consideriamo ora il problema di Cauchy scalare (cioe' con $n = 1$).

Sia $I = [t_0, T]$ l'intervallo temporale di interesse e consideriamo il seguente problema:

$$\text{trovare } y : I \subset \mathbb{R} \to \mathbb{R} \text{ tale che}$$

$$\begin{cases} y'(t) = f(t, y(t)) & \forall t \in I \\ y(t_0) = y_0 \end{cases}$$

con $f : I \times \mathbb{R} \to \mathbb{R}$ funzione assegnata (in generale non lineare in y).

Proposizione

Supponiamo che la funzione $f(t, y)$ sia

1. limitata e continua rispetto ad entrambi gli argomenti;
2. lipschitziana rispetto al secondo argomento, ossia esista una costante L positiva (detta costante di Lipschitz) tale che

$$|f(t, y_1) - f(t, y_2)| \leq L|y_1 - y_2| \qquad \forall t \in I,\ \forall y_1, y_2 \in \mathbb{R}$$

   Allora la soluzione del problema di Cauchy esiste, e' unica ed e' di classe C^1 su I.

A

Fino a pag 30