

**CAP. 5 – METODI PER LA SOLUZIONE DEI SISTEMI DI EQUAZIONI LINEARI****5.1 Introduzione**

Nel capitolo precedente abbiamo visto che per calcolare il campo di spostamenti $\{D\}$ è necessario calcolare le matrici di rigidezza elementari, assemblarle nella matrice di rigidezza globale $[K]$, inserirvi le condizioni al contorno, calcolare il vettore delle forze $\{F\}$ e finalmente risolvere il seguente sistema di equazioni lineari:

$$[K] \cdot \{D\} = \{F\} \quad [5.1.1]$$

Il termine noto $\{F\}$ rappresenta, in generale, m condizioni di carico e, di conseguenza, il vettore degli spostamenti diventa una matrice formata da m colonne ed il sistema deve essere risolto m volte. I metodi utilizzati per realizzare questo obiettivo si dividono in diretti ed iterativi, ma solitamente i programmi di calcolo strutturale utilizzano solo i primi, perché più efficienti dei secondi. Tuttavia, quando si ha a che fare con problemi di grandissime dimensioni che implicano una notevole occupazione di memoria, i metodi iterativi diventano insostituibili. In questi casi la matrice di rigidezza viene memorizzata in forma sparsa e l'operazione centrale, ripetuta ad ogni iterazione, prevede la moltiplicazione della matrice per un vettore. Poiché questa operazione è facilmente parallelizzabile, diventano molto interessanti i calcolatori a più processori. Nel caso estremo, in cui la matrice di rigidezza globale sia talmente grande da non poter essere memorizzata, si ricorre al prodotto delle singole matrici di rigidezza elementari per il vettore delle incognite: la procedura è meno efficiente delle precedenti, ma consente di ottenere la soluzione del problema.

Nel seguito verranno descritti i metodi utilizzati di solito nei codici agli elementi finiti per la meccanica strutturale.

5.2 Metodi diretti: la fattorizzazione LU

Si ipotizzi di scrivere la matrice $[K]$ di dimensione $N \times N$ come prodotto di due matrici:

$$[L][U] = [K] \quad [5.2.1]$$

dove $[L]$ è una matrice triangolare inferiore (i suoi coefficienti diversi da zero si trovano solo sulla diagonale e ad di sotto di essa) e $[U]$ è una matrice triangolare superiore (i suoi coefficienti diversi da zero si trovano solo sulla diagonale e sopra di essa). Per esempio nel caso in cui la matrice $[K]$ avesse dimensione 4×4 , l'equazione (5.2.1) potrebbe essere espansa nel modo seguente:

$$\begin{bmatrix} L_{11} & 0 & 0 & 0 \\ L_{21} & L_{22} & 0 & 0 \\ L_{31} & L_{32} & L_{33} & 0 \\ L_{41} & L_{42} & L_{43} & L_{44} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} & U_{13} & U_{14} \\ 0 & U_{22} & U_{23} & U_{24} \\ 0 & 0 & U_{33} & U_{34} \\ 0 & 0 & 0 & U_{44} \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & K_{13} & K_{14} \\ K_{21} & K_{22} & K_{23} & K_{24} \\ K_{31} & K_{32} & K_{33} & K_{34} \\ K_{41} & K_{42} & K_{43} & K_{44} \end{bmatrix} \quad [5.2.2]$$

Per risolvere il sistema di equazioni lineari:

$$[K]\{x\} = \{F\} \quad [5.2.3]$$

è possibile usare la decomposizione (5.2.1):

$$[K]\{x\} = [L][U]\{x\} = [L]([U]\{x\}) = [L]\{y\} = \{F\} \quad [5.2.4]$$

Per iniziare si calcola il vettore $\{y\}$:

$$[L]\{y\} = \{F\}, \quad \text{per esteso} \quad \begin{bmatrix} L_{11} & 0 & 0 & 0 \\ L_{21} & L_{22} & 0 & 0 \\ L_{31} & L_{32} & L_{33} & 0 \\ L_{41} & L_{42} & L_{43} & L_{44} \end{bmatrix} \begin{Bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{Bmatrix} = \begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{Bmatrix} \quad [5.2.5]$$

e quindi si risolve il sistema:



$$[U]\{x\} = \{y\}, \quad \text{per esteso} \quad \begin{bmatrix} U_{11} & U_{12} & U_{13} & U_{14} \\ 0 & U_{22} & U_{23} & U_{24} \\ 0 & 0 & U_{33} & U_{34} \\ 0 & 0 & 0 & U_{44} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{Bmatrix} \quad [5.2.6]$$

Perché dividere un solo sistema di equazioni lineari in due sistemi da risolvere in due passi successivi? Il vantaggio è che la soluzione di un sistema triangolare è molto semplice, quindi l'eq.(5.2.5) può essere risolta con la seguente sostituzione in avanti (*forward substitution*):

$$\begin{aligned} L_{11}y_1 &= F_1 & \text{da cui: } y_1 &= \frac{F_1}{L_{11}} \\ L_{21}y_1 + L_{22}y_2 &= F_2 & \text{da cui: } y_2 &= \frac{F_2 - L_{21}y_1}{L_{22}} \\ \dots\dots\dots \\ L_{i1}y_1 + L_{i2}y_2 + \dots + L_{ii}y_i &= F_i & \text{da cui: } y_i &= \frac{1}{L_{ii}} (F_i - \sum_{j=1}^{i-1} L_{ij}y_j) \quad \text{con } i = 2, 3, \dots, N \end{aligned} \quad [5.2.7]$$

mentre il sistema (5.2.6) può essere risolto con la sostituzione all'indietro (*back substitution*):

$$\begin{aligned} U_{44}x_4 &= y_4 & \text{da cui: } x_4 &= \frac{y_4}{U_{44}} \\ U_{33}x_3 + U_{34}x_4 &= y_3 & \text{da cui: } x_3 &= \frac{y_3 - U_{34}x_4}{U_{33}} \\ \dots\dots\dots \\ U_{ii}x_i + \dots + U_{i4}x_4 &= y_i & \text{da cui: } x_i &= \frac{1}{U_{ii}} (y_i - \sum_{j=i+1}^N U_{ij}x_j) \quad \text{con } i = N - 1, N - 2, \dots, 1 \end{aligned} \quad [5.2.8]$$

Una volta che la matrice $[K]$ è stata fattorizzata, è possibile risolvere il sistema con numerosi termini noti $\{F\}$, cioè con diversi vettori colonna, presi uno alla volta. Si tratta di un importante vantaggio rispetto ai metodi iterativi.

5.2.1 Esecuzione della decomposizione $[L][U]$

Per capire come calcolare i coefficienti delle matrici $[L]$ e $[U]$ scriviamo per esteso il prodotto della i -esima riga della matrice $[L]$ per la j -esima colonna della matrice $[U]$:

$$L_{i1}U_{1j} + L_{i2}U_{2j} + \dots = \sum_{k=1}^{\min(i,j)} L_{ik}U_{kj} = K_{ij} \quad [5.2.9]$$

Il numero di coefficienti presenti nella sommatoria dipende dal valore più piccolo tra i (la riga) e j (la colonna). Abbiamo infatti i tre casi:

$$i < j; \text{ in tal caso abbiamo: } L_{i1}U_{1j} + L_{i2}U_{2j} + \dots + L_{ii}U_{ij} = K_{ij} \quad [5.2.10]$$

$$i = j; \text{ in tal caso abbiamo: } L_{i1}U_{1j} + L_{i2}U_{2j} + \dots + L_{ii}U_{jj} = K_{ij} \quad [5.2.11]$$

$$i > j; \text{ in tal caso abbiamo: } L_{i1}U_{1j} + L_{i2}U_{2j} + \dots + L_{ij}U_{jj} = K_{ij} \quad [5.2.12]$$

Le equazioni (5.2.10), (5.2.11) e (5.2.12) sono in totale N^2 (una per ogni coefficiente della matrice $[K]$), mentre le incognite da determinare (cioè i coefficienti delle due matrici $[L]$ e $[U]$), sono in totale $2 \left[\frac{N(N+1)}{2} \right] = N^2 + N$. Poiché il numero di incognite è superiore al numero di equazioni, possiamo indicarne N in modo arbitrario e calcolare le rimanenti. Per esempio possiamo inizializzare nel modo seguente la diagonale della matrice $[L]$:

$$L_{ii} = 1 \quad i = 1, \dots, N \quad [5.2.13]$$

Per il calcolo dei coefficienti L_{ij} e U_{ij} , l'*algoritmo di Crout* risolve l'insieme di $N^2 + N$ equazioni (5.2.10), (5.2.11) e (5.2.12) ordinandole nel modo seguente:

- Si pone $L_{ii} = 1 \quad i = 1, \dots, N$ vedi l'eq.(5.2.13)
- Per $j = 1, 2, \dots, N$ si esegue la seguente procedura:



- $U_{1j} = K_{1j}$
- Per $i = 2, \dots, j$ si usano le (5.2.10) e (5.2.11) per calcolare U_{ij} , cioè:

$$U_{ij} = K_{ij} - L_{i1}U_{1j} - L_{i2}U_{2j} + \dots = K_{ij} - \sum_{k=1}^{i-1} L_{ik}U_{kj} \quad [5.2.14]$$

- Per $i = j + 1, j + 2, \dots, N$ si usa l'eq. (5.2.12) per calcolare L_{ij} , cioè:

$$L_{ij} = \frac{1}{U_{jj}} (K_{ij} - \sum_{k=1}^{j-1} L_{ik}U_{kj}) \quad [5.2.15]$$

Il numero di operazioni (una somma ed un prodotto) necessarie alla fattorizzazione $[L][U]$ è dell'ordine $\mathcal{O}\left(\frac{N^3}{3}\right)$. Eseguendo alcune iterazioni del precedente algoritmo, si nota che i coefficienti L_{ik} e U_{kj} che compaiono tra i termini noti delle equazioni (5.2.14) e (5.2.15) sono già disponibili nel momento in cui servono. Si nota anche che ogni coefficiente della matrice $[K]$ viene usato una sola volta e non rientra nella procedura. Ciò significa che i corrispondenti coefficienti L_{ij} e U_{ij} possono essere memorizzati nella locazione di memoria prima occupata da K_{ij} . I coefficienti unitari L_{ii} appartenenti alla diagonale della matrice $[L]$ non vengono memorizzati. In poche parole, il metodo di Crout sostituisce la matrice $[K]$ con la seguente matrice:

$$\begin{bmatrix} U_{11} & U_{12} & U_{13} & U_{14} \\ L_{21} & U_{22} & U_{23} & U_{24} \\ L_{31} & L_{32} & U_{33} & U_{34} \\ L_{41} & L_{42} & L_{43} & U_{44} \end{bmatrix} \quad [5.2.16]$$

Nel caso di una matrice 4×4 abbiamo:

$$L_{11} = L_{22} = L_{33} = L_{44} = 1$$

- Per $j = 1$
 - Per $i = 1$ $U_{11} = K_{11}$
 - Per $i = 2, 3, 4$ $L_{21} = \frac{K_{21}}{U_{11}}$; $L_{31} = \frac{K_{31}}{U_{11}}$; $L_{41} = \frac{K_{41}}{U_{11}}$
- Per $j = 2$
 - Per $i = 1, 2$ $U_{12} = K_{12}$; $U_{22} = K_{22} - L_{21}U_{12}$
 - Per $i = 3, 4$ $L_{32} = \frac{1}{U_{22}} (K_{32} - L_{31}U_{12})$; $L_{42} = \frac{1}{U_{22}} (K_{42} - L_{41}U_{12})$
- Per $j = 3$
 - Per $i = 1, 2, 3$ $U_{13} = K_{13}$; $U_{23} = K_{23} - L_{21}U_{13}$; $U_{33} = K_{33} - L_{31}U_{13} - L_{32}U_{23}$
 - Per $i = 4$ $L_{43} = \frac{1}{U_{33}} (K_{43} - L_{41}U_{13} - L_{42}U_{23})$
- Per $j = 4$
 - Per $i = 1, 2, 3, 4$ $U_{14} = K_{14}$; $U_{24} = K_{24} - L_{21}U_{14}$; $U_{34} = K_{34} - L_{31}U_{14} - L_{32}U_{24}$
 - $U_{44} = K_{44} - L_{41}U_{14} - L_{42}U_{24} - L_{43}U_{34}$

Se la matrice $[K]$ rappresenta la rigidezza di un modello agli elementi finiti allora risulta sempre simmetrica (come si può dimostrare usando il teorema di Betti), ed è possibile risparmiare spazio di memoria registrandone solo il triangolo superiore. Inoltre essendo i suoi coefficienti concentrati intorno alla diagonale principale, se ne possono registrare solo $N \times B$, dove B indica la larghezza della semi banda, collocando nella prima colonna dell'area di memoria la diagonale di $[K]$ e poi, a seguire, i termini alla sua destra. Ma l'algoritmo di Crout precedentemente descritto richiede sia la matrice triangolare inferiore $[L]$ che la matrice triangolare superiore



$[U]$, quindi richiede un'occupazione di memoria pari a $N \times (2B - 1)$. In seguito alla fattorizzazione $[L][U]$ la matrice di rigidezza $[K]$ viene distrutta e sostituita dai suoi fattori $[L]$ e $[U]$.

Se però al momento della fattorizzazione sono disponibili tutti i vettori delle forze $\{F\}$, è possibile evitare la memorizzazione della matrice $[L]$ eseguendo la sostituzione in avanti (*forward substitution*) descritta dall'eq.(5.2.7) e calcolare i vettori $\{y\}$ che andranno ad occupare la locazione di memoria dei vettori delle forze $\{F\}$. La sostituzione all'indietro (la *back substitution*) può essere eseguita in un secondo momento.

Osserviamo un'altra importante proprietà della fattorizzazione $[L][U]$ che può consentire un ulteriore risparmio di memoria. Immaginiamo che la matrice di rigidezza $[K]$ assuma la forma seguente:

$$\begin{bmatrix} K_{11} & K_{12} & K_{13} & K_{14} \\ K_{21} & K_{22} & K_{23} & K_{24} \\ K_{31} & K_{32} & K_{33} & K_{34} \\ K_{41} & K_{42} & K_{43} & K_{44} \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & K_{13} & 0 \\ K_{21} & K_{22} & 0 & 0 \\ K_{31} & 0 & K_{33} & K_{34} \\ 0 & 0 & K_{43} & K_{44} \end{bmatrix} \quad [5.2.17]$$

dove abbiamo posto $K_{23} = K_{14} = K_{24} = K_{32} = K_{41} = K_{42} = 0$.

La fattorizzazione $[L][U]$ di tale matrice è la seguente (in rosso i coefficienti nulli):

- Per $j = 1$
 - Per $i = 1$ $U_{11} = K_{11}$
 - Per $i = 2,3,4$ $L_{21} = \frac{K_{21}}{U_{11}}$; $L_{31} = \frac{K_{31}}{U_{11}}$; $L_{41} = \frac{K_{41}}{U_{11}} = 0$
- Per $j = 2$
 - Per $i = 1,2$ $U_{12} = K_{12}$; $U_{22} = K_{22} - L_{21}U_{12}$
 - Per $i = 3,4$ $L_{32} = \frac{1}{U_{22}}(K_{32} - L_{31}U_{12}) = \frac{-L_{31}U_{12}}{U_{22}}$; $L_{42} = \frac{1}{U_{22}}(K_{42} - L_{41}U_{12}) = 0$
- Per $j = 3$
 - Per $i = 1,2,3$ $U_{13} = K_{13}$; $U_{23} = K_{23} - L_{21}U_{13} = -L_{21}U_{13}$;
 $U_{33} = K_{33} - L_{31}U_{13} - L_{32}U_{23}$
 - Per $i = 4$ $L_{43} = \frac{1}{U_{33}}(K_{43} - L_{41}U_{13} - L_{42}U_{23}) = \frac{K_{43}}{U_{33}}$
- Per $j = 4$
 - Per $i = 1,2,3,4$ $U_{14} = K_{14} = 0$; $U_{24} = K_{24} - L_{21}U_{14} = 0$
 $U_{34} = K_{34} - L_{31}U_{14} - L_{32}U_{24} = K_{34}$
 $U_{44} = K_{44} - L_{41}U_{14} - L_{42}U_{24} - L_{43}U_{34} = K_{44} - L_{43}U_{34}$

In sintesi abbiamo:

$$\begin{bmatrix} K_{11} & K_{12} & K_{13} & 0 \\ K_{21} & K_{22} & 0 & 0 \\ K_{31} & 0 & K_{33} & K_{34} \\ 0 & 0 & K_{43} & K_{44} \end{bmatrix} = \begin{bmatrix} U_{11} & U_{12} & U_{13} & 0 \\ L_{21} & U_{22} & U_{23} & 0 \\ L_{31} & L_{32} & U_{33} & U_{34} \\ 0 & 0 & L_{43} & U_{44} \end{bmatrix}$$

Osserviamo che la locazione di memoria occupata dal coefficiente K_{23} (in origine nullo) viene occupata da un coefficiente $U_{23} \neq 0$ e quindi bisogna allocare la memoria necessaria per poterlo salvare; viceversa le posizioni occupate dai coefficienti K_{14} e K_{24} (in origine nulli) continuano ad essere occupate da coefficienti nulli (U_{14} e U_{24}) e quindi è possibile evitare di allocare la memoria relativa. Questo comportamento è di carattere generale, cioè non legato all'esempio preso in esame: per risparmiare memoria è quindi possibile allocare solo lo



spazio necessario a contenere la skyline della matrice, cioè colonna dopo colonna, solo i coefficienti compresi tra il termine diagonale e quello più alto diverso da zero appartenente alla stessa colonna. Nel nostro esempio:

$$\{K\} = \{K_{11} \quad K_{12} \quad K_{22} \quad K_{13} \quad 0 \quad K_{33} \quad K_{34} \quad K_{44}\}$$

E' inoltre necessario un vettore di numeri interi necessario ad individuare, all'interno del vettore $[K]$, la posizione dei coefficienti appartenenti alla diagonale della matrice originale; nel caso del esempio precedente:

$$\{Kadrs\} = \{1 \quad 3 \quad 6 \quad 8\}$$

Quindi l' i -esimo coefficiente diagonale K_{ii} si trova in posizione $Kadrs(i)$; per esempio il coefficiente K_{33} si trova in posizione $Kadrs(3) = 6$.

La fattorizzazione però comporta il riempimento della matrice nelle locazioni occupate da coefficienti nulli in posizione inferiore alla skyline, per esempio il coefficiente U_{23} nell'esempio precedente. Se la matrice è molto grande e sparsa, è inutile memorizzarla in forma sparsa e poi utilizzare la fattorizzazione $[L][U]$ perché al termine di tale operazione la matrice $[U]$ risulterà comunque molto più densa. In tali casi è consigliabile utilizzare un metodo iterativo che non modifichi la matrice di partenza $[K]$ messa in memoria in forma sparsa.

5.2.2 Fattorizzazione di Cholesky $[L][L]^T$

Se una matrice $[K]$ risulta simmetrica e definita positiva allora è possibile una fattorizzazione triangolare molto efficiente. *Definita positiva* significa che per qualsiasi vettore $\{v\}$ è valida la seguente relazione:

$$\{v\}^T [K] \{v\} > 0 \quad [5.2.18]$$

Benché le matrici simmetriche e definite positive siano un caso molto speciale di matrici, si trovano in numerose applicazioni per cui la fattorizzazione di Cholesky è una buona opportunità in quanto è più veloce di almeno un fattore due rispetto ai metodi alternativi usati per la soluzione dei sistemi di equazioni lineari.

Invece di fattorizzare la matrice $[K]$ in una matrice triangolare inferiore $[L]$ ed in una triangolare superiore $[U]$, la decomposizione di Cholesky costruisce una matrice triangolare inferiore $[L]$ che, moltiplicata per la sua trasposta, è uguale la matrice originale. In altre parole:

$$[L][L]^T = [K] \quad [5.2.19]$$

Scrivendo l'eq.(5.2.19) in modo esteso per una matrice 4×4 abbiamo:

$$\begin{bmatrix} L_{11} & 0 & 0 & 0 \\ L_{21} & L_{22} & 0 & 0 \\ L_{31} & L_{32} & L_{33} & 0 \\ L_{41} & L_{42} & L_{43} & L_{44} \end{bmatrix} \begin{bmatrix} L_{11} & L_{21} & L_{31} & L_{41} \\ 0 & L_{22} & L_{32} & L_{42} \\ 0 & 0 & L_{33} & L_{43} \\ 0 & 0 & 0 & L_{44} \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & K_{13} & K_{14} \\ K_{21} & K_{22} & K_{23} & K_{24} \\ K_{31} & K_{32} & K_{33} & K_{34} \\ K_{41} & K_{42} & K_{43} & K_{44} \end{bmatrix}$$

Dalla prima riga abbiamo:

$$L_{11}^2 = K_{11} \quad ; \quad L_{11}L_{21} = K_{12} \quad ; \quad L_{11}L_{31} = K_{13} \quad ; \quad L_{11}L_{41} = K_{14}$$

da cui:

$$L_{11} = \sqrt{K_{11}} \quad ; \quad L_{21} = \frac{K_{12}}{L_{11}} \quad ; \quad L_{31} = \frac{K_{13}}{L_{11}} \quad ; \quad L_{41} = \frac{K_{14}}{L_{11}}$$

Dalla seconda riga abbiamo:

$$L_{21}^2 + L_{22}^2 = K_{22} \quad \text{da cui:} \quad L_{22} = \sqrt{K_{22} - L_{21}^2}$$

$$L_{21}L_{31} + L_{22}L_{32} = K_{23} \quad \text{da cui:} \quad L_{32} = \frac{K_{23} - L_{21}L_{31}}{L_{22}}$$

$$L_{21}L_{41} + L_{22}L_{42} = K_{24} \quad \text{da cui:} \quad L_{42} = \frac{K_{24} - L_{21}L_{41}}{L_{22}}$$

Dalla terza riga abbiamo:



$$L_{31}^2 + L_{32}^2 + L_{33}^2 = K_{33} \quad \text{da cui:} \quad L_{33} = \sqrt{K_{33} - L_{31}^2 - L_{32}^2}$$

$$L_{31}L_{41} + L_{32}L_{42} + L_{33}L_{43} = K_{34} \quad \text{da cui:} \quad L_{43} = \frac{K_{34} - L_{31}L_{41} - L_{32}L_{42}}{L_{33}}$$

Generalizzando abbiamo:

$$L_{ii} = \sqrt{K_{ii} - \sum_{k=1}^{i-1} L_{ik}^2} \quad [5.2.20]$$

e

$$L_{ji} = \frac{K_{ij} - \sum_{k=1}^{i-1} L_{ik}L_{jk}}{L_{ii}} \quad j = i + 1, i + 2, \dots, N \quad [5.2.21]$$

Applicando le eq.(5.2.20) e (5.2.21) nell'ordine $i = 1, 2, \dots, N$, si può osservare che i coefficienti della matrice $[L]$ che compaiono a destra del segno di uguaglianza sono già noti quando servono. Inoltre si utilizzano solo i coefficienti della matrice $[K]$ per i quali $j \geq i$.

Le operazioni necessari sono $\frac{N^3}{6}$ per l'iterazione interna (ogni operazione comprende una moltiplicazione ed una sottrazione) oltre ad N radici quadrate. Come detto il numero di operazioni è circa la metà di quello richiesto per la migliore decomposizione di tipo $[L][U]$ (che non sfrutta la simmetria della matrice $[K]$).

Un'altra proprietà interessante della decomposizione di Cholesky è che non è necessaria la ricerca del migliore pivot in quanto l'algoritmo è estremamente stabile.

5.2.3 Fattorizzazione $[L][D][L]^T$

Talvolta la matrice di rigidezza può contenere dei coefficienti nulli nella diagonale. Questo capita per esempio inserendo i vincoli multi grado di libertà per mezzo dei moltiplicatori di Lagrange. In tal caso non è possibile utilizzare né la fattorizzazione $[L][U]$ né quella di Cholesky perché i due metodi prevedono la divisione di un numeratore per i coefficienti appartenenti alla diagonale (vedi le eq. (5.2.15) e (5.2.21)). In tal caso si può ricorrere alla fattorizzazione $[L][D][L]^T$.

Immaginiamo che la matrice $[K]$ sia memorizzata in forma skyline. Utilizziamo un vettore di numeri interi di nome $\{skyline\}$ che, per ogni colonna, indica la riga dove è presente il primo coefficiente di rigidezza diverso da zero. Facendo riferimento all'esempio [5.2.17] precedente, questo vettore sarebbe il seguente:

$$\{skyline\} = \{1 \quad 1 \quad 1 \quad 3\}$$

Posto $D_{11} = K_{11}$ l'algoritmo procede nel modo seguente:

- Per $j = 2, \dots, N$
 - $m_j = skyline(j)$
 - $G_{m_j j} = K_{m_j j}$
 - Per $i = m_j + 1, \dots, j - 1$:
 - $m_i = skyline(i)$
 - $m_m = \max(m_i, m_j)$
 - $G_{ij} = K_{ij} - \sum_{k=m_m}^{i-1} L_{ki}G_{kj}$ [5.2.22]
 - Per $i = m_j, \dots, j - 1$:
 - $L_{ij} = \frac{G_{ij}}{D_{ii}}$ [5.2.23]
 - $D_{jj} = K_{jj} - \sum_{k=m_j}^{j-1} L_{kj}G_{kj}$ [5.2.24]

Si può notare che le sommatorie nella (5.2.22) e nella (5.2.24) non comportano la moltiplicazione dei coefficienti nulli esterni alla skyline della matrice e che i coefficienti L_{ij} appartengono alla matrice $[L]^T$. Si può



notare inoltre che gli elementi L_{ij} calcolati per l'esecuzione dell'istruzione (5.2.24) sostituiscono (cioè occupano la stessa posizione di memoria) i coefficienti G_{ij} e che i coefficienti D_{jj} sostituiscono i coefficienti K_{jj} . Quindi alla fine del processo abbiamo che gli elementi D_{jj} si trovano nelle locazioni di memoria prima occupate da K_{jj} e i coefficienti L_{kj} sono memorizzati nelle posizioni prima occupate da K_{kj} con $j > k$.

Vediamo il seguente esempio:

$$[K] = \begin{bmatrix} 2 & -2 & 0 & 0 & -1 \\ -2 & 3 & -2 & 0 & 0 \\ 0 & -2 & 5 & -3 & 0 \\ 0 & 0 & -3 & 10 & 4 \\ -1 & 0 & 0 & 4 & 10 \end{bmatrix} \quad \{skyline\} = \{1 \quad 1 \quad 2 \quad 3 \quad 1\}$$

Eseguiamo l'algoritmo:

$$D_{11} = K_{11} = 2$$

- Per $j = 2$
 - $m_2 = skyline(2) = 1$
 - $G_{12} = K_{12} = -2$
 - Per $i = 1$
 - $L_{12} = \frac{G_{12}}{D_{11}} = \frac{-2}{2} = -1$
 - $D_{22} = K_{22} - L_{12}G_{12} = 3 - (-1) \cdot (-2) = 1$

Al termine della prima iterazione la matrice risulta la seguente: $[K] = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -2 & 1 & -2 & 0 & 0 \\ 0 & -2 & 5 & -3 & 0 \\ 0 & 0 & -3 & 10 & 4 \\ -1 & 0 & 0 & 4 & 10 \end{bmatrix}$

- Per $j = 3$
 - $m_3 = skyline(3) = 2$
 - $G_{23} = K_{23} = -2$
 - Per $i = 2$
 - $L_{23} = \frac{G_{23}}{D_{22}} = \frac{-2}{1} = -2$
 - $D_{33} = K_{33} - L_{23}G_{23} = 5 - (-2)(-2) = 1$

Al termine della seconda iterazione la matrice risulta la seguente: $[K] = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -2 & 1 & -2 & 0 & 0 \\ 0 & -2 & 1 & -3 & 0 \\ 0 & 0 & -3 & 10 & 4 \\ -1 & 0 & 0 & 4 & 10 \end{bmatrix}$

- Per $j = 4$
 - $m_4 = skyline(4) = 3$
 - $G_{34} = K_{34} = -3$
 - Per $i = 3$
 - $L_{34} = \frac{G_{34}}{D_{33}} = \frac{-3}{1} = -3$
 - $D_{44} = K_{44} - L_{34}G_{34} = 10 - (-3)(-3) = 1$



Al termine della terza iterazione la matrice risulta la seguente: $[K] = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -2 & 1 & -2 & 0 & 0 \\ 0 & -2 & 1 & -3 & 0 \\ 0 & 0 & -3 & 1 & 4 \\ -1 & 0 & 0 & 4 & 10 \end{bmatrix}$

- Per $j = 5$
 - $m_5 = \text{skyline}(5) = 1$
 - $G_{15} = K_{15} = -1$
 - Per $i = 2$
 - $m_2 = \text{skyline}(2) = 1$
 - $m_m = \max(m_2, m_5) = 1$
 - $G_{25} = K_{25} - L_{12}G_{15} = 0 - (-1)(-1) = -1$
 - Per $i = 3$
 - $m_3 = \text{skyline}(3)$
 - $m_m = \max(m_3, m_5) = 2$
 - $G_{35} = K_{35} - L_{23}G_{25} = 0 - (-2)(-1) = -2$
 - Per $i = 4$
 - $m_4 = \text{skyline}(4)$
 - $m_m = \max(m_4, m_5) = 3$
 - $G_{45} = K_{45} - L_{34}G_{35} = 4 - (-3)(-2) = -2$
 - Per $i = 1, 2, 3, 4$: $L_{i5} = \frac{G_{i5}}{D_{ii}}$
 - $L_{15} = \frac{G_{15}}{D_{11}} = \frac{-1}{2}$; $L_{25} = \frac{G_{25}}{D_{22}} = \frac{-1}{1}$; $L_{35} = \frac{G_{35}}{D_{33}} = \frac{-2}{1}$; $L_{45} = \frac{G_{45}}{D_{44}} = \frac{-2}{1}$
 - $D_{55} = K_{55} - L_{15}G_{15} - L_{25}G_{25} - L_{35}G_{35} - L_{45}G_{45} = 10 - \left(\frac{-1}{2}\right)(-1) - (-1)(-1) - (-2)(-2) - (-2)(-2) = 10 - 0.5 - 1 - 4 - 4 = 0.5$

Al termine della quarta e ultima iterazione la matrice risulta la seguente: $[K] = \begin{bmatrix} 2 & -1 & 0 & 0 & -0.5 \\ -2 & 1 & -2 & 0 & -1 \\ 0 & -2 & 1 & -3 & -2 \\ 0 & 0 & -3 & 1 & -2 \\ -1 & 0 & 0 & 4 & 0.5 \end{bmatrix}$

I coefficienti della matrice $[D]$ e della matrice $[L]^T$ hanno preso il posto rispettivamente degli elementi K_{ii} e K_{ij} con $j > i$ dell'originale matrice $[K]$.

Una volta che la matrice $[K]$ è stata fattorizzata nelle sue componenti $[D]$ ed $[L]$, è possibile calcolare la soluzione del sistema usando le equazioni:

$$[L]\{y\} = \{F\} \quad \text{forward substitution}$$

e
$$[L]^T\{x\} = [D]^{-1}\{y\} \quad \text{back substitution}$$

A differenza di quanto capita con la fattorizzazione $[L][U]$, è possibile salvare in memoria solo la matrice $[L]^T$ e la diagonale $[D]$; inoltre il calcolo del vettore $\{y\}$ può essere fatto contestualmente alla fattorizzazione oppure in un secondo momento.

Poiché il metodo di Cholesky fattorizza la matrice come indicato nella (5.2.19) è evidente che tra i due metodi esiste la seguente relazione:

$$[\tilde{L}][\tilde{L}]^T = [K] \quad \text{fattorizzazione di Cholesky}$$

$$[L][D][L]^T = [K]$$

da cui: $[\tilde{L}] = [L][D]^{1/2}$

Si può dimostrare che la soluzione di un sistema di equazioni lineari con la fattorizzazione di Cholesky



richiede alcune operazioni in più rispetto al metodo che usa la fattorizzazione $[L][D][L]^T$. Inoltre la fattorizzazione di Cholesky è adatta solo alla soluzione di sistemi definiti positivi per i quali tutti i termini K_{ii} sulla diagonale risultino positivi. Viceversa la fattorizzazione $[L][D][L]^T$ può essere usata anche con sistemi indefiniti.

Vediamo il seguente esempio ottenuto aggiungendo una riga ed una colonna alla matrice usata nell'esempio precedente:

$$[K] = \begin{bmatrix} 2 & -2 & 0 & 0 & -1 & 0 \\ -2 & 3 & -2 & 0 & 0 & 0 \\ 0 & -2 & 5 & -3 & 0 & 1 \\ 0 & 0 & -3 & 10 & 4 & 0 \\ -1 & 0 & 0 & 4 & 10 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \quad \{skyline\} = \{1 \quad 1 \quad 2 \quad 3 \quad 1 \quad 3\}$$

La fattorizzazione delle prime quattro colonne conduce allo stesso risultato visto nell'esempio precedente. Vediamo di fattorizzare anche la sesta colonna. Applicando l'algoritmo abbiamo:

- Per $j = 6$
 - $m_6 = skyline(6) = 3$
 - $G_{36} = K_{36} = 1$
 - Per $i = 4$:
 - $m_4 = skyline(4) = 3$
 - $m_m = \max(m_4, m_6) = 3$
 - $G_{46} = K_{46} - \sum_{k=3}^3 L_{k4}G_{k6} = K_{46} - L_{34}G_{36} = 0 - (-3) \cdot 1 = 3$
 - Per $i = 5$:
 - $m_5 = skyline(5) = 1$
 - $m_m = \max(m_5, m_6) = 3$
 - $G_{56} = K_{56} - \sum_{k=3}^4 L_{k5}G_{k6} = K_{56} - L_{35}G_{36} - L_{45}G_{46} = 1 - (-2) \cdot 1 - (-2) \cdot 3 = 9$
 - Per $i = 3, 4, 5$: $L_{ij} = \frac{G_{ij}}{D_{ii}}$
 - $L_{36} = \frac{G_{36}}{D_{33}} = \frac{1}{1} = 1$; $L_{46} = \frac{G_{46}}{D_{44}} = \frac{3}{1} = 3$; $L_{56} = \frac{G_{56}}{D_{55}} = \frac{9}{0.5} = 18$
- $$D_{66} = K_{66} - \sum_{k=3}^5 L_{kj}G_{kj} = K_{66} - L_{36}G_{36} - L_{46}G_{46} - L_{56}G_{56} = 0 - 1 \cdot 1 - 3 \cdot 3 - 18 \cdot 9 = -172$$

Al termine della quinta e ultima iterazione la matrice risulta la seguente:

$$[K] = \begin{bmatrix} 2 & -1 & 0 & 0 & -0.5 & 0 \\ -2 & 1 & -2 & 0 & -1 & 0 \\ 0 & -2 & 1 & -3 & -2 & 1 \\ 0 & 0 & 1 & 1 & -2 & 3 \\ -1 & 0 & 0 & 4 & 0.5 & 18 \\ 0 & 0 & 1 & 0 & 1 & -172 \end{bmatrix}$$



5.3 I metodi iterativi

A differenza dei metodi diretti che si basano sull'algoritmo di Gauss, i metodi iterativi richiedono un numero indefinito di operazioni, e terminano solo quando la procedura soddisfa un determinato criterio di arresto. Si tratta di metodi meno utilizzati rispetto a quelli diretti, ma godono di alcune proprietà interessanti. Qui ne verranno descritte alcune.

5.3.1 Il metodo di Gauss-Seidel

Il vettore $\{D\}$ che soddisfa il sistema $[K]\{D\} = \{F\}$ è lo stesso che minimizza la seguente funzione:

$$f(\{D\}) = \pi_{pt} = \frac{1}{2}\{D\}^T[K]\{D\} - \{D\}^T\{F\} \quad [5.3.1]$$

In un problema strutturale $[K]$ rappresenta la matrice di rigidità, $\{F\}$ il vettore delle forze applicate, $\{D\}$ il vettore degli spostamenti e π_{pt} l'energia elastica accumulata dalla struttura.

Uno dei metodi per la minimizzazione delle funzioni di più variabili è quello della discesa sequenziale.

Sia $\{D_0\} = \{d_1, d_2, d_3, \dots, d_N\}$ un'approssimazione del punto di minimo della funzione $f(\{D\})$. Ipotizzando che $f(d_1, d_2, d_3, \dots)$ dipenda solo dalla variabile d_1 , si tratta di trovare il valore d_1^* che minimizza la funzione (5.3.1). Calcolata l'approssimazione $f(d_1^*, d_2, d_3, \dots)$ si cercheranno le approssimazioni successive agendo sulle restanti variabili d_2, d_3, \dots . Il processo si ripete ciclicamente. Il calcolo di d_k^* provoca lo spostamento della soluzione lungo la retta parallela all'asse d_k (Fig. 5.3.1), fino al punto in cui $\partial f(\{D\})/\partial d_k = 0$: la funzione $f(\dots, d_k^*, \dots)$ raggiunge il minimo e la retta incontra la linea di livello $f(\dots, d_k^*, \dots) = cost$.

Nel caso bidimensionale, il processo d'approssimazione si presenta come nella Fig.5.3.1.

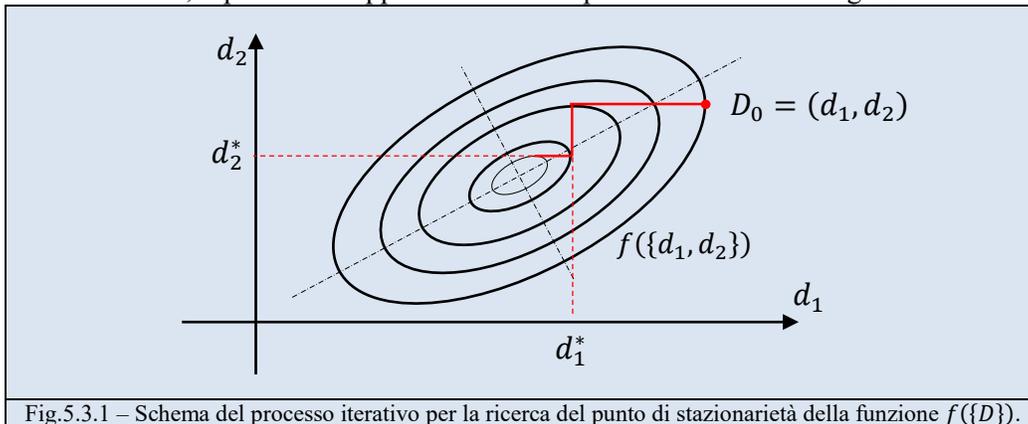


Fig.5.3.1 – Schema del processo iterativo per la ricerca del punto di stazionarietà della funzione $f(\{D\})$.

Vediamo più in dettaglio la successione delle operazioni necessarie all'implementazione del metodo iterativo. Esprimendo per esteso il sistema di equazioni si ottiene:

$$\begin{cases} K(1,1)d_{k+1}(1) + K(1,2)d_k(2) + \dots + K(1,N)d_k(N) = F(1) \\ K(2,1)d_{k+1}(1) + K(2,2)d_{k+1}(2) + \dots + K(2,N)d_k(N) = F(2) \\ \dots \\ K(N,1)d_{k+1}(1) + K(N,2)d_{k+1}(2) + \dots + K(N,N)d_{k+1}(N) = F(N) \end{cases} \quad [5.3.2]$$

si osserva che usando la i -esima equazione è possibile calcolare la i -esima componente. Noto il vettore $\{D\}_k$, dalla prima delle (5.3.2) si calcola $d_{k+1}(1)$:

$$d_{k+1}(1) = \frac{F(1) - K(1,2)d_k(2) - \dots - K(1,N)d_k(N)}{K(1,1)}$$

Dalla seconda si calcola $d_{k+1}(2)$:

$$d_{k+1}(2) = \frac{F(2) - K(2,1)d_{k+1}(1) - K(2,3)d_k(3) - \dots - K(2,N)d_k(N)}{K(2,2)}$$



etc. etc. fino all'ultima equazione. Chiaramente il metodo è utilizzabile purché i coefficienti sulla diagonale siano diversi da zero.

In forma matriciale, posto $[K] = [L] + [U]$, il sistema $[K]\{D\} = \{F\}$ può trasformarsi nel modo seguente:

$$[K]\{D\} = ([L] + [U])\{D\} = \{F\}$$

Quindi la (5.3.2) può esprimersi in forma matriciale nel modo seguente:

$$[L]\{D\}_{k+1} + [U]\{D\}_k = \{F\} \quad [5.3.3]$$

dove:

$$[L] = \begin{bmatrix} K_{11} & 0 & 0 & \cdots & 0 \\ K_{21} & K_{22} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ K_{N1} & K_{N2} & K_{N3} & \cdots & K_{NN} \end{bmatrix} \quad ; \quad [U] = \begin{bmatrix} 0 & K_{12} & K_{13} & \cdots & K_{1N} \\ 0 & 0 & K_{23} & \cdots & K_{2N} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

da cui si ricava:

$$\{D\}_{k+1} = [L]^{-1}\{\{F\} - [U]\{D\}_k\} \quad [5.3.4]$$

E' interessante un'interpretazione geometrica del metodo. Si indichi con L_i il piano:

$$\sum_{j=1}^N K(i,j)d(j) - F(i) = 0$$

Data la soluzione approssimata $\{d_{k+1}(1) \cdots d_{k+1}(i-1) \ d_k(i) \cdots d_k(N)\}$, la successiva soluzione $\{d_{k+1}(1) \cdots d_{k+1}(i) \ d_k(i+1) \cdots d_k(N)\}$ si ottiene spostandosi parallelamente all'asse $d(i)$ fino all'intersezione con il piano L_i ; così, geometricamente, il metodo consiste nello spostamento ciclico parallelamente agli assi $d(i)$ fino all'intersezione con i piani L_i (vedi Fig.5.A1)

Esempio numerico

Ipotizziamo che il sistema da risolvere sia il seguente:

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{Bmatrix} x \\ y \end{Bmatrix} = \begin{Bmatrix} 5 \\ 4 \end{Bmatrix} \quad \text{la cui soluzione è } \begin{Bmatrix} 2 \\ 1 \end{Bmatrix}.$$

Gli autovalori si trovano annullando il seguente determinante:

$$\det \begin{vmatrix} 2 - \lambda & 1 \\ 1 & 2 - \lambda \end{vmatrix} = 0$$

ovvero cercando gli zeri del polinomio caratteristico: $(2 - \lambda)^2 - 1 = 0$, da cui risulta:

$\lambda_1 = 1$ e $\lambda_2 = 3$ e il numero di condizione della matrice vale $\frac{\lambda_2}{\lambda_1} = 3$.

Gli autovettori corrispondenti sono i seguenti:

$$\text{per } \lambda_1 = 1 \quad \begin{bmatrix} 2 - \lambda_1 & 1 \\ 1 & 2 - \lambda_1 \end{bmatrix} \begin{Bmatrix} x_1 \\ y_1 \end{Bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \begin{Bmatrix} x_1 \\ y_1 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

Posto arbitrariamente $x_1 = 1$ risulta $y_1 = -1$.

$$\text{Quando } \lambda_2 = 3 \quad \begin{bmatrix} 2 - \lambda_2 & 1 \\ 1 & 2 - \lambda_2 \end{bmatrix} \begin{Bmatrix} x_2 \\ y_2 \end{Bmatrix} = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \begin{Bmatrix} x_2 \\ y_2 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

Posto arbitrariamente $x_2 = 1$ risulta $y_2 = 1$.



La prima equazione che rappresenta la retta L_1 è la seguente: $2x + y = 5$;
la seconda equazione rappresenta la retta L_2 : $x + 2y = 4$.

La funzione obiettivo da minimizzare è la seguente:

$$f(x, y) = \frac{1}{2} \begin{Bmatrix} x & y \end{Bmatrix} \begin{bmatrix} k_{11} & k_{12} \\ k_{12} & k_{22} \end{bmatrix} \begin{Bmatrix} x \\ y \end{Bmatrix} - \begin{Bmatrix} x & y \end{Bmatrix} \begin{Bmatrix} F_1 \\ F_2 \end{Bmatrix}$$

da cui ricaviamo la quadrica:

$$k_{11}x^2 + 2k_{12}xy + k_{22}y^2 - 2F_1x - 2F_2y = 2f(x, y)$$

Sostituendo il valore dei coefficienti della matrice e del vettore dei termini noti si ottiene:

$$x^2 + xy + y^2 - 5x - 4y = f(x, y)$$

Fissato un valore di f , è possibile rappresentare l'ellisse, luogo dei punti sui quali $f = costante$. Gli assi principali dell'ellisse sono inclinati rispetto agli assi coordinati come indicato dagli autovettori della matrice e il rapporto tra la lunghezza dei due assi è pari al rapporto tra gli autovalori e quindi al numero di condizione.

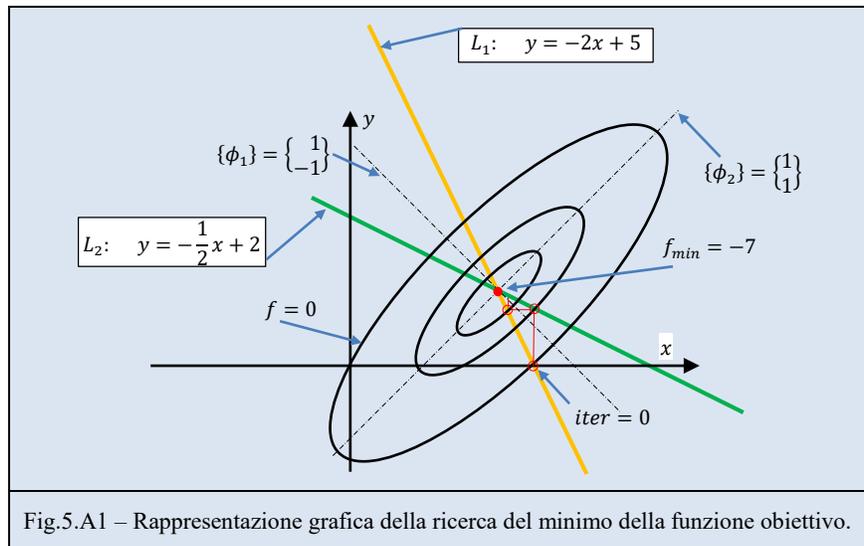


Fig.5.A1 – Rappresentazione grafica della ricerca del minimo della funzione obiettivo.

La soluzione esatta minimizza la precedente funzione obiettivo.

1° iterazione

Alla prima iterazione ipotizziamo $y_0 = 0$ e cerchiamo la coordinata x lungo la retta L_1

Utilizzando la retta L_1 : $2x_1 + y_0 = 5$ ricaviamo la coordinata $x_1 = 5/2$;

Utilizzando la retta L_2 : $x_1 + 2y_1 = 4$ ricaviamo la coordinata $y_1 = \frac{4-x_1}{2} = \frac{3}{4}$;

Al termine della 1° iterazione la funzione obiettivo da minimizzare vale:

$$\left(\frac{5}{2}\right)^2 + \left(\frac{5}{2}\right)\left(\frac{3}{4}\right) + \left(\frac{3}{4}\right)^2 - 5\frac{5}{2} - 4\frac{3}{4} = f_1 = -6.8125$$

2° iterazione

Utilizzando la retta L_1 : $2x_2 + y_1 = 5$ ricaviamo la coordinata $x_2 = \frac{5-y_1}{2} = \frac{17}{8}$;

Utilizzando la retta L_2 : $x_2 + 2y_2 = 4$ ricaviamo la coordinata $y_2 = \frac{4-x_2}{2} = \frac{15}{16}$;

Al termine della 2° iterazione la funzione obiettivo da minimizzare vale:



$$\left(\frac{17}{8}\right)^2 + \left(\frac{17}{8}\right)\left(\frac{15}{16}\right) + \left(\frac{15}{16}\right)^2 - 5\frac{17}{8} - 4\frac{15}{16} = f_2 = -6.9883$$

3° iterazione

Utilizzando la retta $L_1: 2x_3 + y_2 = 5$ ricaviamo la coordinata $x_3 = \frac{5-y_2}{2} = \frac{65}{32}$;

Utilizzando la retta $L_2: x_3 + 2y_3 = 4$ ricaviamo la coordinata $y_3 = \frac{4-x_3}{2} = \frac{63}{64}$;

Al termine della 3° iterazione l'errore vale:

$$\left(\frac{65}{32}\right)^2 + \left(\frac{65}{32}\right)\left(\frac{63}{64}\right) + \left(\frac{63}{64}\right)^2 - 5\frac{65}{32} - 4\frac{63}{64} = f_3 = -6.9993$$

Riassumendo i risultati al termine della terza iterazione abbiamo:

$$\begin{Bmatrix} 2.5 \\ 0.75 \end{Bmatrix} \rightarrow \begin{Bmatrix} 2.125 \\ 0.9375 \end{Bmatrix} \rightarrow \begin{Bmatrix} 2.03125 \\ 0.984375 \end{Bmatrix} \rightarrow \text{che converge verso la soluzione esatta} \rightarrow \begin{Bmatrix} 2 \\ 1 \end{Bmatrix}$$



5.3.2 Metodo del gradiente più rapido.

Il metodo del gradiente è un procedimento molto usato per la minimizzazione delle funzioni di più variabili. Ogni approssimazione si ottiene da quella precedente spostando la soluzione in direzione del gradiente della funzione obiettivo $f(x)$ di una quantità d . In tal modo, ogni nuova approssimazione assume la forma:

$$\{D\}_{k+1} = \{D\}_k - d \times \text{grad}[f(\{D\}_k)]$$

Il parametro d può essere determinato minimizzando la quantità:

$$f(\{D\}_k - d \times \text{grad}[f(\{D\}_k)])$$

In questo caso il metodo si chiama *metodo del gradiente più rapido*. Quando:

$$f(\{D\}) = \frac{1}{2}\{D\}^T[K]\{D\} - \{D\}^T\{F\}$$

il gradiente vale:

$$\text{grad}[f] = [K]\{D\} - \{F\}$$

e

$$\{D\}_{n+1} = \{D\}_n - d \times \text{grad}[f(\{D\}_n)] = \{D\}_n - d \times ([K]\{D\}_n - \{F\})$$

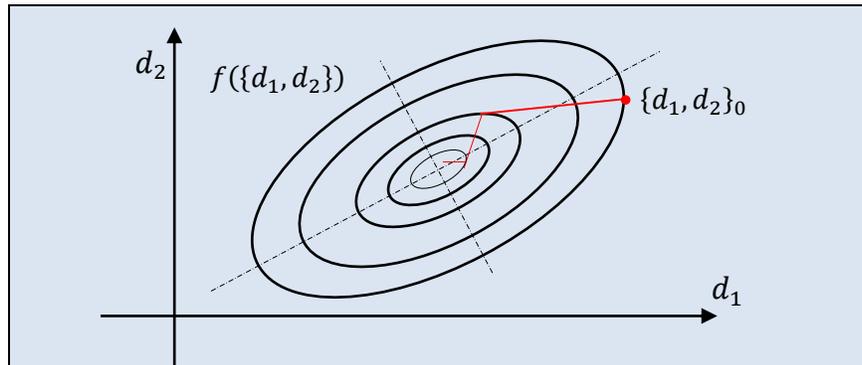


Fig.5.3.2 – Schema del processo iterativo per la ricerca del punto di stazionarietà della funzione $f(\{D\})$. La direzione di ricerca è quella del gradiente della funzione $f(D)$.

5.3.3 Il Metodo del Gradiente Coniugato

La forma più semplice dell’algoritmo del gradiente coniugato risolve il sistema solo nel caso in cui $[K]$ sia simmetrica e definita positiva. Sebbene non esista alcuna garanzia che l’intera procedura non si interrompa a causa di un errore o diventi instabile, in pratica ciò capita molto raramente. E’ molto importante osservare che il processo iterativo converge al massimo dopo N iterazioni (dove N indica il numero di righe e colonne della matrice $[K]$) solo in matematica esatta, ma a causa degli errori di arrotondamento è necessario stabilire un criterio di arresto. Normalmente il processo iterativo si interrompe quando

$$\frac{|[K]\{D\}_k - \{F\}|}{|\{F\}|} < tol$$

dove $\{D\}_k$ indica la stima alla k -esima iterazione e $|\cdot|$ è la norma euclidea. Solitamente si pone $tol = 10^{-6}$. Al programma si fornisce anche il numero massimo di iterazioni $Iter_{max}$ per prevenire loop infiniti o comunque troppo lunghi.

Per usare l’algoritmo bisogna fornire un vettore $\{D\}_1$ di primo tentativo. In mancanza di particolari informazioni, solitamente si pone $\{D\}_1 = \{0\}$, ma è evidente che una buona stima iniziale accelera il processo iterativo. Si calcola quindi il vettore residuo:

$$\{r\}_1 = \{F\} - [K]\{D\}_1$$



Dopo di che inizia il processo iterativo:

Per $k = 1, 2, \dots, Iter_{max}$

$$\{v\} = [K]\{r\}_k$$

$$d_k = \frac{\{r\}_k^T \{r\}_k}{\{r\}_k^T \{v\}}$$

$$\{D\}_{k+1} = \{D\}_k + d_k \{r\}_k$$

$$\{r\}_{k+1} = \{r\}_k - d_k \{v\}$$

$$se \frac{|\{r\}_{k+1}|}{|\{F\}|} \leq tol \quad STOP$$

Oltre alla matrice $[K]$ ed al vettore $\{F\}$, l'algoritmo richiede lo spazio per ulteriori due vettori, $\{v\}$ e $\{r\}$. Al termine del processo $[K]$ ed $\{F\}$ non vengono modificati come capita con la fattorizzazione di Gauss.

Quando la matrice $[K]$ **non è definita positiva o non è simmetrica** bisogna ricorrere al metodo del gradiente bi-coniugato che non verrà descritto in questi appunti.

Esempio numerico

Ipotizziamo che il sistema da risolvere sia il seguente:

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{Bmatrix} x \\ y \end{Bmatrix} = \begin{Bmatrix} 5 \\ 4 \end{Bmatrix} \quad \text{la cui soluzione è } \begin{Bmatrix} 2 \\ 1 \end{Bmatrix}.$$

In questo esempio si porrà: $tol = 10^{-3}$

Vettore di primo tentativo: $\{D\}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$; $\{r\}_1 = \{F\} - [K]\{D\}_1 = \{F\} = \begin{Bmatrix} 5 \\ 4 \end{Bmatrix}$

<p>1) $k = 1$</p> $\{v\} = [K]\{r\}_1 = \begin{Bmatrix} 14 \\ 13 \end{Bmatrix}$ $d_1 = \frac{\{r\}_1^T \{r\}_1}{\{r\}_1^T \{v\}} = 0.336$ $\{D\}_2 = \{D\}_1 + d_1 \{r\}_1 = \begin{Bmatrix} 1.680 \\ 1.344 \end{Bmatrix}$ $\{r\}_2 = \{r\}_1 - d_1 \{v\} = \begin{Bmatrix} 0.295 \\ -0.369 \end{Bmatrix}$ <p>Check: $\frac{ \{r\}_2 }{ \{F\} } = 0.074 < tol ?$ NO</p>	<p>2) $k = 2$</p> $\{v\} = [K]\{r\}_2 = \begin{Bmatrix} 0.221 \\ -0.443 \end{Bmatrix}$ $d_2 = \frac{\{r\}_2^T \{r\}_2}{\{r\}_2^T \{v\}} = 0.976$ $\{D\}_3 = \{D\}_2 + d_2 \{r\}_2 = \begin{Bmatrix} 1.968 \\ 0.984 \end{Bmatrix}$ $\{r\}_3 = \{r\}_2 - d_2 \{v\} = \begin{Bmatrix} 0.079 \\ 0.063 \end{Bmatrix}$ <p>Check: $\frac{ \{r\}_3 }{ \{F\} } = 0.0158 < tol ?$ NO</p>
<p>3) $k = 3$</p> $\{v\} = [K]\{r\}_3 = \begin{Bmatrix} 0.221 \\ 0.206 \end{Bmatrix}$ $d_3 = \frac{\{r\}_3^T \{r\}_3}{\{r\}_3^T \{v\}} = 0.336$ $\{D\}_4 = \{D\}_3 + d_3 \{r\}_3 = \begin{Bmatrix} 1.9995 \\ 1.0054 \end{Bmatrix}$ $\{r\}_4 = \{r\}_3 - d_3 \{v\} = \begin{Bmatrix} 0.0046 \\ -0.0058 \end{Bmatrix}$ <p>Check: $\frac{ \{r\}_4 }{ \{F\} } = 0.0012 < tol ?$ NO</p>	<p>4) $k = 4$</p> $\{v\} = [K]\{r\}_4 = \begin{Bmatrix} 0.0035 \\ -0.007 \end{Bmatrix}$ $d_4 = \frac{\{r\}_4^T \{r\}_4}{\{r\}_4^T \{v\}} = 0.976$ $\{D\}_5 = \{D\}_4 + d_4 \{r\}_4 = \begin{Bmatrix} 1.9995 \\ 0.9998 \end{Bmatrix}$ $\{r\}_5 = \{r\}_4 - d_4 \{v\} = \begin{Bmatrix} 0.0012 \\ 0.001 \end{Bmatrix}$ <p>Check: $\frac{ \{r\}_5 }{ \{F\} } = 0.0003 < tol ?$ SI STOP</p>

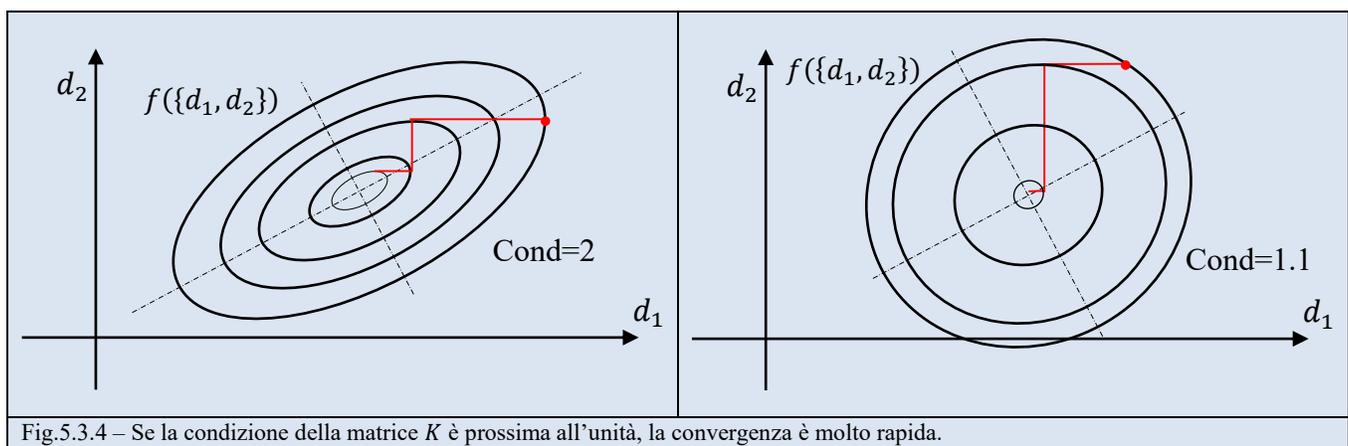
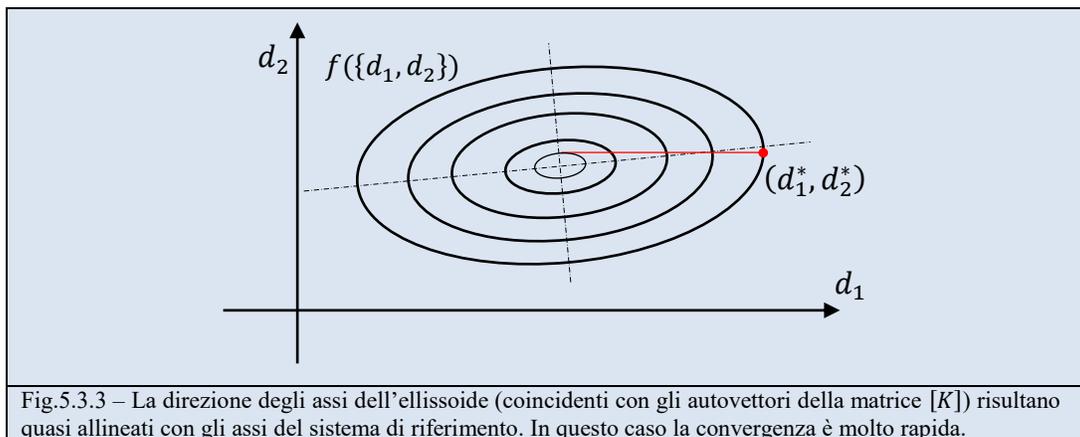


MATLAB nella versione 7.11.0 (R2010b) prevede diverse procedure per la soluzione iterativa dei sistemi di equazioni lineari; qui di seguito ne sono elencate alcune.

bicg	Metodo del gradiente Biconiugato
cgs	Metodo del gradiente coniugato
minres	Metodo del residuo minimo
pcg	Metodo del gradiente coniugato Precondizionato

5.3.4 Velocità di convergenza

In generale la velocità di convergenza dei metodi iterativi è lenta e dipende dal numero di condizione della matrice $[K]$ e dalla direzione dei suoi autovettori rispetto agli assi coordinati. Dalla fig. 5.3.3 si può vedere che il metodo converge più rapidamente se gli autovettori (che indicano la direzione degli assi principali degli ellissoidi) sono allineati alla direzione degli assi coordinati, vale a dire se la matrice $[K]$ è vicina a quella diagonale. Dalla fig. 5.3.4 si può vedere che il metodo converge più rapidamente se il numero di condizione della matrice $[K]$ (ovvero il rapporto tra il suo autovalore più grande e quello più piccolo) è prossimo all'unità: alcune tecniche di accelerazione della convergenza si basano su questa osservazione, cercano cioè, preconditionando la matrice, di ridurre il numero di condizione.



Questo suggerisce di applicare questi metodi alla forma preconditionata dell'equazione:

$$([\tilde{K}]^{-1}[K])\{D\} = [\tilde{K}]^{-1}\{F\}$$

dove $[\tilde{K}]^{-1}[K] \cong [I]$, dove $[I]$ indica la matrice identità. In questo modo l'algorithm converge più rapidamente.



La matrice $[\tilde{K}]$ prende il nome di *precondizionatore*: quando applicata all'algoritmo del gradiente coniugato, lo schema di calcolo complessivo prende il nome di “*Metodo del Gradiente Coniugato Precondizionato*” o PCG. Visto che serve ad accelerare il metodo iterativo, la matrice $[\tilde{K}]$ deve essere calcolata molto rapidamente e non deve occupare troppa memoria. Una scelta pratica consiste nel prendere $[\tilde{K}]$ uguale alla diagonale della matrice $[K]$; in alternativa alcuni algoritmi utilizzano una versione semplificata della fattorizzazione di Cholesky.