

Soluzioni Analitiche e Numeriche Applicate all'Ingegneria Ambientale

Massimiliano Martinelli
massimiliano.martinelli@gmail.com

Università Politecnica delle Marche, Ancona
Facoltà di Ingegneria

21-22 Gennaio 2009

Outline

- 1 Introduzione
- 2 Metodi numerici per la risoluzione di sistemi lineari
 - Metodi diretti

Obiettivi del corso

- Introduzione agli strumenti di base e ai metodi numerici per la soluzione di equazioni differenziali ordinarie ed alle derivate parziali che intervengono nei modelli utilizzati in ingegneria ambientale

Obiettivi del corso

- Introduzione agli strumenti di base e ai metodi numerici per la soluzione di equazioni differenziali ordinarie ed alle derivate parziali che intervengono nei modelli utilizzati in ingegneria ambientale

Esempi

- Trasporto di sostanze inquinanti
- Simulazioni oceanografiche
- Previsioni meteorologiche
- ...

Obiettivi del corso

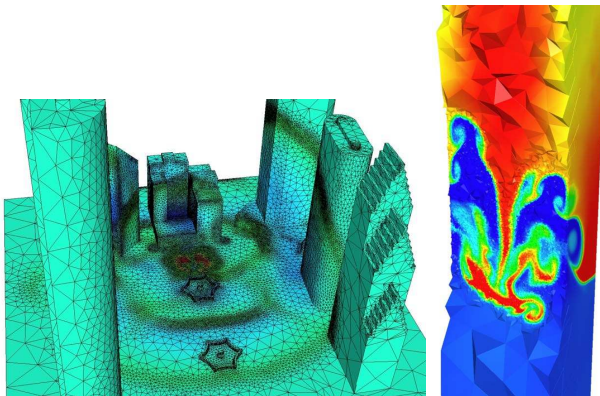
- Introduzione agli strumenti di base e ai metodi numerici per la soluzione di equazioni differenziali ordinarie ed alle derivate parziali che intervengono nei modelli utilizzati in ingegneria ambientale

Esempi

- Trasporto di sostanze inquinanti
- Simulazioni oceanografiche
- Previsioni meteorologiche
- ...

Si descriveranno (alcuni) metodi numerici per la soluzione di:

- Sistemi lineari
- Sistemi non-lineari
- Differenziazione e Integrazione numerica
- Equazioni differenziali ordinarie
- Equazioni alle derivate parziali

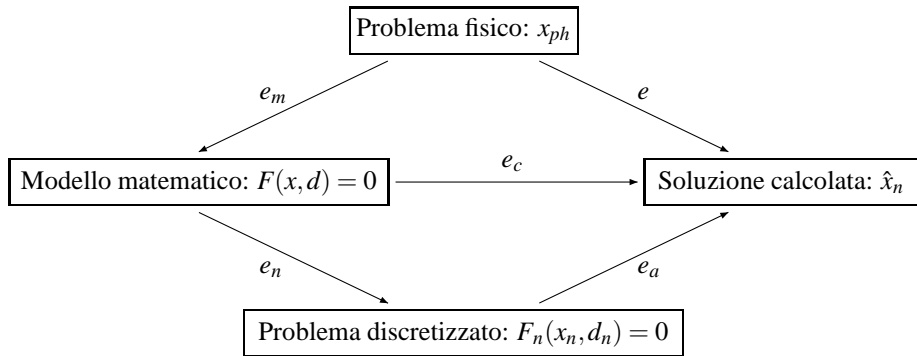


- Animazione 1
- Animazione 2
- Animazione 3

Libri consigliati

- **A. Quarteroni, R. Sacco, F. Saleri, “*Matematica Numerica*”, Springer**
- W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, “*Numerical Recipes: The Art of Scientific Computing, Third Edition*”, Cambridge University Press, <http://www.nr.com>
- Y. Saad, “*Iterative Methods for Sparse Linear Systems*”, SIAM, <http://www-users.cs.umn.edu/~saad/books.html>
- J.W. Thomas, “*Numerical Partial Differential Equations: Finite Difference Methods*”, Springer
- A. Quarteroni, “*Modellistica numerica per problemi differenziali*”, Springer

Errori nei modelli computazionali



- e_m errore del modello matematico
- e_c errore del modello computazionale
- e_n errore di discretizzazione
- e_a errore dovuto dall'algoritmo

Operazioni in virgola mobile

- Ogni $x \in \mathbb{R}$ tale che $x_{\min} \leq x \leq x_{\max}$ viene convertito in *virgola mobile* $\text{fl}(\cdot): \mathbb{R} \mapsto \mathbb{F}$, dove

$$\text{fl}(x) = x(1 + \delta) \quad \text{con } |\delta| \leq u = \frac{1}{2}\beta^{1-t} = \frac{\epsilon_M}{2}$$

u = precisione macchina (o unita di arrotondamento)

Operazioni in virgola mobile

- Ogni $x \in \mathbb{R}$ tale che $x_{\min} \leq x \leq x_{\max}$ viene convertito in *virgola mobile* $\text{fl}(\cdot): \mathbb{R} \mapsto \mathbb{F}$, dove

$$\text{fl}(x) = x(1 + \delta) \quad \text{con } |\delta| \leq u = \frac{1}{2}\beta^{1-t} = \frac{\varepsilon_M}{2}$$

u = precisione macchina (o unita di arrotondamento)

- Ogni operazione aritmetica $\circ: \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ viene sostituita con

$$\boxtimes: \mathbb{R} \times \mathbb{R} \mapsto \mathbb{F} \quad x \boxtimes y = \text{fl}(\text{fl}(x) \circ \text{fl}(y))$$

Operazioni in virgola mobile

- Ogni $x \in \mathbb{R}$ tale che $x_{\min} \leq x \leq x_{\max}$ viene convertito in *virgola mobile* $\text{fl}(\cdot): \mathbb{R} \mapsto \mathbb{F}$, dove

$$\text{fl}(x) = x(1 + \delta) \quad \text{con } |\delta| \leq u = \frac{1}{2}\beta^{1-t} = \frac{\epsilon_M}{2}$$

u = precisione macchina (o unita di arrotondamento)

- Ogni operazione aritmetica $\circ: \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ viene sostituita con

$$\boxtimes: \mathbb{R} \times \mathbb{R} \mapsto \mathbb{F} \quad x \boxtimes y = \text{fl}(\text{fl}(x) \circ \text{fl}(y))$$

- Utilizzando la cifra di arrotondamento, è verificata la proprietà seguente:

$$\forall x, y \in \mathbb{R} \quad \exists \delta \in \mathbb{R} \quad \text{t.c.} \quad x \boxtimes y = (x \circ y)(1 + \delta) \quad \text{con} \quad |\delta| \leq u$$

Operazioni in virgola mobile

- Ogni $x \in \mathbb{R}$ tale che $x_{\min} \leq x \leq x_{\max}$ viene convertito in *virgola mobile* $\text{fl}(\cdot): \mathbb{R} \mapsto \mathbb{F}$, dove

$$\text{fl}(x) = x(1 + \delta) \quad \text{con } |\delta| \leq u = \frac{1}{2}\beta^{1-t} = \frac{\varepsilon_M}{2}$$

u = precisione macchina (o unità di arrotondamento)

- Ogni operazione aritmetica $\circ: \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ viene sostituita con

$$\boxtimes: \mathbb{R} \times \mathbb{R} \mapsto \mathbb{F} \quad x \boxtimes y = \text{fl}(\text{fl}(x) \circ \text{fl}(y))$$

- Utilizzando la cifra di arrotondamento, è verificata la proprietà seguente:

$$\forall x, y \in \mathbb{R} \quad \exists \delta \in \mathbb{R} \quad \text{t.c.} \quad x \boxtimes y = (x \circ y)(1 + \delta) \quad \text{con} \quad |\delta| \leq u$$

- Nel caso in cui $\circ = +$ (somma), si ha che l'*errore relativo*

$$\frac{|x \boxplus y - (x + y)|}{|x + y|} \leq u(1 + u) \frac{|x| + |y|}{|x + y|} + u$$

Operazioni in virgola mobile

Domanda 1:

Utilizzando $\boxplus: \mathbb{R} \times \mathbb{R} \mapsto \mathbb{F}$ invece di $+: \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$, valgono le stesse proprietà?

- Commutatività: $x \boxplus y \stackrel{?}{=} y \boxplus x$
- Associatività: $(x \boxplus y) \boxplus z \stackrel{?}{=} x \boxplus (y \boxplus z)$

Operazioni in virgola mobile

Domanda 1:

Utilizzando $\boxplus: \mathbb{R} \times \mathbb{R} \mapsto \mathbb{F}$ invece di $+: \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$, valgono le stesse proprietà?

- Commutatività: $x \boxplus y \stackrel{?}{=} y \boxplus x$
- Associatività: $(x \boxplus y) \boxplus z \stackrel{?}{=} x \boxplus (y \boxplus z)$

Domanda 2:

L'operazione di somma $\boxplus: \mathbb{R} \times \mathbb{R} \mapsto \mathbb{F}$ è sempre caratterizzata da un piccolo errore relativo?

Costo computazionale

- Quantità di memoria occupata
- Tempo di calcolo (può essere stimato in base al numero di operazioni elementari: $+$, $-$, \times , \div)

Costo computazionale

- Quantità di memoria occupata
- Tempo di calcolo (può essere stimato in base al numero di operazioni elementari: $+$, $-$, \times , \div)

Complessità algoritmica

- Problema di dimensione n
- Stima del costo computazionale $c(n)$ espressa con la notazione di Landau $O(\cdot)$ (O -grande)

$$c(n) \in O(f(n)) \text{ per } n \rightarrow +\infty \iff \exists n_0, \exists M > 0 \text{ t.c. } |c(n)| < M|f(n)| \forall n > n_0$$

Costo computazionale

- Quantità di memoria occupata
- Tempo di calcolo (può essere stimato in base al numero di operazioni elementari: +, -, ×, ÷)

Complessità algoritmica

- Problema di dimensione n
- Stima del costo computazionale $c(n)$ espressa con la notazione di Landau $O(\cdot)$ (O -grande)

$$c(n) \in O(f(n)) \text{ per } n \rightarrow +\infty \iff \exists n_0, \exists M > 0 \text{ t.c. } |c(n)| < M|f(n)| \forall n > n_0$$

$$\text{equivalente a } \limsup_{n \rightarrow +\infty} \left| \frac{c(n)}{f(n)} \right| < +\infty$$

Costo computazionale

- Quantità di memoria occupata
- Tempo di calcolo (può essere stimato in base al numero di operazioni elementari: +, -, ×, ÷)

Complessità algoritmica

- Problema di dimensione n
- Stima del costo computazionale $c(n)$ espressa con la notazione di Landau $O(\cdot)$ (O -grande)

$$c(n) \in O(f(n)) \text{ per } n \rightarrow +\infty \iff \exists n_0, \exists M > 0 \text{ t.c. } |c(n)| < M|f(n)| \forall n > n_0$$

$$\text{equivalente a } \limsup_{n \rightarrow +\infty} \left| \frac{c(n)}{f(n)} \right| < +\infty$$

Domanda:

Dato un algoritmo poco costoso in tempo e uno poco costoso in memoria, quale scegliereste?

Costo di alcune operazioni tipiche in algebra lineare

- Prodotto vettore-vettore
- Prodotto matrice-vettore
- Prodotto matrice-matrice

Costo di alcune operazioni tipiche in algebra lineare

- Prodotto vettore-vettore $\Rightarrow O(n)$
- Prodotto matrice-vettore $\Rightarrow O(n^2)$
- Prodotto matrice-matrice $\Rightarrow O(n^3)$

Costo di alcune operazioni tipiche in algebra lineare

- Prodotto vettore-vettore $\Rightarrow O(n)$
- Prodotto matrice-vettore $\Rightarrow O(n^2)$
- Prodotto matrice-matrice $\Rightarrow O(n^3)$
- Determinante (formula di Leibnitz)

$$\det(A) = \sum_{\sigma \in S_n} \operatorname{sgn}(\sigma) \prod_{i=1}^n a_{i,\sigma(i)}$$

dove σ è una permutazione dell'insieme $\{1, \dots, n\}$ e S_n l'insieme di tutte le permutazioni dell'insieme $\{1, \dots, n\}$

Costo di alcune operazioni tipiche in algebra lineare

- Prodotto vettore-vettore $\Rightarrow O(n)$
- Prodotto matrice-vettore $\Rightarrow O(n^2)$
- Prodotto matrice-matrice $\Rightarrow O(n^3)$
- Determinante (formula di Leibnitz)

$$\det(A) = \sum_{\sigma \in S_n} \text{sgn}(\sigma) \prod_{i=1}^n a_{i, \sigma(i)}$$

dove σ è una permutazione dell'insieme $\{1, \dots, n\}$ e S_n l'insieme di tutte le permutazioni dell'insieme $\{1, \dots, n\}$

+/-	×
$n!$	$(n-1)n!$

 $\Rightarrow O(nn!)$

Outline

1 Introduzione

2 **Metodi numerici per la risoluzione di sistemi lineari**

- **Metodi diretti**

Risoluzione di sistemi lineari

Dove intervengono

- Soluzione numerica di una EDP \implies soluzione di (almeno) un sistema lineare
- Soluzione di problemi non lineari spesso ricondotta alla risoluzione di una sequenza di sistemi lineari (linearizzazione)
- Derivate di funzionali vincolati, ...

Risoluzione di sistemi lineari

Dove intervengono

- Soluzione numerica di una EDP \implies soluzione di (almeno) un sistema lineare
- Soluzione di problemi non lineari spesso ricondotta alla risoluzione di una sequenza di sistemi lineari (linearizzazione)
- Derivate di funzionali vincolati, ...

La soluzione del sistema lineare è spesso la parte più costosa del calcolo



Necessità di avere a disposizione algoritmi accurati, efficienti e robusti

Risoluzione di sistemi lineari

Il problema

Data una matrice $A \in \mathbb{K}^{n,n}$ (dove $\mathbb{K} = \{\mathbb{R}, \mathbb{C}\}$) e un vettore $\mathbf{b} \in \mathbb{K}^n$, trovare $\mathbf{x} \in \mathbb{K}^n$ tale che

$$A\mathbf{x} = \mathbf{b}$$

Risoluzione di sistemi lineari

Il problema

Data una matrice $A \in \mathbb{K}^{n,n}$ (dove $\mathbb{K} = \{\mathbb{R}, \mathbb{C}\}$) e un vettore $\mathbf{b} \in \mathbb{K}^n$, trovare $\mathbf{x} \in \mathbb{K}^n$ tale che

$$A\mathbf{x} = \mathbf{b}$$

Approccio matematico classico (metodo di Cramer)

Se $\det(A) \neq 0$ allora $\mathbf{x} = A^{-1}\mathbf{b}$, dove

- $A^{-1} = \frac{1}{\det(A)}C^T$, con $C_{ij} = (-1)^{i+j}M_{ij}$
- M_{ij} è il (i,j) -esimo *minore* di A

Stima del costo: $c(n) \simeq O(n^2n!)$ (per i metodi numerici invece: $c(n) \lesssim O(n^3)$)

Risoluzione di sistemi lineari

Il problema

Data una matrice $A \in \mathbb{K}^{n,n}$ (dove $\mathbb{K} = \{\mathbb{R}, \mathbb{C}\}$) e un vettore $\mathbf{b} \in \mathbb{K}^n$, trovare $\mathbf{x} \in \mathbb{K}^n$ tale che

$$A\mathbf{x} = \mathbf{b}$$

Approccio matematico classico (metodo di Cramer)

Se $\det(A) \neq 0$ allora $\mathbf{x} = A^{-1}\mathbf{b}$, dove

- $A^{-1} = \frac{1}{\det(A)} C^T$, con $C_{ij} = (-1)^{i+j} M_{ij}$
- M_{ij} è il (i,j) -esimo *minore* di A

Stima del costo: $c(n) \simeq O(n^2 n!)$ (per i metodi numerici invece: $c(n) \lesssim O(n^3)$)

Esempio: $n = 20$ e $t_x \simeq 10^{-9} s$

Risoluzione di sistemi lineari

Il problema

Data una matrice $A \in \mathbb{K}^{n,n}$ (dove $\mathbb{K} = \{\mathbb{R}, \mathbb{C}\}$) e un vettore $\mathbf{b} \in \mathbb{K}^n$, trovare $\mathbf{x} \in \mathbb{K}^n$ tale che

$$A\mathbf{x} = \mathbf{b}$$

Approccio matematico classico (metodo di Cramer)

Se $\det(A) \neq 0$ allora $\mathbf{x} = A^{-1}\mathbf{b}$, dove

- $A^{-1} = \frac{1}{\det(A)}C^T$, con $C_{ij} = (-1)^{i+j}M_{ij}$
- M_{ij} è il (i,j) -esimo *minore* di A

Stima del costo: $c(n) \simeq O(n^2n!)$ (per i metodi numerici invece: $c(n) \lesssim O(n^3)$)

Esempio: $n = 20$ e $t_x \simeq 10^{-9} s$

- metodo di Cramer: 309 secoli
- metodi numerici: $8.0 \cdot 10^{-6} s$

Risoluzione di sistemi lineari

Il problema

Data una matrice $A \in \mathbb{K}^{n,n}$ (dove $\mathbb{K} = \{\mathbb{R}, \mathbb{C}\}$) e un vettore $\mathbf{b} \in \mathbb{K}^n$, trovare $\mathbf{x} \in \mathbb{K}^n$ tale che

$$A\mathbf{x} = \mathbf{b}$$

Approccio matematico classico (metodo di Cramer)

Se $\det(A) \neq 0$ allora $\mathbf{x} = A^{-1}\mathbf{b}$, dove

- $A^{-1} = \frac{1}{\det(A)}C^T$, con $C_{ij} = (-1)^{i+j}M_{ij}$
- M_{ij} è il (i,j) -esimo *minore* di A

Stima del costo: $c(n) \simeq O(n^2n!)$ (per i metodi numerici invece: $c(n) \lesssim O(n^3)$)

Esempio: $n = 20$ e $t_x \simeq 10^{-9} s$

- metodo di Cramer: 309 secoli
- metodi numerici: $8.0 \cdot 10^{-6} s$

Per i problemi reali si arriva anche a $n = 10^7 - 10^8$

Autovalori e autovettori

- Se $A\mathbf{x} = \lambda\mathbf{x}$ con $\mathbf{x} \neq \mathbf{0}$ \Rightarrow λ è un *autovalore* e \mathbf{x} è un *autovettore*

Autovalori e autovettori

- Se $A\mathbf{x} = \lambda\mathbf{x}$ con $\mathbf{x} \neq \mathbf{0} \Rightarrow \lambda$ è un *autovalore* e \mathbf{x} è un *autovettore*
- L'insieme degli *autovalori* di A è chiamato *spettro* di A e verrà denotato con $\sigma(A)$

Autovalori e autovettori

- Se $A\mathbf{x} = \lambda\mathbf{x}$ con $\mathbf{x} \neq \mathbf{0} \Rightarrow \lambda$ è un *autovalore* e \mathbf{x} è un *autovettore*
- L'insieme degli *autovalori* di A è chiamato *spettro* di A e verrà denotato con $\sigma(A)$
- Gli autovalori sono le soluzioni dell'*equazione caratteristica*

$$p_A(\lambda) = \det(A - \lambda I) = 0$$

dove $p_A(\lambda)$ è chiamato *polinomio caratteristico*

Autovalori e autovettori

- Se $A\mathbf{x} = \lambda\mathbf{x}$ con $\mathbf{x} \neq \mathbf{0} \Rightarrow \lambda$ è un *autovalore* e \mathbf{x} è un *autovettore*
- L'insieme degli *autovalori* di A è chiamato *spettro* di A e verrà denotato con $\sigma(A)$
- Gli autovalori sono le soluzioni dell'*equazione caratteristica*

$$p_A(\lambda) = \det(A - \lambda I) = 0$$

dove $p_A(\lambda)$ è chiamato *polinomio caratteristico*

- $\det(A) = \prod_{i=1}^n \lambda_i$ e $\operatorname{tr}(A) = \sum_{i=1}^n \lambda_i$

Autovalori e autovettori

- Se $A\mathbf{x} = \lambda\mathbf{x}$ con $\mathbf{x} \neq \mathbf{0} \Rightarrow \lambda$ è un *autovalore* e \mathbf{x} è un *autovettore*
- L'insieme degli *autovalori* di A è chiamato *spettro* di A e verrà denotato con $\sigma(A)$
- Gli autovalori sono le soluzioni dell'*equazione caratteristica*

$$p_A(\lambda) = \det(A - \lambda I) = 0$$

dove $p_A(\lambda)$ è chiamato *polinomio caratteristico*

- $\det(A) = \prod_{i=1}^n \lambda_i$ e $\operatorname{tr}(A) = \sum_{i=1}^n \lambda_i$
- $\rho(A) = \max_{\lambda \in \sigma(A)} |\lambda|$ è chiamato *raggio spettrale* di A

Autovalori e autovettori

- Se $A\mathbf{x} = \lambda\mathbf{x}$ con $\mathbf{x} \neq \mathbf{0} \Rightarrow \lambda$ è un *autovalore* e \mathbf{x} è un *autovettore*
- L'insieme degli *autovalori* di A è chiamato *spettro* di A e verrà denotato con $\sigma(A)$
- Gli autovalori sono le soluzioni dell'*equazione caratteristica*

$$p_A(\lambda) = \det(A - \lambda I) = 0$$

dove $p_A(\lambda)$ è chiamato *polinomio caratteristico*

- $\det(A) = \prod_{i=1}^n \lambda_i$ e $\text{tr}(A) = \sum_{i=1}^n \lambda_i$
- $\rho(A) = \max_{\lambda \in \sigma(A)} |\lambda|$ è chiamato *raggio spettrale* di A
- Se A è triangolare i suoi autovalori sono gli elementi sulla diagonale

Decomposizione ai valori singolari (Singular Value Decomposition, SVD)

Sia $A \in \mathbb{C}^{m \times n}$. Esistono allora due matrici unitarie $U \in \mathbb{C}^{m \times m}$ e $V \in \mathbb{C}^{n \times n}$ tali che

$$U^H A V = \Sigma = \text{diag}(\sigma_1, \dots, \sigma_p) \quad \text{con} \quad p = \min(m, n)$$

e $\sigma_1 \geq \dots \geq \sigma_p \geq 0$. I numeri σ_i sono chiamati *valori singolari* di A .

Definizione di prodotto scalare

Un *prodotto scalare* su uno spazio vettoriale V definito su un campo \mathbb{K} è un'applicazione $\langle \cdot, \cdot \rangle: V \times V \mapsto \mathbb{K}$ che soddisfa le seguenti proprietà:

- 1 è *lineare* rispetto ai vettori di V , cioè:

$$\langle \gamma \mathbf{x} + \lambda \mathbf{z}, \mathbf{y} \rangle = \gamma \langle \mathbf{x}, \mathbf{y} \rangle + \lambda \langle \mathbf{z}, \mathbf{y} \rangle, \quad \forall \mathbf{x}, \mathbf{y}, \mathbf{z} \in V, \quad \forall \gamma, \lambda \in \mathbb{K}$$

- 2 è *hermitiana*, cioè $\langle \mathbf{x}, \mathbf{y} \rangle = \overline{\langle \mathbf{y}, \mathbf{x} \rangle}$, $\forall \mathbf{x}, \mathbf{y} \in V$

- 3 è *definita positiva*, cioè: $\langle \mathbf{x}, \mathbf{x} \rangle \geq 0$ e $\langle \mathbf{x}, \mathbf{x} \rangle = 0$ se e solo se $\mathbf{x} = \mathbf{0}$

Definizione di norma

Sia V uno spazio vettoriale su un campo \mathbb{K} . Diremo che l'applicazione $\|\cdot\|: V \mapsto \mathbb{K}$ è una *norma* su V se sono verificate le seguenti proprietà

- 1 $\|\mathbf{v}\| \geq 0 \quad \forall \mathbf{v} \in V$ e $\|\mathbf{v}\| = 0$ se e solo se $\mathbf{v} = \mathbf{0}$
- 2 $\|\alpha \mathbf{v}\| = |\alpha| \|\mathbf{v}\| \quad \forall \alpha \in \mathbb{K}, \forall \mathbf{v} \in V$
- 3 $\|\mathbf{v} + \mathbf{w}\| \leq \|\mathbf{v}\| + \|\mathbf{w}\| \quad \forall \mathbf{v}, \mathbf{w} \in V$ (*disuguaglianza triangolare*)

Definizione di norma

Sia V uno spazio vettoriale su un campo \mathbb{K} . Diremo che l'applicazione $\|\cdot\|: V \mapsto \mathbb{K}$ è una *norma* su V se sono verificate le seguenti proprietà

- 1 $\|\mathbf{v}\| \geq 0 \quad \forall \mathbf{v} \in V$ e $\|\mathbf{v}\| = 0$ se e solo se $\mathbf{v} = \mathbf{0}$
- 2 $\|\alpha \mathbf{v}\| = |\alpha| \|\mathbf{v}\| \quad \forall \alpha \in \mathbb{K}, \forall \mathbf{v} \in V$
- 3 $\|\mathbf{v} + \mathbf{w}\| \leq \|\mathbf{v}\| + \|\mathbf{w}\| \quad \forall \mathbf{v}, \mathbf{w} \in V$ (*disuguaglianza triangolare*)

Esempi di norme ($\mathbf{x} \in \mathbb{R}^n$)

- *p-norma:*

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^n |x_i|^p \right)^{\frac{1}{p}} \quad 1 \leq p < \infty$$

- *norma del massimo (o norma infinito)*

$$\|\mathbf{x}\|_\infty = \max_{1 \leq i \leq n} |x_i|$$

Disuguaglianza di Cauchy-Schwarz

Per ogni coppia $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ si ha

$$|\langle \mathbf{x}, \mathbf{y} \rangle| = |\mathbf{x}^T \mathbf{y}| \leq \|\mathbf{x}\|_2 \|\mathbf{y}\|_2$$

Disuguaglianza di Cauchy-Schwarz

Per ogni coppia $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ si ha

$$|\langle \mathbf{x}, \mathbf{y} \rangle| = |\mathbf{x}^T \mathbf{y}| \leq \|\mathbf{x}\|_2 \|\mathbf{y}\|_2$$

Disuguaglianza di Hölder

Per ogni coppia $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ si ha

$$|\langle \mathbf{x}, \mathbf{y} \rangle| \leq \|\mathbf{x}\|_p \|\mathbf{y}\|_q, \quad \text{con } \frac{1}{p} + \frac{1}{q} = 1$$

Disuguaglianza di Cauchy-Schwarz

Per ogni coppia $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ si ha

$$|\langle \mathbf{x}, \mathbf{y} \rangle| = |\mathbf{x}^T \mathbf{y}| \leq \|\mathbf{x}\|_2 \|\mathbf{y}\|_2$$

Disuguaglianza di Hölder

Per ogni coppia $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ si ha

$$|\langle \mathbf{x}, \mathbf{y} \rangle| \leq \|\mathbf{x}\|_p \|\mathbf{y}\|_q, \quad \text{con } \frac{1}{p} + \frac{1}{q} = 1$$

Equivalenza fra norme

Due norme $\|\cdot\|_p$ e $\|\cdot\|_q$ su V sono *equivalenti* se esistono due costanti positive c_{pq} e C_{pq} tali che

$$c_{pq} \|\mathbf{x}\|_q \leq \|\mathbf{x}\|_p \leq C_{pq} \|\mathbf{x}\|_q \quad \forall \mathbf{x} \in V$$

Disuguaglianza di Cauchy-Schwarz

Per ogni coppia $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ si ha

$$|\langle \mathbf{x}, \mathbf{y} \rangle| = |\mathbf{x}^T \mathbf{y}| \leq \|\mathbf{x}\|_2 \|\mathbf{y}\|_2$$

Disuguaglianza di Hölder

Per ogni coppia $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ si ha

$$|\langle \mathbf{x}, \mathbf{y} \rangle| \leq \|\mathbf{x}\|_p \|\mathbf{y}\|_q, \quad \text{con } \frac{1}{p} + \frac{1}{q} = 1$$

Equivalenza fra norme

Due norme $\|\cdot\|_p$ e $\|\cdot\|_q$ su V sono *equivalenti* se esistono due costanti positive c_{pq} e C_{pq} tali che

$$c_{pq} \|\mathbf{x}\|_q \leq \|\mathbf{x}\|_p \leq C_{pq} \|\mathbf{x}\|_q \quad \forall \mathbf{x} \in V$$

In uno spazio normato di dimensione finita, tutte le norme sono equivalenti

Norme di matrici

- *Norma di Frobenius (o norma Euclidea)*

$$\|A\|_F = \sqrt{\sum_{i,j=1}^n |a_{ij}|^2} = \text{tr}(AA^H)$$

- *Norma naturale (o norma indotta)*

$$\|A\|_p = \sup_{\mathbf{x} \neq \mathbf{0}} \frac{\|A\mathbf{x}\|_p}{\|\mathbf{x}\|_p}$$

- $\|A\|_1 = \max_{j=1, \dots, n} \sum_{i=1}^m |a_{ij}|$
- $\|A\|_\infty = \max_{i=1, \dots, m} \sum_{j=1}^n |a_{ij}|$
- se A è reale e simmetrica, allora $\|A\|_2 = \rho(A)$

Stime per $\|\cdot\|_2$

$$\max_{i,j} |a_{ij}| \leq \|A\|_2 \leq n \max_{i,j} |a_{ij}|$$

$$\frac{1}{\sqrt{n}} \|A\|_\infty \leq \|A\|_2 \leq \sqrt{n} \|A\|_\infty$$

$$\frac{1}{\sqrt{n}} \|A\|_1 \leq \|A\|_2 \leq \sqrt{n} \|A\|_1$$

$$\|A\|_2 \leq \sqrt{\|A\|_1 \|A\|_\infty}$$

Stime per $\|\cdot\|_2$

$$\max_{i,j} |a_{ij}| \leq \|A\|_2 \leq n \max_{i,j} |a_{ij}|$$

$$\frac{1}{\sqrt{n}} \|A\|_\infty \leq \|A\|_2 \leq \sqrt{n} \|A\|_\infty$$

$$\frac{1}{\sqrt{n}} \|A\|_1 \leq \|A\|_2 \leq \sqrt{n} \|A\|_1$$

$$\|A\|_2 \leq \sqrt{\|A\|_1 \|A\|_\infty}$$

Norme naturali: proprietà

$$\bullet \|A\mathbf{x}\|_p \leq \|A\|_p \|\mathbf{x}\|_p$$

$$\bullet \|I\|_p = 1$$

$$\bullet \|AB\|_p \leq \|A\|_p \|B\|_p$$

$$\bullet \rho(A) \leq \|A\|_p$$

Norme *consistenti*

Una norma di matrice $|||\cdot|||$ si dice *consistente* con una norma di vettore $||\cdot||$ se

$$||\mathbf{Ax}|| \leq |||\mathbf{A}||| ||\mathbf{x}|| \quad \forall \mathbf{x} \in \mathbb{R}^n$$

Norme consistenti

Una norma di matrice $\|\cdot\|$ si dice *consistente* con una norma di vettore $\|\cdot\|$ se

$$\|\mathbf{Ax}\| \leq \|A\| \|\mathbf{x}\| \quad \forall \mathbf{x} \in \mathbb{R}^n$$

- Le norme naturali sono norme consistenti

Norme consistenti

Una norma di matrice $\|\cdot\|$ si dice *consistente* con una norma di vettore $\|\cdot\|$ se

$$\|\mathbf{Ax}\| \leq \|A\| \|\mathbf{x}\| \quad \forall \mathbf{x} \in \mathbb{R}^n$$

- Le norme naturali sono norme consistenti
- Sia $\|\cdot\|$ una norma consistente. Allora $\rho(A) \leq \|A\|$

Norme consistenti

Una norma di matrice $\|\cdot\|$ si dice *consistente* con una norma di vettore $\|\cdot\|$ se

$$\|\mathbf{Ax}\| \leq \|A\| \|\mathbf{x}\| \quad \forall \mathbf{x} \in \mathbb{R}^n$$

- Le norme naturali sono norme consistenti
- Sia $\|\cdot\|$ una norma consistente. Allora $\rho(A) \leq \|A\|$
- Sia $\varepsilon > 0$. Allora esiste una norma consistente $\|\cdot\|_{A,\varepsilon}$ tale che $\|A\|_{A,\varepsilon} \leq \rho(A) + \varepsilon$

Norme consistenti

Una norma di matrice $\|\cdot\|$ si dice *consistente* con una norma di vettore $\|\cdot\|$ se

$$\|\mathbf{Ax}\| \leq \|A\| \|\mathbf{x}\| \quad \forall \mathbf{x} \in \mathbb{R}^n$$

- Le norme naturali sono norme consistenti
- Sia $\|\cdot\|$ una norma consistente. Allora $\rho(A) \leq \|A\|$
- Sia $\varepsilon > 0$. Allora esiste una norma consistente $\|\cdot\|_{A,\varepsilon}$ tale che $\|A\|_{A,\varepsilon} \leq \rho(A) + \varepsilon$
- $\rho(A) = \inf_{\|\cdot\|} \|A\|$

Norme consistenti

Una norma di matrice $\|\cdot\|$ si dice *consistente* con una norma di vettore $\|\cdot\|$ se

$$\|A\mathbf{x}\| \leq \|A\| \|\mathbf{x}\| \quad \forall \mathbf{x} \in \mathbb{R}^n$$

- Le norme naturali sono norme consistenti
- Sia $\|\cdot\|$ una norma consistente. Allora $\rho(A) \leq \|A\|$
- Sia $\varepsilon > 0$. Allora esiste una norma consistente $\|\cdot\|_{A,\varepsilon}$ tale che $\|A\|_{A,\varepsilon} \leq \rho(A) + \varepsilon$
- $\rho(A) = \inf_{\|\cdot\|} \|A\|$

Potenze di matrici: convergenza

$$\lim_{k \rightarrow \infty} A^k = 0 \quad \Leftrightarrow \quad \rho(A) < 1$$

Condizionamento e stabilità

Numero di condizionamento

$$K_p(A) = \|A\|_p \|A^{-1}\|_p$$

dove $\|A\|_p = \sup_{\mathbf{x} \neq \mathbf{0}} \frac{\|A\mathbf{x}\|_p}{\|\mathbf{x}\|_p}$. Si noti che $K_p(A) \geq 1$.

Condizionamento e stabilità

Numero di condizionamento

$$K_p(A) = \|A\|_p \|A^{-1}\|_p$$

dove $\|A\|_p = \sup_{\mathbf{x} \neq \mathbf{0}} \frac{\|A\mathbf{x}\|_p}{\|\mathbf{x}\|_p}$. Si noti che $K_p(A) \geq 1$.

Analisi *a priori*

Come reagisce la soluzione *esatta* di un sistema lineare $A\mathbf{x} = \mathbf{b}$ al variare dei dati?

Condizionamento e stabilità

Numero di condizionamento

$$K_p(A) = \|A\|_p \|A^{-1}\|_p$$

dove $\|A\|_p = \sup_{\mathbf{x} \neq \mathbf{0}} \frac{\|A\mathbf{x}\|_p}{\|\mathbf{x}\|_p}$. Si noti che $K_p(A) \geq 1$.

Analisi *a priori*

Come reagisce la soluzione *esatta* di un sistema lineare $A\mathbf{x} = \mathbf{b}$ al variare dei dati?

- Perturbazione di \mathbf{b} : $A(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b}$

$$\frac{1}{K_p(A)} \frac{\|\delta\mathbf{b}\|_p}{\|\mathbf{b}\|_p} \leq \frac{\|\delta\mathbf{x}\|_p}{\|\mathbf{x}\|_p} \leq K_p(A) \frac{\|\delta\mathbf{b}\|_p}{\|\mathbf{b}\|_p}$$

Condizionamento e stabilità

Numero di condizionamento

$$K_p(A) = \|A\|_p \|A^{-1}\|_p$$

dove $\|A\|_p = \sup_{\mathbf{x} \neq \mathbf{0}} \frac{\|A\mathbf{x}\|_p}{\|\mathbf{x}\|_p}$. Si noti che $K_p(A) \geq 1$.

Analisi a priori

Come reagisce la soluzione *esatta* di un sistema lineare $A\mathbf{x} = \mathbf{b}$ al variare dei dati?

- Perturbazione di \mathbf{b} : $A(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b}$

$$\frac{1}{K_p(A)} \frac{\|\delta\mathbf{b}\|_p}{\|\mathbf{b}\|_p} \leq \frac{\|\delta\mathbf{x}\|_p}{\|\mathbf{x}\|_p} \leq K_p(A) \frac{\|\delta\mathbf{b}\|_p}{\|\mathbf{b}\|_p}$$

- Perturbazione di A : $(A + \delta A)(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b}$

$$\frac{\|\delta\mathbf{x}\|_p}{\|\mathbf{x}\|_p} \leq \frac{K_p(A)}{1 - K_p(A)\|\delta A\|_p/\|A\|_p} \frac{\|\delta A\|_p}{\|A\|_p}$$

Condizionamento e stabilità

- $K_p(A) \simeq 1 \Rightarrow$ le perturbazioni sui dati non influenzano “eccessivamente” i risultati \Rightarrow sistema *ben condizionato*
- Nella risoluzione numerica di una EDP il tipo di matrice che si ottiene dipende:
 - dal problema
 - dal metodo di discretizzazione usato
 - dalla “qualità” della griglia di calcolo

Condizionamento e stabilità

- $K_p(A) \simeq 1 \Rightarrow$ le perturbazioni sui dati non influenzano “eccessivamente” i risultati \Rightarrow sistema *ben condizionato*
- Nella risoluzione numerica di una EDP il tipo di matrice che si ottiene dipende:
 - dal problema
 - dal metodo di discretizzazione usato
 - dalla “qualità” della griglia di calcolo

Precondizionamento

Invece del sistema $A\mathbf{x} = \mathbf{b}$ si risolve il sistema equivalente

$$PA\mathbf{x} = P\mathbf{b}$$

dove la matrice PA è ben condizionata: $K_p(PA) \ll K_p(A)$.

Condizionamento e stabilità

- $K_p(A) \simeq 1 \Rightarrow$ le perturbazioni sui dati non influenzano “eccessivamente” i risultati \Rightarrow sistema *ben condizionato*
- Nella risoluzione numerica di una EDP il tipo di matrice che si ottiene dipende:
 - dal problema
 - dal metodo di discretizzazione usato
 - dalla “qualità” della griglia di calcolo

Precondizionamento

Invece del sistema $A\mathbf{x} = \mathbf{b}$ si risolve il sistema equivalente

$$PA\mathbf{x} = P\mathbf{b}$$

dove la matrice PA è ben condizionata: $K_p(PA) \ll K_p(A)$.

- Minore propagazione degli errori di arrotondamento
- Aumento della velocità di convergenza dei metodi di risoluzione iterativi

Condizionamento e stabilità

Stabilità di un algoritmo

Si tratta di sapere quanto la soluzione di un sistem lineare ottenuta con un dato algoritmo è sensibile a piccoli errori sui dati A e \mathbf{b} .

- Un algoritmo è detto *stabile* se non amplifica “eccessivamente” le perturbazioni sui dati
- Due nozioni complementari:
 - *condizionamento di una matrice*, che esprime la sensibilità della soluzione esatta agli errori di arrotondamento
 - *stabilità di un algoritmo*, che è legata all’amplificazione degli errori dovuta all’agoritmo di risoluzione usato
- per una buona risoluzione di un sistema lineare bisogna avere una matrice non troppo mal condizionata e un algoritmo sifficientemente stabile

Metodi di risoluzione

Classificazione

- **METODI DIRETTI:** in assenza di errori di arrotondamento, danno la soluzione esatta in un numero finito di operazioni
- **METODI ITERATIVI:** la soluzione è ottenuta come limite di una successione
- **METODI MISTI**

Fattorizzazione LU ed eliminazione gaussiana

Fattorizzazione LU

Considerare la matrice del sistema lineare $A\mathbf{x} = \mathbf{b}$ come prodotto di due matrici

$$A = LU$$

dove

- L è *triangolare inferiore*: $L_{ij} = 0$ per $j > i$ (L = lower)
- U è *triangolare superiore*: $U_{ij} = 0$ per $i > j$ (U = upper)

Fattorizzazione LU ed eliminazione gaussiana

Fattorizzazione LU

Considerare la matrice del sistema lineare $A\mathbf{x} = \mathbf{b}$ come prodotto di due matrici

$$A = LU$$

dove

- L è *triangolare inferiore*: $L_{ij} = 0$ per $j > i$ (L = lower)
- U è *triangolare superiore*: $U_{ij} = 0$ per $i > j$ (U = upper)

La soluzione del sistema $A\mathbf{x} = \mathbf{b}$ è ottenuta risolvendo due sistemi triangolari

- 1 $L\mathbf{y} = \mathbf{b}$
- 2 $U\mathbf{x} = \mathbf{y}$

Fattorizzazione LU ed eliminazione gaussiana

Fattorizzazione LU

Considerare la matrice del sistema lineare $A\mathbf{x} = \mathbf{b}$ come prodotto di due matrici

$$A = LU$$

dove

- L è *triangolare inferiore*: $L_{ij} = 0$ per $j > i$ (L = lower)
- U è *triangolare superiore*: $U_{ij} = 0$ per $i > j$ (U = upper)

La soluzione del sistema $A\mathbf{x} = \mathbf{b}$ è ottenuta risolvendo due sistemi triangolari

- 1 $L\mathbf{y} = \mathbf{b}$
- 2 $U\mathbf{x} = \mathbf{y}$

Vantaggio

L'inversione di una matrice triangolare è facile e poco costosa: $O(n^2)$

Soluzione di sistemi triangolari

- Matrici triangolari inferiori ($L\mathbf{x} = \mathbf{b}$) : *sostituzione in avanti*

$$x_1 = \frac{b_1}{l_{11}}$$

$$x_i = \frac{1}{l_{ii}} \left(b_i - \sum_{j=1}^{i-1} l_{ij} x_j \right) \quad i = 2, \dots, n$$

Soluzione di sistemi triangolari

- Matrici triangolari inferiori ($L\mathbf{x} = \mathbf{b}$): *sostituzione in avanti*

$$x_1 = \frac{b_1}{l_{11}}$$

$$x_i = \frac{1}{l_{ii}} \left(b_i - \sum_{j=1}^{i-1} l_{ij} x_j \right) \quad i = 2, \dots, n$$

- Matrici triangolari superiori ($U\mathbf{x} = \mathbf{b}$): *sostituzione all'indietro*

$$x_n = \frac{b_n}{u_{nn}}$$

$$x_i = \frac{1}{u_{ii}} \left(b_i - \sum_{j=i+1}^n u_{ij} x_j \right) \quad i = n-1, \dots, 1$$

Soluzione di sistemi triangolari

- Matrici triangolari inferiori ($L\mathbf{x} = \mathbf{b}$): *sostituzione in avanti*

$$x_1 = \frac{b_1}{l_{11}}$$

$$x_i = \frac{1}{l_{ii}} \left(b_i - \sum_{j=1}^{i-1} l_{ij} x_j \right) \quad i = 2, \dots, n$$

- Matrici triangolari superiori ($U\mathbf{x} = \mathbf{b}$): *sostituzione all'indietro*

$$x_n = \frac{b_n}{u_{nn}}$$

$$x_i = \frac{1}{u_{ii}} \left(b_i - \sum_{j=i+1}^n u_{ij} x_j \right) \quad i = n-1, \dots, 1$$

- Costo computazionale:

+/-	\times	\div
$\frac{n(n-1)}{2}$	$\frac{n(n-1)}{2}$	n

$$\Rightarrow O(n^2)$$

Eliminazione gaussiana

Algoritmo che permette di ridurre il sistema $A\mathbf{x} = \mathbf{b}$ in un sistema equivalente $U\mathbf{x} = \hat{\mathbf{b}}$, dove U è una matrice triangolare superiore.

Eliminazione gaussiana

Algoritmo che permette di ridurre il sistema $\mathbf{Ax} = \mathbf{b}$ in un sistema equivalente $U\mathbf{x} = \hat{\mathbf{b}}$, dove U è una matrice triangolare superiore.

- Sia $A^{(1)} = A$, $\mathbf{b}^{(1)} = \mathbf{b}$ e supponiamo che $a_{11}^{(1)} \neq 0$

Eliminazione gaussiana

Algoritmo che permette di ridurre il sistema $A\mathbf{x} = \mathbf{b}$ in un sistema equivalente $U\mathbf{x} = \hat{\mathbf{b}}$, dove U è una matrice triangolare superiore.

- Sia $A^{(1)} = A$, $\mathbf{b}^{(1)} = \mathbf{b}$ e supponiamo che $a_{11}^{(1)} \neq 0$
- Introduciamo i *moltiplicatori* $m_{i1} = \frac{a_{i1}^{(1)}}{a_{11}^{(1)}} \quad (i = 2, 3, \dots, n)$

Eliminazione gaussiana

Algoritmo che permette di ridurre il sistema $A\mathbf{x} = \mathbf{b}$ in un sistema equivalente $U\mathbf{x} = \hat{\mathbf{b}}$, dove U è una matrice triangolare superiore.

- Sia $A^{(1)} = A$, $\mathbf{b}^{(1)} = \mathbf{b}$ e supponiamo che $a_{11}^{(1)} \neq 0$
- Introduciamo i *moltiplicatori* $m_{i1} = \frac{a_{i1}^{(1)}}{a_{11}^{(1)}} \quad (i = 2, 3, \dots, n)$
- Sottraendo dalla riga i la prima riga moltiplicata per m_{i1} si ha il sistema equivalente $A^{(2)}\mathbf{x} = \mathbf{b}^{(2)}$, dove

$$A^{(2)} = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \dots & a_{2n}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(k)} & \dots & a_{nn}^{(k)} \end{pmatrix} \quad \text{e} \quad \mathbf{b}^{(2)} = \begin{pmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_n^{(2)} \end{pmatrix}$$

Eliminazione gaussiana

Algoritmo che permette di ridurre il sistema $A\mathbf{x} = \mathbf{b}$ in un sistema equivalente $U\mathbf{x} = \hat{\mathbf{b}}$, dove U è una matrice triangolare superiore.

- Sia $A^{(1)} = A$, $\mathbf{b}^{(1)} = \mathbf{b}$ e supponiamo che $a_{11}^{(1)} \neq 0$
- Introduciamo i *moltiplicatori* $m_{i1} = \frac{a_{i1}^{(1)}}{a_{11}^{(1)}} \quad (i = 2, 3, \dots, n)$
- Sottraendo dalla riga i la prima riga moltiplicata per m_{i1} si ha il sistema equivalente $A^{(2)}\mathbf{x} = \mathbf{b}^{(2)}$, dove

$$A^{(2)} = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \dots & a_{2n}^{(2)} \\ \vdots & \vdots & & \vdots \\ 0 & a_{n2}^{(k)} & \dots & a_{nn}^{(k)} \end{pmatrix} \quad \text{e} \quad \mathbf{b}^{(2)} = \begin{pmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_n^{(2)} \end{pmatrix}$$

- Ripetere il procedimento per le righe $i = 2, 3, \dots, n$

Eliminazione di Gaussiana

- All'ultima iterazione abbiamo

$$\begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \dots & \dots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & & & a_{2n}^{(2)} \\ \vdots & & \ddots & & \vdots \\ 0 & & & \ddots & \vdots \\ 0 & & & & a_{nn}^{(n)} \end{pmatrix} \mathbf{x} = \begin{pmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ \vdots \\ b_n^{(n)} \end{pmatrix}$$

- Ricapitolando, per $k = 1, 2, \dots, n$

$$\begin{cases} m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} & i = k + 1, \dots, n \\ a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)} & i, j = k + 1, \dots, n \\ b_i^{(k+1)} = b_i^{(k)} - m_{ik} b_k^{(k)} & i = k + 1, \dots, n \end{cases}$$

Eliminazione gaussiana

Costo computazionale

Per $k = 1, 2, \dots, n$

$$\left\{ \begin{array}{ll} m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} & i = k + 1, \dots, n \\ a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)} & i, j = k + 1, \dots, n \\ b_i^{(k+1)} = b_i^{(k)} - m_{ik} b_k^{(k)} & i = k + 1, \dots, n \end{array} \right.$$

+	-	×	÷
0	$\frac{n(n+1)(n-1)}{3}$	$\frac{n(n+1)(n-1)}{3}$	$\frac{n(n-1)}{2}$

$\Rightarrow O(n^3)$

Eliminazione gaussiana

Pivoting

- L'eliminazione gaussiana può essere usata solo se $a_{kk}^{(k)} \neq 0$ (*pivot*)

Eliminazione gaussiana

Pivoting

- L'eliminazione gaussiana può essere usata solo se $a_{kk}^{(k)} \neq 0$ (*pivot*)
- Se $a_{kk}^{(k)} = 0$ si permuta la riga k con un'altra riga j ($k < j \leq n$) dove $a_{jk}^{(k)} \neq 0$

Eliminazione gaussiana

Pivoting

- L'eliminazione gaussiana può essere usata solo se $a_{kk}^{(k)} \neq 0$ (*pivot*)
- Se $a_{kk}^{(k)} = 0$ si permuta la riga k con un'altra riga j ($k < j \leq n$) dove $a_{jk}^{(k)} \neq 0$
- Se $a_{jk}^{(k)} = 0$ ($k \leq j \leq n$) allora la matrice A è *singolare*

Eliminazione gaussiana

Pivoting

- L'eliminazione gaussiana può essere usata solo se $a_{kk}^{(k)} \neq 0$ (*pivot*)
- Se $a_{kk}^{(k)} = 0$ si permuta la riga k con un'altra riga j ($k < j \leq n$) dove $a_{jk}^{(k)} \neq 0$
- Se $a_{jk}^{(k)} = 0$ ($k \leq j \leq n$) allora la matrice A è *singolare*
- Corollario: l'eliminazione gaussiana associata a una strategia di pivot è sempre possibile per una matrice non singolare

Eliminazione gaussiana

Pivoting

- L'eliminazione gaussiana può essere usata solo se $a_{kk}^{(k)} \neq 0$ (*pivot*)
- Se $a_{kk}^{(k)} = 0$ si permuta la riga k con un'altra riga j ($k < j \leq n$) dove $a_{jk}^{(k)} \neq 0$
- Se $a_{jk}^{(k)} = 0$ ($k \leq j \leq n$) allora la matrice A è *singolare*
- Corollario: l'eliminazione gaussiana associata a una strategia di pivot è sempre possibile per una matrice non singolare
- Alcune matrici per cui l'eliminazione gaussiana può essere applicata *senza pivoting* sono:
 - Matrici a diagonale dominante per righe (matrici per cui $|a_{ii}| \geq \sum_{j=1, j \neq i} |a_{ij}|$ con $i = 1, \dots, n$)
 - Matrici a diagonale dominante per colonne (matrici per cui $|a_{ii}| \geq \sum_{j=1, j \neq i} |a_{ji}|$ con $i = 1, \dots, n$)
 - Matrici simmetriche e definite positive

Eliminazione gaussiana e fattorizzazione LU

Equivalenza tra l'eliminazione gaussiana e la fattorizzazione LU

- Sia $N^{(i,j)}$ la matrice con elementi nulli tranne l'elemento nella posizione (i,j) pari a 1

Eliminazione gaussiana e fattorizzazione LU

Equivalenza tra l'eliminazione gaussiana e la fattorizzazione LU

- Sia $N^{(i,j)}$ la matrice con elementi nulli tranne l'elemento nella posizione (i,j) pari a 1
- $N^{(i,j)}N^{(r,s)} \neq 0 \Leftrightarrow j = r$

Eliminazione gaussiana e fattorizzazione LU

Equivalenza tra l'eliminazione gaussiana e la fattorizzazione LU

- Sia $N^{(i,j)}$ la matrice con elementi nulli tranne l'elemento nella posizione (i,j) pari a 1
- $N^{(i,j)}N^{(r,s)} \neq 0 \iff j = r$
- Sommare alla i -esima riga α -volte la riga j , equivale a moltiplicare a sinistra per la matrice $I + \alpha N^{(i,j)}$

Eliminazione gaussiana e fattorizzazione LU

Equivalenza tra l'eliminazione gaussiana e la fattorizzazione LU

- Sia $N^{(i,j)}$ la matrice con elementi nulli tranne l'elemento nella posizione (i,j) pari a 1
- $N^{(i,j)}N^{(r,s)} \neq 0 \Leftrightarrow j = r$
- Sommare alla i -esima riga α -volte la riga j , equivale a moltiplicare a sinistra per la matrice $I + \alpha N^{(i,j)}$
- Eliminazione degli elementi sotto $(a_{kk}^{(k)})$

$$\begin{aligned}(I - m_{k+1,k}N^{(k+1,k)})(I - m_{k+2,k}N^{(k+2,k)}) \dots (I - m_{n,k}N^{(n,k)})A^{(k)} &= \\ (I - \sum_{i=k+1}^n m_{i,k}N^{(i,k)})A^{(k)} &= \\ M_k A^{(k)} &= A^{(k+1)}\end{aligned}$$

Eliminazione gaussiana e fattorizzazione LU

Equivalenza tra l'eliminazione gaussiana e la fattorizzazione LU

- Sia $N^{(i,j)}$ la matrice con elementi nulli tranne l'elemento nella posizione (i,j) pari a 1
- $N^{(i,j)}N^{(r,s)} \neq 0 \Leftrightarrow j = r$
- Sommare alla i -esima riga α -volte la riga j , equivale a moltiplicare a sinistra per la matrice $I + \alpha N^{(i,j)}$
- Eliminazione degli elementi sotto $(a_{kk}^{(k)})$

$$\begin{aligned} (I - m_{k+1,k}N^{(k+1,k)})(I - m_{k+2,k}N^{(k+2,k)}) \dots (I - m_{n,k}N^{(n,k)})A^{(k)} &= \\ (I - \sum_{i=k+1}^n m_{i,k}N^{(i,k)})A^{(k)} &= \\ M_k A^{(k)} &= A^{(k+1)} \end{aligned}$$

- In definitiva si ha: $M_{n-1}M_{n-2} \dots M_1 A = MA = U$

Eliminazione gaussiana e fattorizzazione LU

$$M_1 = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ -m_{21} & 1 & & & \vdots \\ \vdots & 0 & 1 & & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -m_{n1} & 0 & 0 & \dots & 1 \end{pmatrix}$$

$$M_2 = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & & & \vdots \\ \vdots & -m_{32} & 1 & & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -m_{n2} & 0 & \dots & 1 \end{pmatrix}$$

Eliminazione gaussiana e fattorizzazione LU

$$M_1 = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ -m_{21} & 1 & & & \vdots \\ \vdots & 0 & 1 & & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -m_{n1} & 0 & 0 & \dots & 1 \end{pmatrix} \quad M_2 = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & & & \vdots \\ \vdots & -m_{32} & 1 & & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & -m_{n2} & 0 & \dots & 1 \end{pmatrix}$$

$$M_2 M_1 = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ -m_{21} & 1 & & & \vdots \\ \vdots & -m_{32} & 1 & & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -m_{n1} & -m_{n2} & 0 & \dots & 1 \end{pmatrix}$$

Eliminazione gaussiana e fattorizzazione LU

$$M = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ -m_{21} & 1 & & & \vdots \\ \vdots & -m_{32} & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ -m_{n1} & -m_{n2} & \dots & -m_{n,n-1} & 1 \end{pmatrix}$$

Eliminazione gaussiana e fattorizzazione LU

$$M = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ -m_{21} & 1 & & & \vdots \\ \vdots & -m_{32} & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ -m_{n1} & -m_{n2} & \dots & -m_{n,n-1} & 1 \end{pmatrix}$$

$$M^{-1} = 2I - M = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ m_{21} & 1 & & & \vdots \\ \vdots & m_{32} & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ m_{n1} & m_{n2} & \dots & m_{n,n-1} & 1 \end{pmatrix}$$

Eliminazione gaussiana e fattorizzazione LU

$$M = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ -m_{21} & 1 & & & \vdots \\ \vdots & -m_{32} & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ -m_{n1} & -m_{n2} & \dots & -m_{n,n-1} & 1 \end{pmatrix}$$

$$M^{-1} = 2I - M = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ m_{21} & 1 & & & \vdots \\ \vdots & m_{32} & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ m_{n1} & m_{n2} & \dots & m_{n,n-1} & 1 \end{pmatrix}$$

Poiché $A = M^{-1}U$ si ha

$$L = M^{-1}$$

...ancora riguardo al pivoting

- P_k = Matrice di permutazione tra la riga k e la riga $i_k \geq k$
- $(P_k)^{-1} = (P_k)^T = P_k$

...ancora riguardo al pivoting

- P_k = Matrice di permutazione tra la riga k e la riga $i_k \geq k$
- $(P_k)^{-1} = (P_k)^T = P_k$
- Applicazione di P_k prima di $A^{(k)}$

$$M_{n-1}P_{n-1}M_{n-2}P_{n-2}\dots M_1P_1A = MA = U$$

- M^{-1} non è più triangolare inferiore

...ancora riguardo al pivoting

- P_k = Matrice di permutazione tra la riga k e la riga $i_k \geq k$
- $(P_k)^{-1} = (P_k)^T = P_k$
- Applicazione di P_k prima di $A^{(k)}$

$$M_{n-1}P_{n-1}M_{n-2}P_{n-2}\dots M_1P_1A = MA = U$$

- M^{-1} non è più triangolare inferiore
- Introducendo $P = P_{n-1}\dots P_1$ si ha

$$PA = (PM^{-1})U$$

dove (PM^{-1}) è triangolare inferiore \Rightarrow poniamo $L = PM^{-1}$

...ancora riguardo al pivoting

- P_k = Matrice di permutazione tra la riga k e la riga $i_k \geq k$
- $(P_k)^{-1} = (P_k)^T = P_k$
- Applicazione di P_k prima di $A^{(k)}$

$$M_{n-1}P_{n-1}M_{n-2}P_{n-2}\dots M_1P_1A = MA = U$$

- M^{-1} non è più triangolare inferiore
- Introducendo $P = P_{n-1}\dots P_1$ si ha

$$PA = (PM^{-1})U$$

dove (PM^{-1}) è triangolare inferiore \Rightarrow poniamo $L = PM^{-1}$

- L e U sono calcolate con l'eliminazione gaussiana partendo da PA

...ancora riguardo al pivoting

- P_k = Matrice di permutazione tra la riga k e la riga $i_k \geq k$
- $(P_k)^{-1} = (P_k)^T = P_k$
- Applicazione di P_k prima di $A^{(k)}$

$$M_{n-1}P_{n-1}M_{n-2}P_{n-2}\dots M_1P_1A = MA = U$$

- M^{-1} non è più triangolare inferiore
- Introducendo $P = P_{n-1}\dots P_1$ si ha

$$PA = (PM^{-1})U$$

dove (PM^{-1}) è triangolare inferiore \Rightarrow poniamo $L = PM^{-1}$

- L e U sono calcolate con l'eliminazione gaussiana partendo da PA
- Una volta ottenuto P , L e U si risolvono
 - 1 $Ly = Pb$
 - 2 $Ux = y$

Algoritmo di Doolittle (Fattorizzazione compatta)

- La fattorizzazione LU equivale a trovare i coefficienti l_{ir} e u_{rj} che soddisfano

$$a_{ij} = \sum_{r=1}^{\min(i,j)} l_{ir} u_{rj}$$

Algoritmo di Doolittle (Fattorizzazione compatta)

- La fattorizzazione LU equivale a trovare i coefficienti l_{ir} e u_{rj} che soddisfano

$$a_{ij} = \sum_{r=1}^{\min(i,j)} l_{ir}u_{rj}$$

- Supponiamo che le prime $k - 1$ colonne di L e U siano disponibili e $l_{kk} = 1$. Allora

$$\begin{cases} a_{kj} = \sum_{r=1}^{k-1} l_{kr}u_{rj} + u_{kj} & j = k, \dots, n \\ a_{ik} = \sum_{r=1}^{k-1} l_{ir}u_{rk} + l_{ik}u_{kk} & i = k + 1, \dots, n \end{cases}$$

Algoritmo di Doolittle (Fattorizzazione compatta)

- La fattorizzazione LU equivale a trovare i coefficienti l_{ir} e u_{rj} che soddisfano

$$a_{ij} = \sum_{r=1}^{\min(i,j)} l_{ir} u_{rj}$$

- Supponiamo che le prime $k-1$ colonne di L e U siano disponibili e $l_{kk} = 1$. Allora

$$\begin{cases} a_{kj} = \sum_{r=1}^{k-1} l_{kr} u_{rj} + u_{kj} & j = k, \dots, n \\ a_{ik} = \sum_{r=1}^{k-1} l_{ir} u_{rk} + l_{ik} u_{kk} & i = k+1, \dots, n \end{cases}$$

- Riordinando i termini, per $k = 1, \dots, n$

$$\begin{cases} u_{kj} = a_{kj} - \sum_{r=1}^{k-1} l_{kr} u_{rj} & j = k, \dots, n \\ l_{ik} = \frac{1}{u_{kk}} \left(a_{ik} - \sum_{r=1}^{k-1} l_{ir} u_{rk} \right) & i = k+1, \dots, n \end{cases}$$

Fattorizzazione di Cholesky

Sia $A \in \mathbb{R}^{n \times n}$ una matrice *simmetrica e definita positiva*. Allora esiste un'unica matrice triangolare superiore H con elementi sulla diagonale strettamente positivi tale che

$$A = H^T H$$

Gli elementi h_{ij} di H possono essere calcolati con il seguente algoritmo: $h_{11} = \sqrt{a_{11}}$ e, per $i = 2, \dots, n$

$$\begin{cases} h_{ij} = \frac{1}{h_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} h_{ik} h_{jk} \right) & j = 1, \dots, i-1 \\ h_{ii} = \left(a_{ii} - \sum_{k=1}^{i-1} h_{ik}^2 \right)^{1/2} \end{cases}$$

Raffinamento successivo della soluzione

- Supponiamo di avere una soluzione approssimata $\tilde{\mathbf{x}}$ del sistema $A\mathbf{x} = \mathbf{b}$

Raffinamento successivo della soluzione

- Supponiamo di avere una soluzione approssimata $\tilde{\mathbf{x}}$ del sistema $A\mathbf{x} = \mathbf{b}$
- Sia $\mathbf{z} = \mathbf{x} - \tilde{\mathbf{x}}$ la differenza tra la soluzione esatta (sconosciuta) e quella approssimata.

Raffinamento successivo della soluzione

- Supponiamo di avere una soluzione approssimata $\tilde{\mathbf{x}}$ del sistema $A\mathbf{x} = \mathbf{b}$
- Sia $\mathbf{z} = \mathbf{x} - \tilde{\mathbf{x}}$ la differenza tra la soluzione esatta (sconosciuta) e quella approssimata.
- Sia $\mathbf{r} = \mathbf{b} - A\tilde{\mathbf{x}}$ il *residuo*

Raffinamento successivo della soluzione

- Supponiamo di avere una soluzione approssimata $\tilde{\mathbf{x}}$ del sistema $A\mathbf{x} = \mathbf{b}$
- Sia $\mathbf{z} = \mathbf{x} - \tilde{\mathbf{x}}$ la differenza tra la soluzione esatta (sconosciuta) e quella approssimata.
- Sia $\mathbf{r} = \mathbf{b} - A\tilde{\mathbf{x}}$ il *residuo*
- Con le quantità appena introdotte abbiamo $A\mathbf{z} = \mathbf{r}$

Raffinamento successivo della soluzione

- Supponiamo di avere una soluzione approssimata $\tilde{\mathbf{x}}$ del sistema $A\mathbf{x} = \mathbf{b}$
- Sia $\mathbf{z} = \mathbf{x} - \tilde{\mathbf{x}}$ la differenza tra la soluzione esatta (sconosciuta) e quella approssimata.
- Sia $\mathbf{r} = \mathbf{b} - A\tilde{\mathbf{x}}$ il *residuo*
- Con le quantità appena introdotte abbiamo $A\mathbf{z} = \mathbf{r}$
- Si può definire dunque l'algoritmo

$$\begin{cases} \mathbf{r}^{(k)} = \mathbf{b} - A\tilde{\mathbf{x}}^{(k)} \\ A\mathbf{z}^{(k)} = \mathbf{r}^{(k)} \\ \tilde{\mathbf{x}}^{(k+1)} = \tilde{\mathbf{x}}^{(k)} + \mathbf{z}^{(k)} \end{cases}$$

Raffinamento successivo della soluzione

- Supponiamo di avere una soluzione approssimata $\tilde{\mathbf{x}}$ del sistema $A\mathbf{x} = \mathbf{b}$
- Sia $\mathbf{z} = \mathbf{x} - \tilde{\mathbf{x}}$ la differenza tra la soluzione esatta (sconosciuta) e quella approssimata.
- Sia $\mathbf{r} = \mathbf{b} - A\tilde{\mathbf{x}}$ il *residuo*
- Con le quantità appena introdotte abbiamo $A\mathbf{z} = \mathbf{r}$
- Si può definire dunque l'algoritmo

$$\begin{cases} \mathbf{r}^{(k)} = \mathbf{b} - A\tilde{\mathbf{x}}^{(k)} \\ A\mathbf{z}^{(k)} = \mathbf{r}^{(k)} \\ \tilde{\mathbf{x}}^{(k+1)} = \tilde{\mathbf{x}}^{(k)} + \mathbf{z}^{(k)} \end{cases}$$

- $\lim_{k \rightarrow \infty} \tilde{\mathbf{x}}^{(k)} = \mathbf{x}$

Raffinamento successivo della soluzione

- Supponiamo di avere una soluzione approssimata $\tilde{\mathbf{x}}$ del sistema $A\mathbf{x} = \mathbf{b}$
- Sia $\mathbf{z} = \mathbf{x} - \tilde{\mathbf{x}}$ la differenza tra la soluzione esatta (sconosciuta) e quella approssimata.
- Sia $\mathbf{r} = \mathbf{b} - A\tilde{\mathbf{x}}$ il *residuo*
- Con le quantità appena introdotte abbiamo $A\mathbf{z} = \mathbf{r}$
- Si può definire dunque l'algoritmo

$$\begin{cases} \mathbf{r}^{(k)} = \mathbf{b} - A\tilde{\mathbf{x}}^{(k)} \\ A\mathbf{z}^{(k)} = \mathbf{r}^{(k)} \\ \tilde{\mathbf{x}}^{(k+1)} = \tilde{\mathbf{x}}^{(k)} + \mathbf{z}^{(k)} \end{cases}$$

- $\lim_{k \rightarrow \infty} \tilde{\mathbf{x}}^{(k)} = \mathbf{x}$
- La fattorizzazione LU viene eseguita una volta sola

Matrici sparse

- Una matrice $A \in \mathbb{R}^{n \times n}$ è ritenuta *sparsa* se ha un numero di elementi non nulli dell'ordine di $O(n)$ (invece che $O(n^2)$)

Matrici sparse

- Una matrice $A \in \mathbb{R}^{n \times n}$ è ritenuta *sparsa* se ha un numero di elementi non nulli dell'ordine di $O(n)$ (invece che $O(n^2)$)
- Soluzione di EDP mediante differenze finite, elementi finiti e volumi finiti \Rightarrow matrici sparse di grandi dimensioni

Matrici sparse

- Una matrice $A \in \mathbb{R}^{n \times n}$ è ritenuta *sparsa* se ha un numero di elementi non nulli dell'ordine di $O(n)$ (invece che $O(n^2)$)
- Soluzione di EDP mediante differenze finite, elementi finiti e volumi finiti \Rightarrow matrici sparse di grandi dimensioni
- Sono stati definiti metodi diretti efficienti per matrici sparse con struttura particolare (matrice a banda, tridiagonale, a blocchi, ecc.)

Matrici sparse

- Una matrice $A \in \mathbb{R}^{n \times n}$ è ritenuta *sparsa* se ha un numero di elementi non nulli dell'ordine di $O(n)$ (invece che $O(n^2)$)
- Soluzione di EDP mediante differenze finite, elementi finiti e volumi finiti \Rightarrow matrici sparse di grandi dimensioni
- Sono stati definiti metodi diretti efficienti per matrici sparse con struttura particolare (matrice a banda, tridiagonale, a blocchi, ecc.)
- In generale, il processo di fattorizzazione non conserva la sparsità (fenomeno del *fill-in*) \Rightarrow alto costo in termini di memoria necessaria
 - Utilizzo di algoritmi di riordinamento (Cuthill-McKee, Reverse Cuthill-McKee, etc.)

Matrici sparse

- Una matrice $A \in \mathbb{R}^{n \times n}$ è ritenuta *sparsa* se ha un numero di elementi non nulli dell'ordine di $O(n)$ (invece che $O(n^2)$)
- Soluzione di EDP mediante differenze finite, elementi finiti e volumi finiti \Rightarrow matrici sparse di grandi dimensioni
- Sono stati definiti metodi diretti efficienti per matrici sparse con struttura particolare (matrice a banda, tridiagonale, a blocchi, ecc.)
- In generale, il processo di fattorizzazione non conserva la sparsità (fenomeno del *fill-in*) \Rightarrow alto costo in termini di memoria necessaria
 - Utilizzo di algoritmi di riordinamento (Cuthill-McKee, Reverse Cuthill-McKee, etc.)
- Per matrici sparse non strutturate di grande dimensione, i metodi iterativi sono spesso utilizzati come alternativa ai metodi diretti

Matrici tridiagonali: algoritmo di Thomas

$$A = \begin{pmatrix} a_1 & c_1 & & 0 \\ b_2 & a_2 & \ddots & \\ & \ddots & \ddots & c_{n-1} \\ 0 & & b_n & a_n \end{pmatrix}$$

Matrici tridiagonali: algoritmo di Thomas

$$A = \begin{pmatrix} a_1 & c_1 & & 0 \\ b_2 & a_2 & \ddots & \\ & \ddots & \ddots & c_{n-1} \\ 0 & & b_n & a_n \end{pmatrix}$$

$$L = \begin{pmatrix} 1 & & & 0 \\ \beta_2 & 1 & & \\ & \ddots & \ddots & \\ 0 & & \beta_n & 1 \end{pmatrix}$$

$$U = \begin{pmatrix} \alpha_1 & c_1 & & 0 \\ & \alpha_2 & \ddots & \\ & & \ddots & c_{n-1} \\ 0 & & & \alpha_n \end{pmatrix}$$

Dove $\alpha_1 = a_1$ e, per $i = 2, \dots, n$

$$\beta_i = \frac{b_i}{a_{i-1}} \quad \alpha_i = a_i - \beta_i c_{i-1}$$

Matrici tridiagonali: algoritmo di Thomas

$$A = \begin{pmatrix} a_1 & c_1 & & 0 \\ b_2 & a_2 & \ddots & \\ & \ddots & \ddots & c_{n-1} \\ 0 & & b_n & a_n \end{pmatrix}$$

$$L = \begin{pmatrix} 1 & & & 0 \\ \beta_2 & 1 & & \\ & \ddots & \ddots & \\ 0 & & \beta_n & 1 \end{pmatrix}$$

$$U = \begin{pmatrix} \alpha_1 & c_1 & & 0 \\ & \alpha_2 & \ddots & \\ & & \ddots & c_{n-1} \\ 0 & & & \alpha_n \end{pmatrix}$$

Dove $\alpha_1 = a_1$ e, per $i = 2, \dots, n$

$$\beta_i = \frac{b_i}{a_{i-1}} \quad \alpha_i = a_i - \beta_i c_{i-1}$$

Costo computazionale:

+/-	×	÷
$n-1$	$n-1$	$n-1$

 $\Rightarrow O(n)$

Algoritmo di Thomas per $A\mathbf{x} = \mathbf{f}$ (A tridiagonale)

Dobbiamo risolvere $L\mathbf{y} = \mathbf{f}$ e $U\mathbf{x} = \mathbf{y}$ con

$$L = \begin{pmatrix} 1 & & & 0 \\ \beta_2 & 1 & & \\ & \ddots & \ddots & \\ 0 & & \beta_n & 1 \end{pmatrix} \quad U = \begin{pmatrix} \alpha_1 & c_1 & & 0 \\ & \alpha_2 & \ddots & \\ & & \ddots & c_{n-1} \\ 0 & & & \alpha_n \end{pmatrix}$$

- 1 $L\mathbf{y} = \mathbf{f} \Rightarrow$ Sostituzione in avanti:

$$y_1 = f_1, \quad y_i = f_i - \beta_i y_{i-1} \quad i = 2, \dots, n$$

- 2 $U\mathbf{x} = \mathbf{y} \Rightarrow$ Sostituzione all'indietro:

$$y_n = \frac{y_n}{\alpha_n}, \quad x_i = (y_i - c_i x_{i+1}) / \alpha_i \quad i = n-1, \dots, 1$$

Algoritmo di Thomas per $A\mathbf{x} = \mathbf{f}$ (A tridiagonale)

Dobbiamo risolvere $L\mathbf{y} = \mathbf{f}$ e $U\mathbf{x} = \mathbf{y}$ con

$$L = \begin{pmatrix} 1 & & & 0 \\ \beta_2 & 1 & & \\ & \ddots & \ddots & \\ 0 & & \beta_n & 1 \end{pmatrix} \quad U = \begin{pmatrix} \alpha_1 & c_1 & & 0 \\ & \alpha_2 & \ddots & \\ & & \ddots & c_{n-1} \\ 0 & & & \alpha_n \end{pmatrix}$$

1 $L\mathbf{y} = \mathbf{f} \Rightarrow$ Sostituzione in avanti:

$$y_1 = f_1, \quad y_i = f_i - \beta_i y_{i-1} \quad i = 2, \dots, n$$

2 $U\mathbf{x} = \mathbf{y} \Rightarrow$ Sostituzione all'indietro:

$$y_n = \frac{y_n}{\alpha_n}, \quad x_i = (y_i - c_i x_{i+1}) / \alpha_i \quad i = n-1, \dots, 1$$

Costo computazionale:

+/-	\times	\div
$2(n-1)$	$2(n-1)$	n

 $\Rightarrow O(n)$