POLITECNICO DI TORINO

Corso di Laurea Magistrale
in Ingegneria Matematica

Tesi di Laurea Magistrale

Sparse Approximate Inverse per il precondizionamento di matrici

Relatore
Prof. Stefano Berrone
firma del relatore

. . . . . .

Candidato
Edoardo Failla
firma del candidato

. . . . . .

Anno Accademico 2024-2025

Sommario

Una caratteristica fondamentale di ogni sistema lineare è il suo numero di condizionamento, che è definito mediante la matrice del sistema. Tale quantità è strettamente connessa alla stabilità del processo risolutivo e all'accuratezza della soluzione ottenuta. Proprio per questo motivo si introducono i precondizionatori, costruiti per ottimizzare il condizionamento di un sistema lineare. I precondizionatori del tipo Sparse Approximate Inverse costituiscono una nuova categoria di precondizionatori, nati dall'idea di minimizzare, in norma di Frobenius, la differenza fra la matrice del sistema lineare A e la matrice identità, attraverso una matrice M sparsa che approssima l'inversa di A. Da questa idea si sviluppano gli algoritmi SPAI, ideato per matrici generiche, e FSAI, ideato per matrici simmetriche definite positive. L'input principale di questi algoritmi è lo sparsity pattern di M, e partendo da uno sparsity pattern predefinito, si risolve il problema di minimizzazione per ottenere M. In questo studio si presenteranno i due algoritmi citati e alcune procedure comunemente adottate per ottenere un'euristica sullo sparsity pattern in input. Si osserverà inoltre, dall'esposizione degli algoritmi stessi, che grazie alle proprietà della norma di Frobenius il problema di minimizzazione è suddivisibile in sottoproblemi indipendenti, rivelando quindi la natura intrinsecamente parallela degli algoritmi SPAI e FSAI, ed evidenziando dunque il loro potenziale uso in sistemi di calcolo paralleli.

Indice

Parte I

Sparse Approximate Inverse

Capitolo 1

Condizionamento dei sistemi lineari

1.1 Introduzione

Il problema a cui siamo interessati è la risoluzione di un sistema lineare del tipo $Ax = b$, con A matrice non singolare e b vettore dei termini noti. A seconda della struttura della matrice A, la soluzione del sistema lineare non è banale, e richiede, nel caso in cui la dimensione del problema sia elevata, degli approcci diversificati. Gli algoritmi di risoluzione per sistemi lineari ricadono principalmente in due categorie: i metodi *diretti* e i metodi *iterativi*. I metodi diretti genericamente sono costruiti sulla base di una fattorizzazione di matrice (e.g. decomposizione QR), e sono generalmente robusti, ma hanno una complessità computazionale elevata e richiedono un gran quantitativo di memoria. I metodi iterativi invece costruiscono iterazione per iterazione una approssimazione della soluzione $x^{(n)}$, cercando di convergere alla soluzione reale x. Uno dei fattori determinanti per la convergenza del metodo iterativo è il condizionamento del sistema lineare, espresso anche come numero di condizionamento della matrice A. In genere quindi si effettua quello che è comunemente chiamato *precondizionamento*. Il termine precondizionamento si riferisce a una tecnica attraverso la quale si trasforma il sistema lineare originario in un altro sistema lineare avente medesima soluzione, ma di più facile risoluzione. L'obiettivo è trovare e utilizzare una matrice M non singolare, trasformando il sistema

$$Ax = b \quad \Rightarrow \quad MAx = Mb$$

dove M rappresenta un'approssimazione dell'inversa A^{-1}. Il precondizionatore può anche essere applicato da destra, oppure sotto forma di fattorizzazione $M = M_1 M_2$.

In questo capitolo riassumeremo le formule per i sistemi precondizionati e il concetto di sistema mal condizionato.

1.2 Condizionamento e matrici

1.2.1 Nozioni preliminari

Introduciamo alcune notazioni e definizioni fondamentali.

Definizione 1.2.1. *Sia V uno spazio vettoriale su K. L'applicazione $\|\cdot\| : V \to \mathbb{R}$ è una* ***norma*** *su V se sono soddisfatte le seguenti proprietà:*

1. $\|x\| \geq 0, \ \forall x \in V, \ \ e \ \ \|x\| = 0 \iff x = 0.$
2. $\|\alpha x\| = |\alpha| \|x\| \quad \forall \alpha \in K, \forall x \in V.$
3. $\|x + y\| \leq \|x\| + \|y\| \quad \forall x, y \in V.$

essendo $|\alpha|$ il valore assoluto di α se $K = \mathbb{R}$. □

Esempio 1.2.1. *Consideriamo l'insieme $\mathbb{R}^n$. Chiamiamo* ***norma p*** *o norma di Hölder, la norma:*

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

∘

Passiamo adesso a definire la nozione di norma per una matrice generica.

Definizione 1.2.2. *Si definisce* ***norma matriciale*** *un'applicazione $\|\cdot\| : \mathbb{C}^{m \times n} \to \mathbb{R}$ tale che siano soddisfatte le seguenti proprietà:*

1. $\|A\| \geq 0, \ \forall A \in \mathbb{C}^{m \times n}, \ \ e \ \ \|A\| = 0 \iff A = 0.$
2. $\|\alpha A\| = |\alpha| \|A\| \quad \forall \alpha \in \mathbb{C}, \forall A \in \mathbb{C}^{m \times n}.$
3. $\|A + B\| \leq \|A\| + \|B\| \quad \forall A, B \in \mathbb{C}^{m \times n}.$ □

Definizione 1.2.3. *Si definisce una* ***norma matriciale*** *come* ***compatibile*** *con una norma vettoriale $\|\cdot\|$ se*

$$\|Ax\| \leq \|A\| \|x\|, \quad \forall x \in \mathbb{C}^n \tag{1.1}$$

□

Esempio 1.2.2. *Sia $A \in \mathbb{C}^{m\times n}$. Chiamiamo* ***norma di Frobenius*** *la norma:*

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

Si può anche notare che questa norma sia compatibile con la norma vettoriale euclidea $\|A\|_2$. Si ha che:

$$\|Ax\|_2^2 = \sum_{i=1}^{m}|\sum_{j=1}^{n}a_{ij}x_j|^2 \leq \sum_{i=1}^{m}\left(\sum_{j=1}^{n}|a_{ij}|\sum_{j=1}^{n}|x_j|^2\right) = \|A\|_F^2\|x\|_2^2$$

∘

Per una generica matrice $A \in \mathbb{C}^{m\times n}$, si definisce inoltre la norma

$$\|A\|_p = \sup_{x\in\mathbb{C}^n, x\neq 0}\frac{\|Ax\|_p}{\|x\|_p}$$

Tale norma viene definita norma indotta dalla norma $\|\cdot\|_p$, e soddisfa le proprietà di norma 1.2.2. Altri esempi sono:

$$\|A\|_1 = \max_{j=1,\dots,n}\sum_{i=1}^{m}|a_{ij}|, \tag{1.2}$$

$$\|A\|_\infty = \max_{i=1,\dots,m}\sum_{j=1}^{n}|a_{ij}| \tag{1.3}$$

Introducendo invece la quantità chiamata *raggio spettrale* come:

$$\rho(A) = \max_{\lambda\in\Lambda(A)}|\lambda|$$

con $\Lambda(A)$ insieme degli autovalori di A. Ovvero come il massimo dei moduli degli autovalori di A, si ricava allora per la *norma 2* che:

Teorema 1.2.1. *Definito $\sigma_1(A)$ come il più grande valore singolare di $A \in \mathbb{C}^{m\times n}$, allora si ha che:*

$$\|A\|_2 = \sqrt{\rho(A^HA)} = \sqrt{\rho(AA^H)} = \sigma_1(A) \tag{1.4}$$

□

Inoltre per quanto riguarda il raggio spettrale si può dimostrare che:

Teorema 1.2.2. *Consideriamo $\|\cdot\|$ norma matriciale compatibile con una norma*

vettoriale, allora si ha che:

$$\rho(A) \leq \|A\| \qquad \forall A \in \mathbb{C}^{n\times n}. \tag{1.5}$$

Dimostrazione. λ autovalore generico di A e $w \neq 0$ autovettore di λ associato:

$$|\lambda|\|w\| = \|\lambda w\| = \|Aw\| \leq \|A\|\|w\| \implies |\lambda| \leq \|A\|$$

□

Le differenti norme sono messe in relazione tra loro attraverso una costante dipendente dalla dimensione n, come nel caso di matrici quadrate $A \in \mathbb{R}^{n\times n}$:

Esempio 1.2.3.

$$\|A\|_1 \leq n\|A\|_\infty, \qquad \|A\|_\infty \leq n\|A\|_1, \tag{1.6}$$
$$\|A\|_2 \leq \sqrt{n}\|A\|_\infty, \qquad \|A\|_\infty \leq \sqrt{n}\|A\|_2, \tag{1.7}$$
$$\|A\|_2 \leq \sqrt{\|A\|_1\|A\|_\infty} \tag{1.8}$$

∘

1.2.2 Condizionamento e stabilità

Supponiamo ora di avere il generico sistema lineare:

$$Ax = b \tag{1.9}$$

con

- $A \in \mathbb{R}^{n\times n}$, *matrice dei coefficienti non singolare.*
- $x \in \mathbb{R}^n$, *vettore delle incognite.*
- $b \in \mathbb{R}^n$, *vettore dei termini noti.*

La soluzione del sistema, data la non singolarità di A, risulta $x = A^{-1}b$. Ipotizziamo allora una una pertubazione δb soltanto sul termine noto (i.e. $\delta A = 0$), e definiamo il risultato esatto del sistema perturbato $x + \delta x$. Il sistema diventa dunque:

$$A(x + \delta x) = b + \delta b$$

Da cui si ricava

$$A\delta x = \delta b, \qquad \delta x = A^{-1}\delta b$$

Data allora una qualsiasi norma matriciale indotta, si avrà che:

$$\begin{cases} \|\delta x\| = \|A^{-1}\delta b\| \leq \|A^{-1}\|\|\delta b\| \\ \|b\| = \|Ax\| \leq \|A\|\|x\| \end{cases} \implies \frac{\|\delta x\|}{\|x\|} \leq \|A\|\|A^{-1}\|\frac{\|\delta b\|}{\|b\|} \tag{1.10}$$

Si nota allora il bound che otteniamo mediante la pertubazione sul terminte noto della soluzione esatta $x + \delta x$. La costante moltiplicativa $\|A\|\|A^{-1}\|$ assume quindi un significato centrale nello studio dei sistemi lineari, e identifica proprio la stabilità del sistema lineare rispetto alla presenza di errori, come errori di arrotondamento o troncamento. Si procede allora dando una definizione precisa del termine in esame

Definizione 1.2.4. *Si definisce* ***numero di condizionamento*** *della matrice A la quantità:*

$$\kappa(A) = \|A\|\|A^{-1}\| \tag{1.11}$$

con $\|\cdot\|$ norma di matrice indotta. □

In generale $\kappa(A)$ dipende dalla norma scelta ed è sempre un numero maggiore o uguale a 1:

$$1 = \|I\| = \|AA^{-1}\| \leq \|A\|\|A^{-1}\| = \kappa(A)$$

Un caso particolare si ha per $p = 2$, dove $\kappa_2(A)$ viene definito come:

$$\kappa_2(A) = \|A\|_2\|A^{-1}\|_2 = \frac{\sigma_1(A)}{\sigma_n(A)} \tag{1.12}$$

dove $\sigma_1(A), \sigma_n(A)$ rappresentano il massimo e il minimo valore singolare di A. Se inoltre la matrice A è anche simmetrica si ricava allora che:

$$\kappa_2(A) = \|A\|_2\|A^{-1}\|_2 = \rho(A)\rho(A^{-1}) = \frac{|\lambda_{max}|}{|\lambda_{min}|} \tag{1.13}$$

Inoltre il significato di $\kappa(A)$ può essere maggiormente riconosciuto attraverso le seguenti formule tra di loro equivalenti [12]:

$$\kappa(A) = \lim_{\varepsilon \to 0} \sup_{\|\delta A\| \leq \varepsilon \|A\|} \frac{\|(A + \delta A)^{-1} - A^{-1}\|}{\varepsilon \|A^{-1}\|} \tag{1.14}$$

$$\kappa(A) = \frac{1}{dist(A)} \quad con \quad dist(A) = \min_{A + \delta A \ singolare} \frac{\|\delta A\|}{\|A\|} \tag{1.15}$$

le quali indicano che più il numero di condizionamento è basso, più la matrice è lontana da essere una matrice singolare. Nella rappresentazione floating-point, la rappresentazione numerica avviene in aritmetica finita, ed è realistico aspettarsi un errore $\|\delta A\| \leq \zeta \|A\|$ con ζ errore di *roundoff*, con la conseguenza che un numero di condizionamento inferiore si traduce in una maggiore stabilità nella risoluzione del

sistema. Considerando quindi la formulazione generica del problema perturbato (i.e. $\delta A \neq 0$) la 1.10 diventa quindi

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{\kappa(A)}{1 - \kappa(A)\|\delta A\|/\|A\|}(\|\delta A\|/\|A\| + \|\delta b\|/\|b\|) \tag{1.16}$$

Ritornando al sistema lineare iniziale, supponiamo di avere una soluzione approssimata $\bar{x}$ del sistema $Ax = b$. Definiamo allora il vettore dei residui:

$$\bar{r} = b - A\bar{x} \implies x - \bar{x} = A^{-1}\bar{r}$$

Si ricava allora l'espressione che lega la norma dell'errore relativo:

$$\epsilon(\bar{x}) \equiv \frac{\|x - \bar{x}\|}{\|x\|} = \frac{\|A^{-1}\bar{r}\|}{\|x\|} \leq \frac{\|A^{-1}\|\|\bar{r}\|}{\|x\|} \leq \frac{\|A^{-1}\|\|A\|\|\bar{r}\|}{\|b\|} = \kappa(A)\frac{\|\bar{r}\|}{\|b\|} \tag{1.17}$$

e la diseguaglianza inversa:

$$\frac{1}{\kappa(A)}\frac{\|\bar{r}\|}{\|b\|} = \frac{\|b - A\bar{x}\|}{\|A^{-1}\|\|A\|\|b\|} \leq \frac{\|A(A^{-1}b - \bar{x})\|}{\|A\|\|x\|} \leq \frac{\|x - \bar{x}\|}{\|x\|} = \epsilon(\bar{x}) \tag{1.18}$$

e in modo più compatto

$$\frac{1}{\kappa(A)}\frac{\|\bar{r}\|}{\|b\|} \leq \epsilon(\bar{x}) \leq \kappa(A)\frac{\|\bar{r}\|}{\|b\|} \tag{1.19}$$

Queste disuguaglianze legano l'errore relativo sulla soluzione con la norma relativa sul residuo, e permettono di giungere a diverse conclusioni:

- Per valori molto elevati di $\kappa(A)$, per un errore relativo sul residuo molto piccolo si può comunque ottenere un errore relativo sulla soluzione considerevole.
- In generale quindi, un residuo piccolo non implica necessariamente un errore sulla soluzione altrettanto contenuto.
- Il bound sull'errore relativo in 1.19 cresce con l'aumentare del numero di condizionamento. Ciò significa che maggiore è il numero condizionamento, minore potrebbe essere l'accuratezza della soluzione del sistema lineare.

Il numero di condizionamento $\kappa(A)$ quindi influenza la procedura di risoluzione di un sistema lineare anche considerando la diseguaglianza rispetto all'errore relativo data dalla 1.19. Un basso numero di condizionamento di conseguenza ci permette di associare un residuo relativo piccolo a un errore relativo contenuto.

1.3 Metodi iterativi e precondizionamento

Come abbiamo precedentemente sostenuto, il vantaggio dei metodi iterativi risiede nel minor costo computazionale e nel poter risolvere il sistema utilizzando soltanto prodotti di tipo matrice vettore. Lo svantaggio invece risiede nel fatto che essi dipendono fortemente dalle proprietà spettrali della matrice A e dal numero di condizionamento. L'idea alla base dei metodi iterativi è quella di approssimare la soluzione reale x attraverso delle soluzioni approssimate $x^{(k)}$, ottenendo una successione che possibilmente arrivi a convergenza.

$$\lim_{k\to\infty} x^k = x, \quad con \quad x = A^{-1}b$$

Nella realtà però quello che si fa è fissare dei criteri d'arresto affinché il calcolo si fermi all'iterazione k-esima massima prestabilita, o al raggiungimento di una tolleranza sull'errore relativo, come in 1.19. La classe dei metodi iterativi si distingue all'interno per la presenza di diverse categorie in base alle loro proprietà. Per dare un esempio, considerando il sistema avente forma

$$\begin{aligned} x^{(0)} &= f_0(A, b) \\ x^{(k+1)} &= f_{k+1}(x^{(k)}, x^{(k-1)}, \dots, A, b) \end{aligned}$$

esso si chiamerà *stazionario* se le funzioni f_j sono indipendenti dalla j-esima iterazione, viceversa *non stazionario*. Inoltre si chiamerà *lineare* se ogni f dipende linearmente dalle approssimazioni precedenti $(x^{(k)}, x^{(k-1)}, ..., x^{(k-m)})$, con $k - m$ *ordine* del metodo.

1.3.1 Metodi iterativi classici

L'obiettivo dei metodi iterativi classici è verificare se sia possibile usare una funzione $g : \mathbb{R}^n \to \mathbb{R}^n$ per costruire una successione di soluzioni approssimate $x^{(k)}$ attraverso iterazioni di punto fisso lineari.

$$g(x^{(k)}) = x^{(k+1)} \in \mathbb{R}^n,\ k \geq 0 \quad con \quad x^{(k+1)} = Gx^{(k)} + b \quad con \quad G := I - A$$

Affinché il metodo sia consistente con il sistema lineare, deve essere verificata la condizione $x = Gx + b$. Le condizioni quindi dovranno essere del tipo

$$\lim_{k\to\infty} x^{(k)} = x = A^{-1}b \qquad con \quad x = Gx + b$$

Si ha allora che

Teorema 1.3.1 (**Convergenza di metodi iterativi**). *Per un metodo iterativo di punto fisso lineare e consistente, si ha che dato $x^{(0)}$ approssimazione iniziale:*

$$\lim_{k\to\infty} x^{(k)} = x \qquad \Longleftrightarrow \qquad \rho(G) < 1 \tag{1.20}$$

Dimostrazione. Data la consistenza del metodo si ricava che

$$x^{(k+1)} = Gx^{(k)} + b \implies x^{(k+1)} - x = Gx^{(k)} + b - x \quad con \quad x = Gx + b$$

$$\implies x^{(k+1)} - x = Gx^{(k)} + b - Gx - b \implies x^{(k+1)} - x = G(x^{(k)} - x)$$

Indicando con $e^{(j)} = x^{(j)} - x$, si ricava che

$$e^{(k+1)} = Ge^{(k)} \implies e^{(k+1)} = G^k e^{(0)}$$

sapendo che per una generica matrice B quadrata $\lim_{k\to\infty} B^k = 0 \iff \rho(B) < 1$, si ha che

$$\lim_{k\to\infty} G^k e^{(0)} = 0 \quad \forall e^{(0)} \qquad \Longleftrightarrow \qquad \rho(G) < 1$$

□

Metodi stazionari

Per migliorare l'iterazione del punto fisso, i principali metodi iterativi classici utilizzano una decomposizione additiva di $A = M - N$, ottenendo

$$x^{(k+1)} = M^{-1}Nx^{(k)} + M^{-1}b \quad equivalente\ a \quad M^{-1}Ax = M^{-1}b \tag{1.21}$$

con M non singolare, chiamata matrice di precondizionamento. Il sistema precondizionato così avrà un numero di condizionamento minore e come precedentemente dimostrato, la convergenza si ha per $\rho(M^{-1}N) = \rho(I - M^{-1}A) < 1$. Purtroppo, i metodi stazionari possono mostrare una convergenza lenta che risulta insufficiente a risolvere il problema. Per superare questo inconveniente, l'accelerazione può essere generalmente ottenuta utilizzando un parametro di rilassamento α. Considerando $\alpha = 1$ la 1.21 può essere riscritta come

$$x^{(k+1)} = x^{(k)} + \alpha M^{-1}r^{(k)}, \quad k \geq 0$$

la quale rappresenta una generica forma del metodo di *Richardson* stazionario. Ovviamente α, è il parametro di rilassamento il quale valore può essere diverso da 1, proprio poiché rappresenta un fattore di accelerazione. Nel caso in cui α dipende dal passo d'iterazione *k-esimo*, si ha

$$x^{(k+1)} = x^{(k)} + \alpha_k M^{-1}r^{(k)}, \quad con \quad r^{(k)} = b - Ax^{(k)}, \qquad k \geq 0$$

ottenendo una forma del metodo di *Richardson* non stazionario.

1.3.2 Metodi non stazionari: proiezione su sottospazi di Krylov

Uno dei metodi moderni per la soluzione di sistemi lineari, consiste nell'idea di ricercare una soluzione di $Ax = b$, che appartenga ad un determinato spazio $\mathcal{K}$. Sia $A \in \mathbb{R}^{n\times n}, v \in \mathbb{R}^n$, definiamo lo *spazio di Krylov*

$$\mathcal{K}_m(A, v) = span\{v, Av, A^2v, \dots, A^{m-1}v\} \subseteq \mathbb{R}^n \quad m = 1,2,3,\dots$$

I metodi di proiezione su sottospazi di Krylov ricercano per ogni m una soluzione approssimata del sistema lineare che appartenga a $\mathcal{K}_m(A, r^{(0)})$, dove $r^{(0)}$ è il residuo iniziale $r^{(0)} = b - Ax^{(0)}$. Questi metodi generalmente determinano la soluzione approssimata $x^{(m)}$ imponendo m vincoli, e tipicamente vengono imposti m condizioni indipendenti di ortogonalità sul vettore residuo, definite in un altro spazio chiamato $\mathcal{L}_m \subseteq \mathbb{R}^n$. Avremo quindi che

$$x^{(k)} \in x^{(0)} + \mathcal{K}_k \qquad con \qquad (b - Ax^{(k)}) \perp \mathcal{L}_k \tag{1.22}$$

dove la dimensione di entrambi gli spazi $\mathcal{K}_k, \mathcal{L}_k$ sarà $m \leq n$.

Il metodo del gradiente e gradiente coniugato

Per il caso particolare di matrici simmetriche definite positive (SPD), il problema si può riformulare come la ricerca del minimo di un funzionale quadratico ϕ definito come segue

$$\phi(x) \coloneqq \frac{1}{2}x^T Ax - b^T x \tag{1.23}$$

Infatti, considerando la soluzione esatta $x = A^{-1}b$, si ha che essendo ϕ un funzionale convesso la soluzione coincide nel trovare il punto in cui si annulla il gradiente

$$x = \arg\min_{\hat{x}\in\mathbb{R}^n} \phi(\hat{x}) \iff \nabla\phi(x) = \frac{1}{2}(A + A^T)x - b = Ax - b = 0 \tag{1.24}$$

Si può quindi procedere con un metodo iterativo non stazionario facendo si che

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

che partendo da un'approssimazione iniziale $x^{(0)}$, minimizza il funzionale per una data direzione d, che varia ad ogni passo. In un primo approccio si potrebbe

pensare di scegliere la direzione di massima discesa (*steepest descent*), ricavando

$$d^{(k)} = -\nabla\phi(x^{(k)}) = r^{(k)}$$

e come valore di α

$$\min_{\alpha_k} \phi(x^{(k)} + \alpha_k d^{(k)}) \iff \frac{d}{d\alpha_k}\phi(x^{(k)} + \alpha_k d^{(k)}) = 0 \iff \alpha_k = \frac{\langle r^{(k)}, r^{(k)}\rangle}{\langle r^{(k)}, Ar^{(k)}\rangle}.$$

dove $\langle\cdot,\cdot\rangle$ indica il prodotto scalare.
Il metodo può manifestare una scarsa convergenza a causa della potenziale dipendenza lineare tra le direzioni di ricerca. In termini generali, non è garantito che converga entro un massimo di n iterazioni in aritmetica finita. Si osserva, inoltre, che tale inefficienza è particolarmente evidente nel caso in cui la matrice A sia malcondizionata. Per ovviare a queste problematiche si procede quindi con una diversa selezione delle direzioni. La relazione ricorsiva per $x^{(k)}$ rimane uguale, ma le direzioni assumono una diversa forma poiché

$$\begin{aligned} d^{(0)} &= r^{(0)} \\ d^{(k+1)} &= r^{(k+1)} + \beta_k d^{(k)}, \quad k \geq 0 \end{aligned}$$

Differentemente dal metodo *steepest descent*, il nuovo metodo chiamato metodo del *gradiente coniugato* impone che la direzione $d^{(k+1)}$ sia *A-ortogonale* con le precedenti direzioni $\langle d^{(j)}, Ad^{(k)}\rangle = 0$ *per* $j = 0,1,\dots,k-1$, ottenendo quindi

$$\beta_k = -\frac{\langle r^{(k+1)}, Ad^{(k)}\rangle}{\langle d^{(k)}, Ad^{(k)}\rangle}, \qquad \alpha = \frac{\langle r^{(k)}, d^{(k)}\rangle}{\langle d^{(k)}, Ad^{(k)}\rangle}$$

Quello che otterremo sarà allora che

$$x^{(k+1)} \in x^{(0)} + span\{d^{(0)}, d^{(1)}, \dots, d^{(k)}\},$$

e data la formulazione della soluzione in 1.22 mediante sottospazi di Krylov, abbiamo che

$$span\{d^{(0)}, d^{(1)}, \dots, d^{(k)}\} = \mathcal{K}_{k+1}(A, r^{(0)}), \quad k = 0,1,2,\dots \tag{1.25}$$

inoltre vi è associata al metodo *la condizione di Galerkin*

$$\mathcal{K}_k \perp r^{(k)} \in \mathcal{K}_{k+1}$$

la quale implica che lo spazio dove vengono definiti i vincoli è $\mathcal{L}_k = \mathcal{K}_k$, e che i residui $\{r^{(n)}\}_{n=0}^{k-1}$ sono ortogonali fra loro e formano una base di $\mathcal{K}_k$. Inoltre si

assume anche che le direzioni ricercate $\{d^{(n)}\}_{n=0}^{k-1}$ formato una base coniugata di $\mathcal{K}_k$. Il metodo del gradiente coniugato trova la soluzione, in aritmetica esatta, in al più n passi. Tuttavia, nella realtà applicativa si definisce nell'implementazione una tolleranza, che consente all'algoritmo di interrompere il calcolo anticipatamente, al raggiungimento di un criterio di arresto predefinito.

Convergenza per il gradiente coniugato

Definita la seguente norma vettoriale

Definizione 1.3.1. *Sia A simmetrica definita positiva, si indica con $\|\cdot\|_A$ la norma*

$$\|x\|_A = \sqrt{\langle x, Ax \rangle} \tag{1.26}$$

*detta **norma dell'energia** del vettore x* □

si procede quindi ad enunciare il teorema che esprime un limite sull'errore $e^{(k)}$, attraverso il numero di condizionamento

Teorema 1.3.2. ***(Errore di approssimazione del CG)*** *Per una matrice $A \in \mathbb{R}^{nxn}$ simmetrica definita positiva, $x \in \mathbb{R}^n$ soluzione del sistema lineare $Ax = b$, l'errore di approssimazione $e^{(k)} = x^{(k)} - x$, corrispondente al passo k-esimo del metodo del gradiente coniugato, rispetta la diseguaglianza*

$$\|x^{(k)} - x\|_A \leq 2 \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^k \|x^{(0)} - x\|_A \tag{1.27}$$

□

La velocità di convergenza dipende fortemente dal numero di condizionamento, e in questo caso dall'autovalore massimo e minimo di A. L'idea quindi di utilizzare un precondizionatore consiste proprio nel cercare una matrice $M \approx A^{-1}$ non singolare tale che $\kappa(AM) \ll \kappa(A)$, in modo da risolvere un sistema più stabile e che arrivi a convergenza.

CG per il caso non simmetrico

Grazie alla popolarità e al successo del metodo del gradiente coniugato per le matrici SPD, l'utilizzo di questo metodo iterativo è stato ampliato per matrici non SPD. Una estensione del metodo del gradiente coniugato è applicare tale metodo al sistema simmetrico definito positivo

$$A^T Ax = A^T b \tag{1.28}$$

applicando quindi il gradiente coniugato alle equazioni normali. In realtà quando si applica il metodo alla 1.28, non è necessario costruire esplicitamente la matrice A^TA, poiché bastano i vettori ausiliari $z^{(j)} = Ad^{(j)}$ e $q^{(j)} = A^Tr^{(j)}$, notando che

$$\langle A^TAd^{(j)}, d^{(j)}\rangle = \langle Ad^{(j)}, Ad^{(j)}\rangle = \langle z^{(j)}, z^{(j)}\rangle$$

L'algoritmo quindi calcola il residuo $q^{(j)} = A^T(b - Ax^{(j)})$, e non calcola $r^{(j)}$, che viene direttamente calcolato nel caso in cui fosse richiesto. Le proiezioni quindi sono ottenute sul sottospazio di Krylov $\mathcal{K}_m(A^TA, q_0) = \mathcal{K}_m(A^TA, A^Tr_0)$. Quello che si ottiene è che i vettori $q^{(j)}$ sono ortogonali fra loro come anche i vettori $w^{(j)}$, e inoltre i vettori $d^{(j)}$ sono tra loro A^TA-*coniugati*. La diseguaglianza 1.27 in questa riformulazione diventa

$$\|x^{(k)} - x\|_{A^TA} \leq 2\left(\frac{\sqrt{\kappa(A^TA)} - 1}{\sqrt{\kappa(A^TA)} + 1}\right)^k \|x^{(0)} - x\|_{A^TA} \tag{1.29}$$

dove in questo caso $\kappa(A^TA) = \sigma_1^2/\sigma_n^2$, dove σ_1 e σ_n sono rispettivamente il valore singolare massimo e minimo di A. Da ciò deriva che se la matrice A è malcondizionata, il sistema delle equazioni normali sarà notevolmente più malcondizionato, poiché $\kappa(A^TA)$ è il quadrato di $\kappa(A)$.

Altri metodi di Krylov

Dal momento della pubblicazione del metodo del gradiente coniugato, nei decenni successivi sono stati introdotti una varietà di altri metodi di Krylov fino ad oggi. Due dei più popolari fra gli altri metodi di Krylov sono il **GMRES** e il **BiCGSTAB**. Il metodo GMRES (Generalized Minimum Residual) è una tecnica iterativa per risolvere sistemi lineari $Ax = b$, particolarmente adatta a matrici generiche, non necessariamente simmetriche o definite positive. Esso costruisce una base ortogonale completa dello spazio di Krylov, minimizzando il residuo al termine di ogni iterazione. Sebbene il GMRES fornisca una convergenza rapida per problemi ben condizionati, la sua principale limitazione risiede nell'aumento dei requisiti di memoria, poiché la base ortogonale cresce con il numero di iterazioni. Per affrontare questo problema, si utilizza spesso il GMRES con restart, che limita le iterazioni memorizzate e riavvia il metodo utilizzando la soluzione corrente come nuovo punto di partenza. Il BiCGSTAB invece deriva dal **BiCG** (Bi-Conjugate Gradient), ed è progettato per risolvere sistemi lineari non simmetrici, aumentando la stabilizzazione del metodo BiCG. La biconiugazione utilizzata nel metodo BiCG implica due sequenze di aggiornamenti dei residui e delle direzioni di ricerca, una per la matrice A e una per la matrice trasposta A^T, corrispondenti a $\mathcal{K}^{(k)}$ e $\mathcal{K}_T^{(k)}$ rispettivamente. Sebbene non richieda una base ortogonale completa come il

GMRES, il BiCGSTAB può risultare più efficiente in termini di memoria, ma pur essendo efficace, può talvolta manifestare convergenza lenta o inefficiente.

1.3.3 Precondizionamento e matrici sparse

Come abbiamo precedentemente analizzato, la convergenza di un metodo iterativo dipende fortemente dal numero di condizionamento della matrice A. Per risolvere quindi il sistema lineare si effettua quindi un *precondizionamento* del sistema lineare, ottenendo un sistema con un numero di condizionamento inferiore. Esistono principalmente tre tipi di precondizionamento, riassunti nella tabella 1.1.
Le caratteristiche fondamentali di un precondizionatore devono riassumersi in:

- **Non singolare:** è ovviamente la prima caratteristica richiesta per la non singolarità del sistema precondizionato.
- **Bassa complessità computazionale:** la matrice di precondizionamento non deve essere computazionalmente onerosa da calcolare.
- **Preservare le proprietà della matrice:** in particolare se A è simmetrica definita positiva (SPD), si deve garantire che la matrice precondizionata AM mantenga tali proprietà, permettendo così l'applicabilità del metodo del gradiente coniugato (CG) anche con l'uso del precondizionatore, dando luogo al metodo del gradiente coniugato precondizionato (PCG).

Tabella 1.1: Classificazione e forme di precondizionamento.

Precondizionatore	**Diretto** $(M \approx A)$	**Inverso** $(M \approx A^{-1})$
da Sinistra	$M^{-1}Ax = M^{-1}b$	$MAx = Mb$
da Destra	$AM^{-1}y = b,\ x = M^{-1}y$	$AMy = b,\ x = My$
Fattorizzato	$M_1^{-1}AM_2^{-1}y = M_1^{-1}b,\ x = M_2^{-1}y$	$M_1AM_2y = M_1b,\ x = M_2y$

L'idea quindi del precondizionatore di una matrice non singolare A può quindi essere tradotta nel cercare di ottenere una matrice non singolare $M \approx A^{-1}$, con basso costo computazionale. Tuttavia, ottenere un'approssimazione dell'inversa di una matrice non è un'operazione banale. Un caso particolare di interesse riguarda le *matrici sparse*, ovvero matrici con una bassa densità di elementi diversi da zero, per le quali le tecniche di precondizionamento devono essere progettate in modo da sfruttare la struttura sparsa, evitando operazioni computazionalmente costose. Introduciamo per completezza la definizione di *sparsity pattern*.

Definizione 1.3.2. *Si definisce* ***sparsity pattern*** *di* $A \in \mathbb{R}^{m\times n}$*, indicato con* $SP\{A\}$*, l'insieme degli indici degli elementi diversi da zero:*

$$SP\{A\} = \{(i,j) :\ a_{ij} = (A)_{ij} \neq 0,\ \ \forall i,j\ \ 1 \leq i \leq m,\ 1 \leq j \leq n\} \tag{1.30}$$

□

Il numero di elementi diversi da zero può essere espresso come $|SP\{A\}|$ o con $nz(A)$. Per quanto riguarda la sparsità di A, non esiste una vera e propria definizione di matrice sparsa. L'idea generale è che la densità di elementi diversi da zero sia piccola rispetto alla dimensione $n \times n$ della matrice. Viceversa, se il numero di elementi diversi da zero è significativa rispetto alla dimensione della matrice, essa si chiamerà matrice *densa*. Una particolarità delle matrici sparse consiste nella struttura della loro inversa, poiché anche se la matrice A è sparsa, non è detto che lo sia anche la sua inversa A^{-1}. Per dare una idea di questo fenomeno, mostriamo un esempio utilizzando una serie di matrici A_n.

Esempio 1.3.1.

Sia $\{A_n\}_{n\in\mathbb{N}}$ una serie di matrici $n \times n$, la cui frazione di $nz(A)$ sul totale converge a 0. Consideriamo le A_n definite in questo modo

$$A_n = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

con 1 nella diagonale e negli elementi sopra la diagonale. Il numero di $nz(A)$ è $n + (n-1) = 2n - 1$, e conseguentemente all'aumentare della dimensione n, si ha che la frazione

$$\lim_{n\to\infty} \frac{nz(A)}{n^2} = \lim_{n\to\infty} \frac{2n-1}{n^2} = 0$$

Viceversa l'inversa di A_n^{-1} sarà formata dagli elementi b_{ij}

$$b_{ij} = \begin{cases} 0 & se \quad i > j, \\ 1 & se \quad i \leq j \quad e \quad j - i \; pari, \\ -1 & se \quad i \leq j \quad e \quad j - i \; dispari, \end{cases}$$

cioè

$$A_n^{-1} = \begin{pmatrix} 1 & -1 & 1 & \cdots & \pm 1 \\ 0 & 1 & -1 & \cdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1 \\ \vdots & \vdots & \vdots & \ddots & -1 \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

dove si ha che la frazione

$$\lim_{n\to\infty} \frac{nz(A)}{n^2} = \lim_{n\to\infty} \frac{(n^2+n)/2}{n^2} = \frac{1}{2}$$

∘

In figura 1.1 attraverso il comando `spy` di `MATLAB`, riusciamo a vedere lo sparsity pattern delle matrici in esempio. In generale quindi per una matrice sparsa

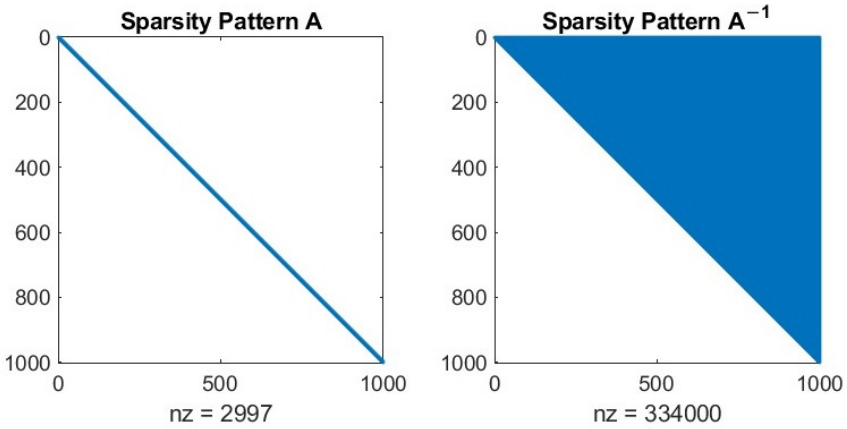


Figura 1.1: Sparsity pattern per $n = 1000$ per la matrice in 1.3.1

l'approssimazione della matrice inversa non è un'operazione a basso costo computazionale. Per questo motivo una delle classi di precondizionatori maggiormente riconosciuta risulta essere la classe dei metodi a fattorizzazione incompleta.

Precondizionatori a fattorizzazione incompleta

I precondizionatori a fattorizzazione incompleta sono metodi, ampiamente studiati e adottati, di approssimazione diretta (i.e. *Forwad based* $M \approx A$). Due precondizionatori largamente diffusi che ricadono in questa categoria sono **iLU** e **iCHOL**. Essi si riferiscono rispettivamente alla fattorizzazione **LU** incompleta, utilizzabile per matrici generiche, e la fattorizzazione incompleta di **Cholesky** nel caso di matrici SPD. Per quanto riguarda le fattorizzazioni, esse sono definite come

Problema 1.1. ***(Fattorizzazione LU)***
Sia $A \in \mathbb{R}^{(n\times n)}$. *Determinare se esistono*

1. *L matrice triangolare inferiore con elementi diagonali uguali a 1*

2. *U matrice triangolare superiore*

avendo quindi

$$A = LU$$

□

Non tutte le matrici possono essere fattorizzate mediante LU. Infatti essa esiste solo se tutti i *minori principali* $A^{(k)} = (a_{ij})_{i,j=1,...,k}$, *con* $k = 1, ..., n-1$ *siano non singolari*. Nel caso in cui questo non si verifichi, per determinare la fattorizzazione LU quello che si può applicare è una matrice di permutazione P tale che

$$PA = LU$$

la quale effettua una permutazione delle righe in modo da non avere elementi pivotali nulli. Nel caso di matrici SPD abbiamo il seguente teorema, che definisce la *fattorizzazione di Cholesky*

Teorema 1.3.3. ***(Fattorizzazione di Cholesky)***
Sia $A \in \mathbb{R}^{n\times n}$ *simmetrica definita positiva SPD. Allora esiste ed è unica la fattorizzazione di Cholesky*

$$A = LL^T$$

con $L = (l_{ij})$ *matrice triangolare inferiore con elementi principali* $l_{ii} > 0$. □

Il costo computazionale delle fattorizzazioni risulta molto oneroso ed è riassunto nella tabella sottostante. Quello che si ottiene, in termini di sparsity pattern, è

Tabella 1.2: Costo computazionale fattorizzazione

Fattorizzazione	$PA = LU$	$A = LL^T$
Matrici	generiche	SPD
Costo	$O(n^3/3)$	$O(n^3/6)$

in generale che $SP\{A\} \subseteq SP\{L + U\}$. L'idea delle fattorizzazioni incomplete è quindi di utilizzare due matrici $\tilde{L}$ e $\tilde{U}$, con uno sparsity pattern

$$S_A := SP\{A\} = SP\{\tilde{L} + \tilde{U}\} \tag{1.31}$$

ottenendo quindi una approssimazione

$$A = \tilde{L}\tilde{U} + E \tag{1.32}$$

con E matrice di errore. Quindi l'idea è di mantenere lo stesso pattern di zeri di A, utilizzando i fattori incompleti $\tilde{L}$ e $\tilde{U}$. Lo stesso metodo è applicato alla fattorizzazione di Choleksy, utilizzando i fattori incompleti $\tilde{L}$ e $\tilde{L^T}$. Questa tecnica è chiamata fattorizzazione **iLU(0)** (**iCHOL(0)**), chiamata anche fattorizzazione

incompleta senza riempimento (no fill-in), e i fattori $\tilde{L}$ e $\tilde{U}$ vengono chiamati comunemente L_0 e U_0.

Algorithm 1: Fattorizzazione LU incompleta

Input: A: matrice $\in \mathbb{R}^{n\times n}$

1 **for** $i = 2 : n$ **do**
2 **for** $k = 1 : i-1$ *and* $(i,k) \in SP\{A\}$ **do**
3 Calcola $a_{ik} = a_{ik}/a_{kk}$;
4 **for** $j = k+1 : n$ *and* $(i,j) \in SP\{A\}$ **do**
5 Calcola $a_{ij} = a_{ij} - a_{ik} * a_{kj}$;

Tuttavia il pattern S_A può risultare insufficiente per ricavare una buona approssimazione di A. L'accuratezza di iLU(0) può essere aumentata al costo di accettare nuovi elementi non nulli laddove A presenta elementi nulli. Per questo motivo si definisce un livello $p \neq 0$ di "fill-in", o di "riempimento" del pattern. Più precisamente il *livello di fill-in* è associato ad ogni elemento a_{ij}, indicato con lev_{ij}, con valore iniziale

$$lev_{ij} \begin{cases} 0 & \text{se} \quad a_{ij} \neq 0 \\ \infty & \text{altrimenti} \end{cases}$$

e durante il processo di eliminazione Gaussiana, aggiorniamo il *livello di fill-in* di un elemento attraverso

$$lev_{ij} = \min\{lev_{ij}, lev_{ik} + lev_{kj} + 1\}.$$

La condizione $(i,k) \in SP\{A\}$ nell'algoritmo 1 diventa $lev_{ik} \leq p$, così il livello p intuitivamente indica gli elementi trasformati in elementi non nulli durante il passo p del processo di eliminazione di Gauss. Infine nel precondizionatore iLU(p) si accetteranno tutti gli elementi che hanno *livello di fill-in* $\leq p$.

Esempio 1.3.2. *fattorizzazione* ***iLU(0)***

Per visualizzare come agisce iLU(0), consideriamo per esempio la matrice tridiagonale a blocchi risultante dalla discretizzazione a differenze finite dell'equazione di Poisson. In `MATLAB` attraverso il comando `gallery('poisson')` è possibile creare tale matrice e visualizzarne lo sparsity pattern. Per convenzione richiamiamo le matrici $\tilde{L}$ e $\tilde{U}$ identificandole attraverso L_0 e U_0 e dalla condizione 1.32 si avrà che

$$A_{ij} = 0 \implies (L_0)_{ij} = (U_0)_{ij} = 0$$

come mostrato nella figura sottostante 1.2, e con $A_0 = L_0 U_0$.

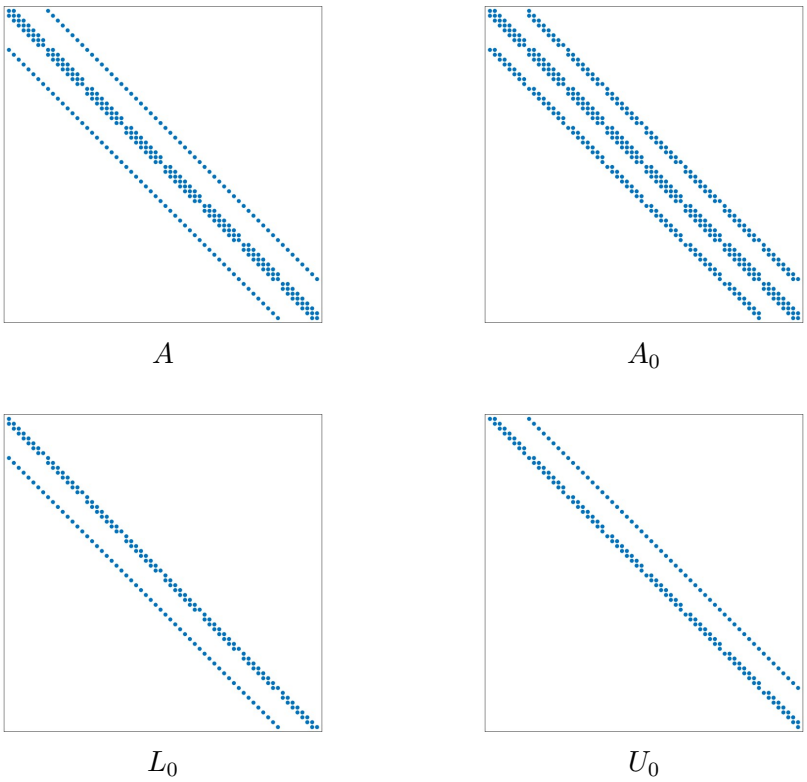


Figura 1.2: fattorizzazione incompleta iLU(0) di A

Si ricava inoltre che la norma $\|A - L_0 U_0\|_2 \approx 0.5$. ∘

Un limite delle fattorizzazioni incomplete è l'instabilità, e il *breakdown* dovuti a pivot nulli. Inoltre Chow e Saad [7] riportano che altre principali cause di fallimento sono l'inesattezza e l'instabilità delle risoluzioni triangolari. Questo può verificarsi quando l'**iLU** viene applicato a sistemi fortemente nonsimmetrici o indefiniti. Un altro importante potenziale svantaggio degli approcci di tipo diretto (i.e. forward $M \approx A$) risiede nel fatto che, in ogni passo di un metodo iterativo, è necessario risolvere un sistema lineare in M. Questo non deve risultare più costoso che risolvere il sistema originale. Di conseguenza, i precondizionatori di tipo *forward* devono essere costruiti in modo da consentire una inversione rapida e facile di M, ad esempio in forma diagonale o triangolare. Questo è un vincolo che la categoria dei precondizionatori inversi non soffre. Quello che analizzeremo nel prossimo capitolo saranno appunto due tipi di precondizionatori basati sull'inversa $M \approx A^{-1}$, che sfruttano un problema di minimizzazione della norma di Frobenius della quantità $\|I - AM\|_F$. Il vantaggio di questi due precondizionatori, **SPAI** e

FSAI, risiede appunto nel costruire direttamente una approssimazione dell'inversa, e di essere basati, come vedremo successivamente, su un algoritmo che si presta naturalmente all'esecuzione parallela.

Capitolo 2

SPAI

Il metodo SPAI, inizialmente proposto in [10], così come anche il metodo FSAI, sviluppato in [14], appartengono alla categoria degli *sparse approximate inverse*. L'idea base del metodo SPAI, come già anticipato precedentemente, è di approssimare l'inversa di una matrice sparsa attraverso un problema di minimizzazione, che si riduce nella soluzione di n problemi ai minimi quadrati indipendenti. Questo fa si che l'algoritmo si presenta intrinsecamente parallelo, potendo sfruttare così il vantaggio dei moderni calcolatori, basati su una architettura parallela del processore. Questo fa si che anche un problema computazionalmente costoso, risolto in parallelo, possa in termini reali avere un pratico utilizzo. Uno dei passaggi chiave di questi tipi di algoritmi è lo sparsity pattern della matrice finale $M \approx A^{-1}$. Vedremo come nello SPAI sia possibile aggiungere nuovi indici di elementi diversi da zero risolvendo un problema di minimizzazione. Tuttavia, anche se è possibile in modo dinamico aggiornare lo sparsity pattern iniziale di M dato in input all'algoritmo, l'euristica gioca un ruolo fondamentale nel selezionare *a priori* lo sparsity pattern di M. Si approfondirà infatti come identificare il pattern dell'inversa A^{-1} e di come usare questa informazione per una euristica a priori di $SP\{M\}$.

2.1 Minimizzazione della norma di Frobenius

L'idea alla base del metodo *sparse approximate inverse* è di costruire direttamente un'approssimazione dell'inversa, considerando quindi $M \approx A^{-1}$. L'idea principale è quella di trovare una matrice M, tale da minimizzare la quantità

$$\|I - AM\|_F \tag{2.1}$$

con $SP\{M\} \in \mathcal{S}$ sparsity pattern preconfigurato. Possiamo quindi ricavare che

$$\|I - AM\|_F^2 = \sum_{k=1}^{n} \|(I - AM)e_k\|_2^2 = \sum_{k=1}^{n} \|e_k - Am_k\|_2^2 \tag{2.2}$$

con e_k colonna k-esima della matrice identità. La possibilità di suddividere la norma in una somma di norme Euclidee è infatti una proprietà della norma di Frobenius. Il problema quindi si riassume in n problemi ai minimi quadrati risolvibili in modo indipendente.

$$\min_{m_k} \|e_k - Am_k\|_2^2, \quad con \quad k = 1, ..., n \tag{2.3}$$

con $SP\{m_k\} \in \mathcal{J}$ pattern predefinito di m_k. Ottenendo che è proprio questa formulazione che rende lo SPAI un precondizionatore vantaggioso, ovvero la possibilità di parallelizzare l'algoritmo senza aggiungere complessità implementativa, grazie alla natura parallelizzabile dell'algoritmo stesso. Il passaggio principale quindi è risolvere il problema dei minimi quadrati colonna per colonna.

2.1.1 Fattorizzazione QR per problemi ai minimi quadrati

Esistono diversi metodi per risolvere problemi ai minimi quadrati, come il problema 2.3. Uno dei metodi più diffusi riguarda l'uso della fattorizzazione QR della matrice A, di cui inoltre è riconosciuta la stabilità anche per sistemi malcondizionati, soprattutto nell'implementazione comunemente chiamata *Householder QR*. Consideriamo $A \in \mathbb{R}^{m \times n}$, $m \geq n$, e consideriamo che A ha rango delle colonne pieno, ovvero $rank(A) = n$, allora A ha un unica fattorizzazione del tipo

$$A = QR \qquad con \quad Q \in \mathbb{R}^{m \times m},\ R \in \mathbb{R}^{m \times n} \tag{2.4}$$

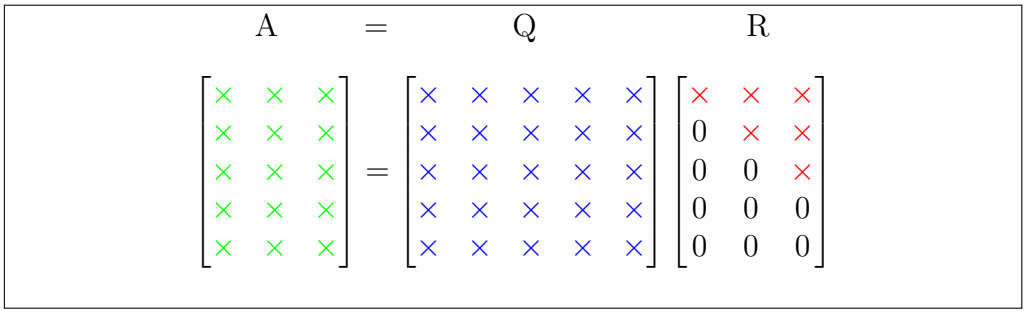


Figura 2.1: Esempio decomposizione QR

Con Q matrice ortogonale e R matrice triangolare superiore. Una rappresentazione

alternativa di questa fattorizzazione, si può ottenere considerando la matrice R ridotta, ottenendo quindi

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix}, \quad con \quad R \in \mathbb{R}^{n \times n} \tag{2.5}$$

Per calcolare questa fattorizzazione, si possono utilizzare i riflettori di Householder, che consentono di annullare gli elementi al di sotto della diagonale di A in modo sistematico.

Procedura per la fattorizzazione QR

La fattorizzazione QR può essere realizzata attraverso diversi metodi, che variano a seconda della specifica applicazione e delle proprietà numeriche richieste. La fattorizzazione QR mediante i riflettori di Householder si realizza iterativamente applicando i riflettori a ciascuna colonna di A per ridurla a forma triangolare superiore. L'algoritmo procede come segue:

Definizione 2.1.1. *Un **riflettore di Householder** è una matrice della forma:*

$$H = I - 2\frac{vv^T}{\|v\|^2},$$

dove $v \in \mathbb{R}^m$ è un vettore non nullo.

La matrice H è simmetrica ($H = H^T$) e ortogonale ($H^T H = I$). Il riflettore H è costruito in modo da trasformare un dato vettore $x \in \mathbb{R}^m$ in un vettore multiplo di un vettore base $e_1 = (1, 0, \ldots, 0)^T$.

$$Hx = \tau e_1$$

con $\tau \in \mathbb{R}$, e poiché H è ortogonale si ha che

$$\|Hx\| = \|x\| = |\tau|$$

considerando quindi la matrice $A \in \mathbb{R}^{m \times n}$ rappresentata come vettore di colonne

$$A = [a_1 \,|\, a_2 \,|\ldots|\, a_n], \quad con \quad a_i \text{ colonna } i\text{-esima di } A$$

per il primo passo ($k = 1$), consideriamo il vettore colonna $a_1 \in \mathbb{R}^m$ (la prima colonna di A) e costruiamo il vettore v_1 per annullare tutti gli elementi sotto il primo:

$$v_1 = a_1 + \text{sign}(a_{11})\|a_1\|_2 \, e_1,$$

dove $e_1 \in \mathbb{R}^m$ è il primo vettore della base canonica e $\text{sign}(a_{11})$ è il segno della componente a_{11}. Definiamo il riflettore di Householder:

$$H_1 = I - 2\frac{v_1 v_1^T}{v_1^T v_1}.$$

applicato quindi a sinistra di A, ottenendo la matrice

$$A^{(2)} = H_1 A = \left[a_1^{(2)} \,|\, a_2^{(2)} \,|...|\, a_n^{(2)}\right], \quad con \quad a_1^{(2)} = \tau e_1$$

e si definisce ricorsivamente

$$H_2 = \begin{pmatrix} 1 & 0 \\ 0 & I_{m-1} - 2\dfrac{v_2 v_2^T}{v_2^T v_2} \end{pmatrix}, \quad da\ cui \longrightarrow H_k = \begin{pmatrix} I_{k-1} & 0 \\ 0 & I_{m-k-1} - 2\dfrac{v_k v_k^T}{v_k^T k_2} \end{pmatrix}$$

e si costruisce così una matrice triangolare superiore R

$$H_n ... H_2 H_1 A = R$$

Grazie alle proprietà delle matrici ortogonali si ottiene che

$$A = H_1^T H_2^T ... H_n^T H_n ... H_2 H_1 A = H_1^T H_2^T ... H_n^T R = QR$$

con

$$Q := H_1^T H_2^T ... H_n^T \tag{2.6}$$

L'algoritmo dei riflettori di Householder è numericamente stabile e riduce l'errore di round-off rispetto ad altri metodi, come il metodo di Gram-Schmidt. Inoltre, la triangolazione attraverso i riflettori di Householder ha complessità computazionale $O(2mn^2 - 2n^3/3)$, il metodo quindi risulta abbastanza efficiente per matrici dense.

Space complexity della fattorizzazione QR.

Per quanto riguarda la *space complexity*, quindi il costo in termini di memoria, dell'algoritmo QR tramite Householder, dobbiamo considerare la struttura triangolare della matrice R e la procedura di costruzione della matrice Q. Considerando la matrice A di dimensioni $m \times n$, la matrice R che è triangolare superiore, nella sua forma ridotta è una matrice $n \times n$, e il costo per memorizzare tale matrice è la somma degli $\left(\frac{n(n-1)}{2}\right)$ elementi sopra la diagonale, e degli elementi della diagonale stessa, ottenendo un costo pari a $\left(\frac{n(n+1)}{2}\right)$ elementi totali. Per quanto riguarda la matrice Q, tale matrice è una matrice piena di dimensioni $m \times m$, con costo di memorizzazione $O(m^2)$. Data la richiesta elevata di spazio, nascono problemi

di memorizzazione all'aumentare della dimensione m. Per cercare di ridurre la memoria nei contesti ove necessario, una possibile ottimizzazione risulta essere la memorizzazione dei soli vettori v_k corrispondenti ai riflettori di Householder. La eventuale costruzione di Q in questo caso viene calcolata al momento.

Soluzione del problema ai minimi quadrati.

La soluzione del problema ai minimi quadrati rappresenta una delle applicazioni più rilevanti della fattorizzazione QR, in particolare per la sua efficienza computazionale e per la stabilità numerica. La fattorizzazione QR consente di trasformare il problema originale in un sistema equivalente più semplice da risolvere.

Nel caso di un sistema che risulti sovradeterminato, in cui sono presenti più equazioni che incognite, il sistema sarà del tipo

$$Ax = b$$

dove si avrà una matrice rettangolare $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ e $x \in \mathbb{R}^n$. In questo caso quindi, quello che è ragionevole chiederci è quando lo scarto quadratico medio tra primo e secondo membro sia minimo:

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

la cui soluzione coincide con la soluzione originale del sistema $Ax = b$ nel caso di A matrice quadrata non singolare. Utilizzando la fattorizzazione QR è possibile

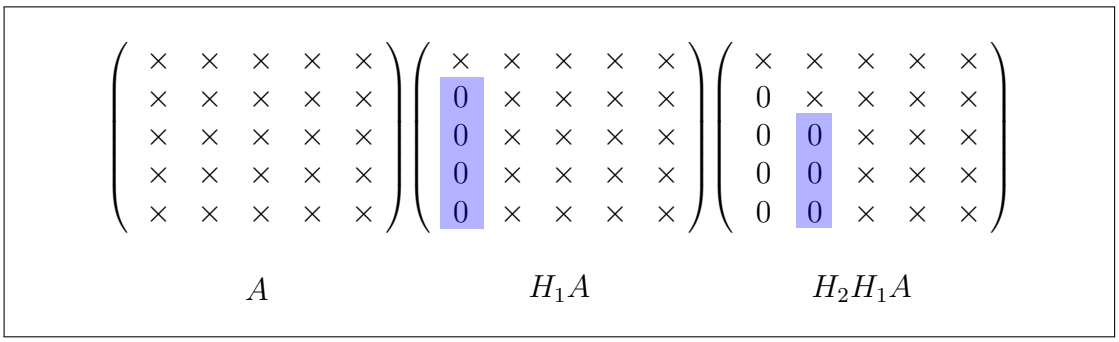


Figura 2.2: Triangolazione di A mediante riflettori di Householder.

risolvere il sistema considerando il quadrato del residuo:

$$\begin{aligned} \|Ax-b\|^2 &= \|QRx-b\|^2 = \|QRx-b\|^2 \\ &= \|Q(Rx-Q^Tb)\|^2 \\ &= \|Rx-Q^Tb\|^2 \end{aligned} \tag{2.7}$$

e scrivendo la matrice R in forma ridotta, ossia con

$$R = \begin{pmatrix} \hat{R} \\ 0 \end{pmatrix}, \qquad \hat{R} \in \mathbb{R}^{n\times n}$$

si potrà partizionare il vettore Q^Tb come:

$$Q^Tb = c = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}, \quad \text{con} \quad c_1 \in \mathbb{R}^n, \quad c_2 \in \mathbb{R}^{m-n} \tag{2.8}$$

Così da ottenere

$$\begin{aligned} \|Rx-c\|^2 &= \|\begin{pmatrix} \hat{R}x \\ 0 \end{pmatrix} - \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}\|^2 \\ &= \|\hat{R}x-c_1\|^2 + \|c_2\|^2 \end{aligned} \tag{2.9}$$

e la minimizzazione si ha risolvendo quindi il sistema

$$\hat{R}x = c_1$$

ricavando x e si ottiene quindi che

$$\min_{x\,\in\,\mathbb{R}^n} \|Ax-b\|_2 = \|c_2\|_2 \tag{2.10}$$

Rifacendo allora una analogia con il sistema lineare $Ax = b$, nel caso di A matrice quadrata non singolare, il vettore c_2 è un vettore nullo.

Osservazioni sulla notazione. Nel seguente paragrafo utilizzeremo alcune notazioni coerenti con il linguaggio `MATLAB` [16], dove per esempio con $A(:,i)$ si considera la colonna i-esima della matrice A.

2.1.2 Formulazione e Risoluzione del Problema

Ritornando al problema di minimo 2.3, sappiamo che la matrice di cui vogliamo approssimare l'inversa è sparsa, quindi avrà una densità molto consistente di elementi uguali a zero. Inoltre, la matrice $M \approx A^{-1}$ ricercata avrà anch'essa una

struttura sparsa. Considerata quindi la colonna k-esima di M, indicandola con m_k, si considera lo sparsity pattern iniziale $\mathcal{J}$ della colonna k-esima di M. Questo fa si che il problema dei minimi quadrati per A, si può ricondurre ad un problema dei minimi quadrati su una sottomatrice $A(\mathcal{I}, \mathcal{J})$, con $\mathcal{I}, \mathcal{J}$ insieme di indici di riga e colonna opportuni.

$$A(\mathcal{I}, \mathcal{J}) = \left[a_{ij}\right]_{i\in\mathcal{I},\, j\in\mathcal{J}} \quad \text{con} \quad \mathcal{I}, \mathcal{J} \subseteq \{1, \dots, n\}$$

L'idea è quindi di ricondurre il problema ad un sistema ridotto, utilizzando l'informazione della sparsità della matrice A. Iniziamo considerando le matrici $A, M \in \mathbb{R}^{n\times n}$ e con m_k la colonna k-esima di M, e data la natura parallela, possiamo quindi descrivere l'algoritmo per una generica colonna m_k. Consideriamo

$$\begin{aligned} \mathcal{J} &\coloneqq \{j : \ m_k(j) \neq 0, \ 1 \leq j \leq n\} \\ n_2 &\coloneqq |\mathcal{J}| \end{aligned} \tag{2.11}$$

il quale racchiude l'informazione dello sparsity pattern di m_k, e indichiamo il vettore delle incognite $m_k(\mathcal{J})$ con $\hat{m}_k$. La matrice ridotta sarà quindi $A(\mathcal{I}, \mathcal{J})$ con $\mathcal{I}$, insieme degli indici delle righe di $A(i, \mathcal{J})$ diverse da 0, ovvero

$$\begin{aligned} \mathcal{I} &= \{i : \ \sum_{j\in\mathcal{J}} |a_{ij}| \neq 0\} \\ n_1 &\coloneqq |\mathcal{I}| \end{aligned} \tag{2.12}$$

Il sistema così ottenuto sarà del tipo

$$\begin{aligned} \hat{m}_k &\coloneqq m_k(\mathcal{J}) \in \mathbb{R}^{n_2} \\ \hat{A} \ &\coloneqq A(\mathcal{I}, \mathcal{J}) \in \mathbb{R}^{n_1\times n_2} \\ \hat{e}_k \ &\coloneqq e_k(\mathcal{I}) \in \mathbb{R}^{n_1} \end{aligned} \tag{2.13}$$

Il problema 2.3 si riconduce quindi al problema ridotto

$$\min_{\hat{m}_k} \|\hat{e}_k - \hat{A}\hat{m}_k\|_2^2, \quad con \quad k = 1, ..., n \tag{2.14}$$

Il sistema così ottenuto risulta essere un sistema denso, inoltre, se A è non singolare, si ricava che $\hat{A}$ ha rango delle colonne pieno n_2. Si può utilizzare quindi la decomposizione QR di $\hat{A}$ per risolvere il problema ai minimi quadrati. Il sistema dunque ha dimensioni $n_1 \times n_2$, che è molto minore rispetto alle dimensioni di A, perché sia A che M sono matrici sparse. Così consideriamo la fattorizzazione di $\hat{A}$ direttamente in forma ridotta

$$\hat{A} = Q \begin{pmatrix} R \\ 0 \end{pmatrix} \tag{2.15}$$

con R matrice triangolare superiore non singolare di dimensioni $n_2 \times n_2$. Allora otteniamo

$$\hat{c} = Q^T \hat{e}_k \implies \hat{m}_k = R^{-1}\hat{c}\,(1 : n_2) \tag{2.16}$$

Risolvendo il sistema 2.16 per ogni colonna k, possiamo aggiornare il valore di m_k fissando $m_k(\mathcal{J}) = \hat{m}_k$. Questo ci permette di ottenere una approssimazione dell'inversa M, secondo 2.2, per un sparsity pattern definito. Per ottenere una buona approssimazione quindi risulta fondamentale un'euristica sullo sparsity pattern di M, e in generale sull'inversa A^{-1}, con proprietà che analizzeremo successivamente. Poniamo invece adesso l'obiettivo di analizzare i residui, colonna per colonna, con l'idea di aggiornare lo sparsity pattern di M aggiungendo nuovi indici a $SP\{M\}$. I residui ottenuti risolvendo il problema ai minimi quadrati 2.3, sono pari a

$$r_k = A(:, \mathcal{J})\,\hat{m}_k - e_k \tag{2.17}$$

Se $r_k = 0$, allora m_k è esattamente la k-esima colonna di A^{-1}. Se invece $r_k \neq 0$, possiamo migliorare l'approssimazione aggiungendo indici a $\mathcal{J}$, in modo da ridurre $\|r_k\|_2$. Consideriamo quindi l'insieme degli indici $\mathcal{L}$ tali che il residuo sia diverso da zero

$$\mathcal{L} = \{l : r_k(l) \neq 0\}. \tag{2.18}$$

Considerando la definizione di $\mathcal{I}$, e gli errori di arrotondamento, avremo che $\mathcal{L}$ sarà uguale a $\mathcal{I} \cup k$, unito k poiché se $k \notin \mathcal{I} \implies r(k) = -1$. Definiamo quindi per ogni l l'insieme $\mathcal{N}_l$, corrispondente all'insieme degli indici degli elementi diversi da zero di $A(l, :)$ non presenti ancora in $\mathcal{J}$. Ottenendo quindi

$$\mathcal{N}_l = \{j : a_{lj} \neq 0,\ j \notin \mathcal{J}\} \tag{2.19}$$

e infine indicando l'unione di tutti gli $\mathcal{N}_l$ attraverso

$$\tilde{\mathcal{J}} = \bigcup_{l \in \mathcal{L}} \mathcal{N}_l \tag{2.20}$$

con $\tilde{\mathcal{J}} \cap \mathcal{J} = \emptyset$, e dove in $\tilde{\mathcal{J}}$ sono inclusi i potenziali nuovi indici da aggiungere in $\mathcal{J}$. Per poter individuare i potenziali indici j da aggiungere, si considera la riduzione del residuo $\|r_k\|_2$. Per conseguire tale risultato in modo computazionalmente economico ma efficace si considera il problema di minimizzazione unidimensionale per ogni $j \in \tilde{\mathcal{J}}$

$$\min_{\mu_j} \|r_k + \mu_j A e_j\|_2 \tag{2.21}$$

Considerando $Ae_j = a_j$ colonna j-esima di A, otteniamo

$$\min_{\mu_j} \|r_k + \mu_j a_j\|_2^2 = \mu_j^2 \|a_j\|_2^2 + 2\mu_j \langle r_k, a_j \rangle + \|r_k\|_2^2$$

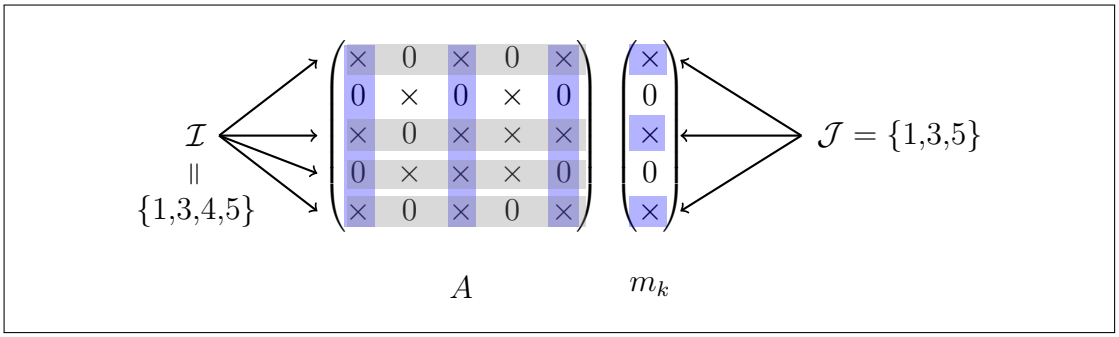


Figura 2.3: Processo di selezione della sottomatrice $\hat{A}$ dato $\mathcal{J}$.

e differenziando per μ_j ed eguagliando a zero otteniamo

$$2\mu_j \|a_j\|_2^2 + 2\langle r_k, a_j \rangle = 0$$

con soluzione

$$\mu_j = -\frac{r_k^T a_j}{\|a_j\|_2^2} \tag{2.22}$$

Quindi per verificare quale dei potenziali indici produca una maggiore riduzione del residuo $\|r_k\|_2$, si considera la quantità ρ_j per ogni indice j tale che

$$\rho_j^2 := \|r_k + \mu_j A e_j\|_2^2 = \|r_k\|_2^2 - \frac{(r_k^T A e_j)^2}{\|A e_j\|_2^2} \tag{2.23}$$

e sapendo che esiste almeno un indice $j \in \tilde{\mathcal{J}}$ con $r_k^T A e_j \neq 0$, che produca una diminuzione del residuo in 2.23, poiché per il seguente teorema

Teorema 2.1.1. *Se il residuo è $r_k \neq 0$ allora esiste almeno un indice j tale che*

$$\frac{(r_k^T A e_j)^2}{\|A e_j\|_2^2} > 0$$

Dimostrazione. Assumendo $r_k \neq 0$, e $\frac{(r_k^T A e_j)^2}{\|A e_j\|_2^2} = 0$ per ogni $j \in \{1, ..., n\}$, otteniamo

$$\forall\, j \;:\; \frac{(r_k^T A e_j)^2}{\|A e_j\|_2^2} = 0 \iff \forall\, j : e_j^T A^T r_k = 0 \iff A^T r_k = 0 \overset{\text{A non singolare}}{\iff} r_k = 0$$

che contraddice l'assunzione iniziale. □

Si nota inoltre che $\mathcal{J} \cup \tilde{\mathcal{J}}$ contiene gli indici delle colonne di tutti gli elementi diversi da zero di $A(\mathcal{L}, :)$. Si procede quindi riducendo $\tilde{\mathcal{J}}$ all'insieme degli indici

più vantaggiosi da aggiungere a $\mathcal{J}$, selezionando i ρ_j più piccoli. Inoltre denotiamo con $\tilde{\mathcal{I}}$ le nuove righe, che corrispondono all'insieme degli indici di riga diversi da zero di $A(:, \mathcal{J} \cup \tilde{\mathcal{J}})$ non ancora presenti in $\mathcal{I}$, ovvero con $\mathcal{I} \cap \tilde{\mathcal{I}} = \emptyset$. Aumentando quindi ad ogni iterazione il numero di indici di m_k, si risolve nuovamente il problema ai minimi quadrati, con uno sparsity pattern di M aumentato e una nuova sottomatrice di A. Indicando con $\tilde{n}_1$ e $\tilde{n}_2$ la cardinalità degli insiemi $\tilde{\mathcal{I}}$ e $\tilde{\mathcal{J}}$ rispettivamente, si ha che la nuova sottomatrice di A è definita come

$$\tilde{A} = A(\mathcal{I} \cup \tilde{\mathcal{I}}, \mathcal{J} \cup \tilde{\mathcal{J}}) \tag{2.24}$$

di dimensioni $(n_1 + \tilde{n}_1) \times (n_2 + \tilde{n}_2)$. Il nuovo problema quindi risulta

$$\min_{m_k(\mathcal{J} \cup \tilde{\mathcal{J}})} \|A(\mathcal{I} \cup \tilde{\mathcal{I}}, \mathcal{J} \cup \tilde{\mathcal{J}}) m_k(\mathcal{J} \cup \tilde{\mathcal{J}}) - e_k(\mathcal{I} \cup \tilde{\mathcal{I}})\|_2 \tag{2.25}$$

Per risolvere questo problema possiamo utilizzare la decomposizione QR di $\tilde{A}$, come già effettuato per la il problema non aumentato. Per diminuire il costo computazionale, e per far si che non sia necessario ricalcolare interamente la decomposizione, si può utilizzare la precedente decomposizione QR di $\hat{A}$ con un processo di aggiornamento per calcolare la decomposizione della matrice aumentata $\tilde{A}$.

2.1.3 Decomposizione QR con aggiornamento iterativo

Per risolvere il problema aumentato 2.25, si può aggiornare la precedente decomposizione QR, considerando il partizionamento di $\tilde{A}$ in

$$\tilde{A} = A(\mathcal{I} \cup \tilde{\mathcal{I}}, \mathcal{J} \cup \tilde{\mathcal{J}}) = \begin{pmatrix} \hat{A} & A(\mathcal{I}, \tilde{\mathcal{J}}) \\ A(\tilde{\mathcal{I}}, \mathcal{J}) & A(\tilde{\mathcal{I}}, \tilde{\mathcal{J}}) \end{pmatrix} = \begin{pmatrix} \hat{A} & A(\mathcal{I}, \tilde{\mathcal{J}}) \\ 0 & A(\tilde{\mathcal{I}}, \tilde{\mathcal{J}}) \end{pmatrix} \tag{2.26}$$

dove $A(\tilde{\mathcal{I}}, \mathcal{J}) = 0$ poiché $\mathcal{I}$ contiene già tutte le righe di $A(:, \mathcal{J})$ con elementi diversi da zero. Si può inoltre riscrivere $\tilde{A}$ come prodotto di matrici:

$$\tilde{A} = \begin{pmatrix} \hat{A} & A(\mathcal{I}, \tilde{\mathcal{J}}) \\ 0 & A(\tilde{\mathcal{I}}, \tilde{\mathcal{J}}) \end{pmatrix} = \begin{pmatrix} Q & \\ & I_{\tilde{n}_1} \end{pmatrix} \begin{pmatrix} R & Q^T A(\mathcal{I}, \tilde{\mathcal{J}}) \\ 0 & A(\tilde{\mathcal{I}}, \tilde{\mathcal{J}}) \end{pmatrix} \tag{2.27}$$

dove il prodotto $Q^T A(\mathcal{I}, \tilde{\mathcal{J}})$ a sua volta può essere partizionato, definendo dunque $Q_1^T A(\mathcal{I}, \tilde{\mathcal{J}})$ di dimensioni $n_2 \times \tilde{n}_2$, e $Q_2^T A(\mathcal{I}, \tilde{\mathcal{J}})$ di dimensioni $(n_1 - n_2) \times \tilde{n}_2$. Così facendo, si può riscrivere la matrice $\tilde{A}$ come

$$\tilde{A} = \begin{pmatrix} Q & \\ & I_{\tilde{n}_1} \end{pmatrix} \left(\begin{array}{c|c} R & Q_1^T A(\mathcal{I}, \tilde{\mathcal{J}}) \\ \hline 0 & Q_2^T A(\mathcal{I}, \tilde{\mathcal{J}}) \\ 0 & A(\tilde{\mathcal{I}}, \tilde{\mathcal{J}}) \end{array}\right) = \begin{pmatrix} Q & \\ & I_{\tilde{n}_1} \end{pmatrix} \begin{pmatrix} R & B_1 \\ 0 & B_2 \end{pmatrix} \tag{2.28}$$

dove B_2 è una matrice di dimensioni $(\tilde{n}_1 + n_1 - n_2) \times \tilde{n}_2$. Con questa riscrittura della matrice $\tilde{A}$, si richiede solamente la decomposizione QR della matrice B_2

$$B_2 = \tilde{Q} \begin{pmatrix} \tilde{R} \\ 0 \end{pmatrix} \tag{2.29}$$

per ottenere una riscrittura di $\tilde{A}$ nella forma finale

$$\tilde{A} = \begin{pmatrix} Q & \\ & I_{\tilde{n}_1} \end{pmatrix} \begin{pmatrix} I_{n_2} & \\ & \tilde{Q} \end{pmatrix} \begin{pmatrix} R & B_1 \\ 0 & \tilde{R} \\ 0 & 0 \end{pmatrix} \tag{2.30}$$

notando che

$$\begin{pmatrix} Q & \\ & I_{\tilde{n}_1} \end{pmatrix} \begin{pmatrix} Q^T & \\ & I_{\tilde{n}_1} \end{pmatrix} = I_{n_1+\tilde{n}_1}$$

e altresì

$$\begin{pmatrix} I_{n_2} & \\ & \tilde{Q} \end{pmatrix} \begin{pmatrix} I_{n_2} & \\ & \tilde{Q}^T \end{pmatrix} = I_{n_1+\tilde{n}_1}$$

si ottiene la decomposizione QR di $\tilde{A}$ come

$$\tilde{A} = \underbrace{\begin{pmatrix} Q & \\ & I_{\tilde{n}_1} \end{pmatrix} \begin{pmatrix} I_{n_2} & \\ & \tilde{Q} \end{pmatrix}}_{Q_{new}} \underbrace{\begin{pmatrix} R & B_1 \\ 0 & \tilde{R} \\ 0 & 0 \end{pmatrix}}_{R_{new}} = Q_{new} R_{new} \tag{2.31}$$

procedendo così a risolvere il sistema aumentato per m_k, utilizzando la nuova fattorizzazione. Questa procedura ci permette di aggiungere nuovi indici in $\mathcal{J}$ risolvendo il nuovo problema aumentato ai minimi quadrati, senza necessariamente ricalcolare tutta la decomposizione QR della nuova sottomatrice aumentata di A. Ovviamente se non si introducono criteri d'arresto, l'algoritmo calcolerà l'intera $k-$esima colonna di A^{-1}. Per limitare quindi la densità di M si adottano dei criteri di arresto riguardo la tolleranza per l'errore r_k, o un limite sui nuovi indici da aggiungere. L'algoritmo 2 riassume sotto forma di pseudocodice la procedura per il calcolo mediante SPAI.

Osservazioni:

1. Lo sparsity pattern iniziale di M è arbitrario. Si può utilizzare un euristica per trovare uno sparsity pattern iniziale più adatto per l'approssimazione dell'inversa. La scelta iniziale può essere una struttura diagonale se non abbiamo nessuna informazione sulla struttura di A^{-1}. Uno dei vantaggi di poter selezionare lo sparsity pattern iniziale è che se abbiamo diversi problemi che presentano uno sparsity pattern simile, allora possiamo utilizzare lo sparsity

pattern del problema precedente già risolto come input.

2. il ciclo `while` nell'algoritmo 2 può anche contenere una differente condizione, riguardo per esempio il numero massimo di iterazioni o numero massimo di indici da aggiungere per colonna. La condizione ovviamente risiede nel mantenere comunque una struttura sparsa della matrice M

3. L'algoritmo è dato per singola colonna poiché intrinsecamente parallelo. La matrice M finale è la combinazione di tutte le colonne m_k calcolate separatamente.

4. Per la riduzione e selezione degli indici presenti in $\tilde{\mathcal{J}}$ nell'algoritmo 2 si possono utilizzare differenti procedure. La più semplice riguarda la scelta degli s indici con ρ_j^2 minore. Si può anche considerare una media di tutti i ρ_j^2, e successivamente considerare solamente i ρ_j^2 con valore minore o uguale alla media, di cui si prendono soltanto i primi $q \leq s$ elementi più piccoli.

5. L'approssimazione dell'inversa M è invariante per permutazioni. Se A è sostituita da $P_1 A P_2$, dove P_1, P_2 sono matrici di permutazione, otteniamo come risultato $P_2^T M P_1^T$.

Il metodo SPAI è stato progettato per essere applicabile a matrici generiche, senza richiedere ipotesi particolari sulla struttura o sulle proprietà della matrice originale. La procedura mostrata non presenta restrizioni e può essere utilizzata per matrici generiche, sia simmetriche che non simmetriche, definite positive o indefinite.

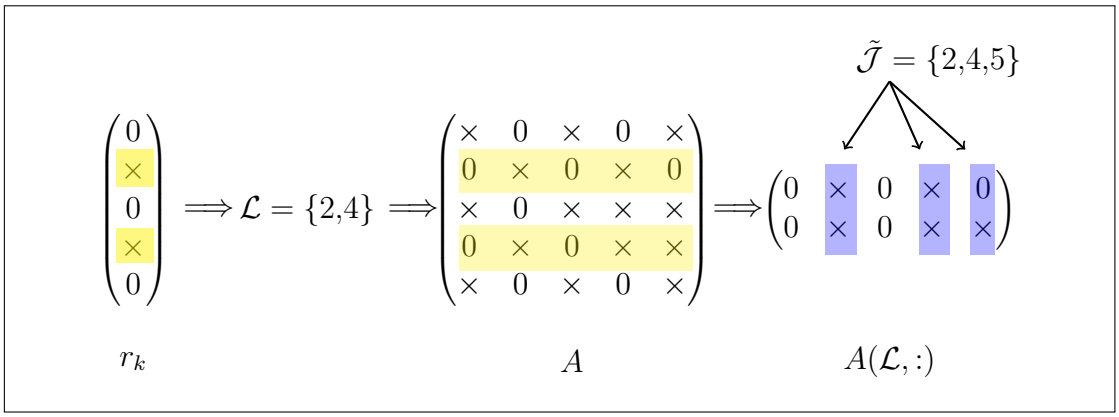


Figura 2.4: Illustrazione del processo di calcolo dei potenziali nuovi indici $\tilde{\mathcal{J}}$.

Algorithm 2: SPAI

Input: A: matrice $n \times n$ sparsa, tol: tolleranza, $\mathcal{J}$: sparsity pattern

Output: M: matrice $n \times n$ con $M \approx A^{-1}$

foreach $k = 1 : n$ **do**
- Determinare l'insieme $\mathcal{I}$ delle componenti non nulle di $A(:, \mathcal{J})$;
- Definire $\hat{A} = A(\mathcal{I}, \mathcal{J})$;
- Eseguire la decomposizione QR di $\hat{A}$;
- $\hat{c} = Q^T \hat{e}_k$;
- $\hat{m}_k = R^{-1}\hat{c}$;
- Calcolare il residuo r_k;
- **while** $\|r_k\| > tol$ **do**
  - Definire l'insieme $\mathcal{L}$ degli indici l tali che $r_k(l) \neq 0$;
  - Identificare l'insieme dei potenziali nuovi indici $\tilde{\mathcal{J}}$;
  - **foreach** $j \in \tilde{\mathcal{J}}$ **do**
    - $\rho_j^2 = \|r_k\|_2^2 - \frac{(r_k^T A e_j)^2}{\|A e_j\|_2^2}$;
  - Ridurre $\tilde{\mathcal{J}}$ ai s indici j corrispondenti ai minimi valori di ρ_j^2;
  - Identificare i nuovi indici di riga $\tilde{\mathcal{I}}$;
  - Aggiornare $\mathcal{I} \leftarrow \mathcal{I} \cup \tilde{\mathcal{I}}$ e $\mathcal{J} \leftarrow \mathcal{J} \cup \tilde{\mathcal{J}}$;
  - Aggiornare la decomposizione QR;
  - Costruire $A(\mathcal{I}, \tilde{\mathcal{J}})$ e $A(\tilde{\mathcal{I}}, \tilde{\mathcal{J}})$;
  - $\bar{A} = Q^T A(\mathcal{I}, \tilde{\mathcal{J}})$;
  - Determinare $B_1 = \bar{A}(0 : n_2, 0 : \tilde{n}_2)$;
  - Determinare
$$B_2 = \begin{pmatrix} \bar{A}(n_2 + 1 :, 0 : \tilde{n}_2) \\ A(\tilde{\mathcal{I}}, \tilde{\mathcal{J}}) \end{pmatrix};$$
  - Eseguire la decomposizione QR di B_2;
  - Definire Q_{new} e R_{new} dalla decomposizione aggiornata;
  - Risolvere il problema ai minimi quadrati aumentato per $\hat{m}_k$;
  - Calcolare il residuo r_k;
  - Aggiornare $\mathcal{I} \leftarrow \mathcal{I} \cup \tilde{\mathcal{I}}$, $\mathcal{J} \leftarrow \mathcal{J} \cup \tilde{\mathcal{J}}$
- Assegna $m_k(\mathcal{J}) = \hat{m}_k$;

Assegna m_k come k-esima colonna di M;

2.1.4 Proprietà teoriche

Una delle questioni fondamentali consiste nell'individuare le condizioni sotto le quali la matrice M risulta non singolare. Una condizione semplice per lo studio della singolarità di M è data dal seguente teorema riportato in [17]

Teorema 2.1.2. *Sia A non singolare e M approssimazione dell'inversa, allora se*

$$\|I - AM\| < 1$$

con $\|\cdot\|$ norma consistente. Allora M è non singolare.

Dimostrazione. Il risultato deriva dalla seguente uguaglianza

$$AM = I - (I - AM) \equiv I - N$$

e dato che se $\|N\| < 1$ allora $I - N$ è non singolare e M è non singolare. □

Grote e Huckle nel loro articolo [10] presentano inoltre altri risultati teorici riguardo il metodo SPAI. Indicando con r_k il residuo della colonna m_k, e assumendo che

$$\|r_k\| = \|Am_k - e_k\| < \varepsilon \tag{2.32}$$

Teorema 2.1.3. *Sia $p = \max_{1\leq k\leq n}\{$numero di elementi non-zero di $r_k\}$. Allora*

$$\|AM - I\|_F \leq \sqrt{n}\varepsilon$$
$$\|AM - I\|_2 \leq \sqrt{n}\varepsilon$$
$$\|AM - I\|_1 \leq \sqrt{p}\varepsilon$$

e inoltre

$$\|M - A^{-1}\|_F \leq \|A^{-1}\|_2 \sqrt{n}\varepsilon$$
$$\|M - A^{-1}\|_2 \leq \|A^{-1}\|_2 \sqrt{n}\varepsilon$$
$$\|M - A^{-1}\|_1 \leq \|A^{-1}\|_1 \sqrt{p}\varepsilon$$

Data la struttura sparsa di A e di M, sappiamo che $p \ll n$. Inoltre possiamo si ricava il seguente corollario

Corollario 1. *Se $\sqrt{p}\varepsilon < 1$, allora M è non singolare.*

Come già discusso nel capitolo precedente, la convergenza dei metodi iterativi dipende dalla distribuzione degli autovalori o dei valori singolari. Quindi è importante derivare delle stime riguardanti lo spettro di AM. I seguenti due teoremi riassumono i risultati riguardante lo spettro di AM.

Teorema 2.1.4. *Sia* $p = \max_{1\leq k\leq n}\{$*numero di elementi non-zero di* $r_k\}$*. Allora gli autovalori* λ_k *di* AM *sono concentrati in* 1 *e ricadono dentro un cerchio di raggio* $\sqrt{p}\varepsilon$*. Inoltre se* $\sqrt{p}\varepsilon < 1$*, allora* λ_{max} *e* λ_{min} *soddisfano*

$$\left|\frac{\lambda_{max}}{\lambda_{min}}\right| \leq \frac{1+\sqrt{p}\varepsilon}{1-\sqrt{p}\varepsilon}$$

Teorema 2.1.5. *I valori singolari di* AM *sono concentrati in* 1 *e ricadono nell'intervallo* $[1-\delta, 1+\delta]$*, con* $\delta = \sqrt{n}\varepsilon(2+\sqrt{n}\varepsilon)$*. Inoltre, se* $\delta < 1$*, allora il numero di condizionamento di* AM *soddisfa*

$$\kappa_2(AM) \leq \sqrt{\frac{1+\delta}{1-\delta}}$$

Le dimostrazioni dei precedenti teoremi si trovano tutte in [10]. In un articolo più recente Wang, Lawlor e Kale [18], dimostrano una serie di risultati riguardante la nonsingolarità di M. Qui riassumiamo i risultati in un unica formulazione

Teorema 2.1.6. *Sia* M *approssimazione di* A^{-1}*, con* $A \in \mathbb{R}^{n\times n}$*. Definiamo con* $C \coloneqq AM - I$*, e consideriamo che* $SP = \{M\}$ *contenga la diagonale principale. Se*

$$\sum_{k=1}^{n} |c_{kk}| < 1$$

allora

$$\|AM - I\|_F < 1\,,$$

e M *è non singolare.*

Dimostrazione. Consideriamo la colonna $\hat{m}_k$ come soluzione del problema ridotto

$$\min_{\hat{m}_k} \|\hat{A}\hat{m}_k - \hat{e}_k\|_2^2$$

che è anche soluzione dell'equazione normale

$$\hat{A}^T\hat{A}\hat{m}_k = \hat{A}^T\hat{e}_k \tag{2.33}$$

Considerando quindi $D \coloneqq AM$, abbiamo che

$$d_{ij} = \sum_{h=1}^{n} a_{ih} m_{hj} = e_i^T A M e_j = e_i^T A m_j$$

ottenendo che

$$d_{kk} = \sum_{h=1}^{n} a_{kh} m_{hk} = \hat{m}_k^T \hat{A}^T \hat{e}_k$$

considerando che se $h \notin \mathcal{J} \implies m_{hk} = 0$, con $\mathcal{J}$ sparsity pattern di m_k, e moltiplicando a sinistra 2.33 per $\hat{m}_k^T$ si ottiene

$$\|d_k\|_2^2 = \hat{m}_k^T \hat{A}^T \hat{A} \hat{m}_k = \hat{m}_k^T \hat{A}^T \hat{e}_k = d_{kk}$$

con infine

$$d_{kk} = \|d_k\|_2^2 = \sum_{j=1}^{n} d_{jk}^2 \tag{2.34}$$

Quindi per costruzione si ha che

$$d_{jk} = \begin{cases} c_{jk} & \text{se} \quad j \neq k \\ c_{jk} + 1 & \text{se} \quad j = k \end{cases} \tag{2.35}$$

e per 2.35 e 2.34, si ha che

$$\begin{aligned} c_{kk} = \quad d_{kk} - 1 \quad &= \quad \sum_{j=1}^{n} d_{jk}^2 - 1 \quad = \quad \sum_{j=1, j\neq k}^{n} d_{jk}^2 + d_{kk}^2 - 1 = \\ = \sum_{j=1, j\neq k}^{n} c_{jk}^2 + (c_{kk}+1)^2 - 1 &= \sum_{j=1, j\neq k}^{n} c_{jk}^2 + c_{kk}^2 + 2c_{kk} - 1 + 1 = \\ &= \sum_{j=1}^{n} c_{jk}^2 + 2c_{kk} \end{aligned} \tag{2.36}$$

ricavando

$$c_{kk} = -\sum_{j=1}^{n} c_{jk}^2 \tag{2.37}$$

data l'ipotesi $\sum_{k=1}^{n} |c_{kk}| < 1$ si ottiene

$$\left| \sum_{k=1}^{n} c_{kk} \right| = \left| -\sum_{k=1}^{n} c_{kk} \right| = \left| \sum_{k=1}^{n} \sum_{j=1}^{n} c_{jk}^2 \right| = \|C\|_F^2 = \|AM - I\|_F^2 < 1 \tag{2.38}$$

quindi per 2.1.2 AM è non singolare e di conseguenza M è non singolare. □

Osservazione 1. *Possiamo verificare* $\sum_{k=1}^{n} |c_{kk}| < 1$ *calcolando*

$$\sum_{k=1}^{n} |c_{kk}| = \sum_{k=1}^{n} |d_{kk} - 1| = \sum_{k=1}^{n} |a_k m_k - 1|$$

con a_k *riga k-esima di A. Data la sparsità di A ed M, l'operazione consiste in n prodotti sparsi, con un costo computazionale inferiore rispetto alla moltiplicazione matriciale AM.*

Altre proprietà teoriche più generali riguardanti il pattern A^{-1}, risultano necessarie per "catturare" a priori lo sparsity pattern iniziale da assegnare ad M. Per questo motivo nella successiva sezione, si mostreranno alcuni risultati riguardanti il pattern di A^{-1}.

2.1.5 Sparsity pattern input

Per il pattern iniziale di M si possono effettuare delle scelte più accurate rispetto alla semplice diagonale. Infatti grazia ad una euristica a priori è possibile individuare quale sarà il pattern di A^{-1}, per via delle potenze di A. Grazie infatti al teorema di Cayley-Hamilton si ha che

Teorema 2.1.7. ***(Cayley-Hamilton)*** *Sia A matrice* $n \times n$*, e sia* $p(x) = det(A - xI)$ *il polinomio caratteristico di A. Allora*

$$p(A) = 0$$

□

Da cui si ricava che

Teorema 2.1.8. *Sia A una matrice* $n \times n$ *, invertibile, allora* A^{-1} *è combinazione lineare di* $I, A, A^2, ..., A^{n-1}$*.*

Dimostrazione. Consideriamo il polinomio caratteristico $p(x)$

$$p(x) = (-1)^n x^n + c_{n-1} x^{n-1} + ... + det(A)$$

applicando Cayley-Hamilton 2.1.7,

$$p(A) = (-1)^n A^n + c_{n-1} A^{n-1} + ... + det(A) \cdot I = 0$$

e raccogliendo A si ottiene

$$(-1)^n A^{n-1} + c_{n-1} A^{n-2} + ... = -det(A) \cdot A^{-1}$$

□

In modo più compatto quindi possiamo scrivere che

$$A^{-1} = -\frac{1}{c_0}\sum_{i=0}^{n-1} c_{i+1}A^i$$

ottenendo per lo sparsity pattern che

$$SP\{A^{-1}\} \subseteq SP\{(I + abs(A))^{n-1}\} \quad \text{con} \quad abs(A)_{ij} = |a_{ij}| \quad i,j = 1,...,n \tag{2.39}$$

Similarmente si può utilizzare il teorema di Cayley-Hamilton per A^TA, con risultato

$$(A^TA)^{-1} = -\frac{1}{b_0}\sum_{i=0}^{n-1} b_{i+1}(A^TA)^i \implies A^{-1} = -\frac{1}{b_0}\sum_{i=0}^{n-1} b_{i+1}(A^TA)^iA^T$$

e dato che A^TA è SPD e sapendo che per una matrice SPD si ha che gli elementi della diagonale sono $a_{ii} > 0$, si ricava che

$$SP\{A^{-1}\} \subseteq SP\{\Big(abs(A)^Tabs(A)\Big)^{n-1} abs(A)^T\} \tag{2.40}$$

L'idea quindi risiede nel troncare la potenza ad un certo $q \ll n-1$ tale che non si raggiunga un pattern troppo denso. Alcuni esempi numerici hanno mostrato che già con un $q \leq 3$ si riescono ad ottenere pattern soddisfacenti [11]. L'azione di questi pattern possono essere visti in una duplice maniera. Infatti si possono utilizzare come pattern iniziali con un eventuale aumento dinamico degli indici da aggiungere al pattern di M, oppure possono essere utilizzati come pattern limite per M, in modo che i nuovi indici da testare e potenzialmente aggiungere siano esclusivamente quelli che appartengono già al pattern limite per M. Per velocizzare quindi l'algoritmo, le tecniche di selezione del pattern a priori ricoprono un ruolo di rilievo per l'algoritmo SPAI. Vedremo inoltre nella seguente sezione come è possibile trattare il caso particolare delle matrici SPD, e di una generazione del pattern a priori per questa categoria di matrici.

2.2 Matrici SPD: il metodo FSAI

Il metodo FSAI, inizialmente introdotto in [14], acronimo di *factorized sparse approximate inverse*, risulta un metodo efficiente per le matrici simmetriche definite positive (SPD). Esso infatti permette di trovare un precondizionatore che è anch'esso SPD. L'obiettivo è trovare una matrice G tale che

$$G^TG \approx A^{-1} \tag{2.41}$$

dove considerando la decomposizione di Cholesky $A = LL^T$, con L fattore di Cholesky triangolare inferiore, si ottiene G come minimizzazione della norma

$$\|I - GL\|_F^2 = tr\left[(I - GL)^T(I - GL)\right] \tag{2.42}$$

per uno sparsity pattern fissato $\mathcal{S}_L$ di G. Il vantaggio di questo algoritmo risiede nel fatto che non è necessario il calcolo esplicito di L per il calcolo di G. Infatti

$$tr\left[(I - GL)^T(I - GL)\right] = tr\left[I^TI - 2(GL)^TI + (GL)^T(GL)\right] =$$

$$= tr\left[I^TI\right] - 2\,tr\left[(GL)^TI\right] + tr\left[(GL)^T(GL)\right] \tag{2.43}$$

differenziando 2.43 rispetto agli elementi g_{ij} di G

$$\frac{\partial}{\partial g_{ij}}(-2\,tr\left[(GL)^T\right]) = -2L_{ji}\,,$$

$$\frac{\partial}{\partial g_{ij}}(tr\left[L^TG^TGL\right]) = 2(GLL^T)_{ij}$$

ricavando che

$$\frac{\partial}{\partial g_{ij}}\|I - GL\|_F^2 = 2(GLL^T)_{ij} - 2L_{ji}$$

e uguagliando a zero si ottiene

$$(GLL^T)_{ij} = (GA)_{ij} = (L^T)_{ij} \quad \forall (i,j) \in \mathcal{S}_L \tag{2.44}$$

e dato che L^T è una matrice triangolare superiore, mentre $\mathcal{S}_L$ è lo sparsity pattern triangolare inferiore, si ricava che

$$(GA)_{ij} = \begin{cases} 0 & \text{se} \quad i \neq j\,, \quad (i,j) \in \mathcal{S}_L \\ l_{ij} & \text{se} \quad i = j \end{cases} \tag{2.45}$$

ma G non è calcolabile poiché L non è stata calcolata. Così si procede invece a calcolare $\tilde{G}$ tale che

$$(\tilde{G}A)_{ij} = \delta_{ij} \quad \forall\,(i,j) \in \mathcal{S}_L, \tag{2.46}$$

dove δ_{ij} è la delta di Kronecker. La matrice finale G quindi è ottenuta mediante l'operazione

$$G = D\tilde{G} \tag{2.47}$$

dove D è la matrice di riscalatura diagonale. Una scelta opportuna per D è

$$D = \left[diag(\tilde{G})\right]^{-1/2}, \tag{2.48}$$

tale che

$$(GAG^T)_{ii} = 1, \qquad 1 \leq i \leq n \tag{2.49}$$

Il seguente teorema mostra come il precondizionatore FSAI esiste per ogni pattern $\mathcal{S}_L$ che include la diagonale principale di A.

Teorema 2.2.1. *Si supponga A SPD. Se lo sparsity pattern triangolare inferiore $\mathcal{S}_L$ contiene tutti gli indici della diagonale principale, allora G esiste ed è unica.*

Dimostrazione. Poniamo $\mathcal{P}_i = \{j : (i,j) \in \mathcal{S}_L\}$, e sia $A[\mathcal{P}_i, \mathcal{P}_i]$ la sottomatrice di ordine $nz_i = |\mathcal{P}_i|$ con elementi a_{kl} tale che $k, l \in \mathcal{P}_i$. Poniamo quindi $\tilde{g}_i$ e g_i vettori densi contenenti gli elementi diversi da zero della riga i-esima di $\tilde{G}$ e G rispettivamente. Allora risolvere la 2.46 equivale a risolvere n sistemi lineari SPD indipendenti

$$A[\mathcal{P}_i, \mathcal{P}_i]\tilde{g}_i = e_{nz_i}, \qquad 1 \leq i \leq n, \tag{2.50}$$

e inoltre

$$(\tilde{G}A\tilde{G}^T)_{ii} = \sum_{j \in \mathcal{P}_i} \delta_{ij}\tilde{G}_{ij} = \tilde{G}_{ii} = (A[\mathcal{P}_i, \mathcal{P}_i]^{-1})_{ii} \tag{2.51}$$

Il quale implica che gli elementi diagonali di D date da 2.48 sono diverse da zero. Di conseguenza le righe di G calcolate esistono e la soluzione risulta unica. □

Il calcolo quindi per il precondizionatore FSAI è intrinsecamente parallelo per righe, e ed è riassunto nell'algoritmo 3. Come per l'algoritmo 2 (SPAI), anche per la procedura FSAI si effettua una euristica sullo sparsity pattern. Inoltre il seguente teorema mostra come il precondizionatore calcolato con 3 risulta in senso lato ottimale.

Algorithm 3: precondizionatore FSAI

Input: A: matrice SPD, $\mathcal{S}_L$: sparsity pattern triangolare inferiore

Output: G: matrice triangolare inferiore tale che $G^TG \approx A^{-1}$

1 **for** $i = 1 : n$ **do**

2 Determinare $\mathcal{P}_i = \{j : (i,j) \in \mathcal{S}_L\}$, $A[\mathcal{P}_i, \mathcal{P}_i]$ e $nz_i = |\mathcal{P}_i|$

3 Risolvere $A[\mathcal{P}_i, \mathcal{P}_i]\,\tilde{g}_i = e_{nz_i}$

4 Riscalare $g_i = d_{ii}\tilde{g}_i$ con $d_{ii} = (\tilde{g}_{i,nz_i})^{-1/2}$, con $\tilde{g}_{i,nz_i}$ ultimo elemento di $\tilde{g}_i$

5 Assegnare g_i a $G(i, 1:i)$ secondo il pattern $\mathcal{S}_L$

Teorema 2.2.2. *Sia L il fattore di Cholesky di una matrice A SPD. Sia inoltre $\mathcal{S}_L$ lo sparsity pattern triangolare inferiore che include tutti gli indici della diagonale principale, e G il precondizionatore calcolato mediante FSAI. Allora qualsiasi*

matrice triangolare inferiore G_1 *con* $SP\{G_1\} \subseteq \mathcal{S}_L$ *e* $(G_1 A G_1^T)_{ii} = 1\,(1 \leq i \leq n)$ *soddisfa*

$$\|I - GL\|_F \leq \|I - G_1 L\|_F$$

□

Inoltre, riguardo il pattern S_L si ha che

Teorema 2.2.3. *Sia* L *il fattore di Cholesky di una matrice* A *SPD. Siano inoltre* $\mathcal{S}_{L_1}, \mathcal{S}_{L_2}$ *pattern triangolari inferiori che includono tutti gli indici della diagonale principale. Allora se* $\mathcal{S}_{L_1} \subseteq \mathcal{S}_{L_2}$*, si ha che*

$$\|I - G_2 L\|_F \leq \|I - G_1 L\|_F$$

□

2.2.1 Calcolo del pattern

Come per il metodo SPAI, ad un aumento del numero di elementi nello sparsity pattern si richiede un aumento del costo computazionale. Per ovviare a questa situazione si cerca a priori di individuare un pattern per l'approssimazione di L^{-1}. Uno dei metodi più riconosciuti per il precondizionatore FSAI implementato in [13] e [5], è utilizzare una versione filtrata della matrice A iniziale, chiamata $\tilde{A}$, eliminando gli elementi fuori diagonale di A soddisfacenti

$$|a_{ij}| \leq \tau \sqrt{a_{ii} a_{jj}} \tag{2.52}$$

per un parametro $\tau \in [0,1]$ fissato. Interessati però allo sparsity pattern di L^{-1} triangolare inferiore, possiamo adottare una strategia ricorsiva studiata in [11], di q passi in cui

$$B_{p+1} \leftarrow tril(B_P \tilde{A}) \qquad p = 0,1,...,q-1$$

con $B_0 = I$, e $tril(\cdot)$ indica la parte triangolare inferiore. Questa euristica iniziale ci permette quindi, come analogamente studiato nel caso del precondizionatore SPAI, di ricercare uno sparsity pattern iniziale che migliori l'approssimazione. Definire il valore ottimale di τ è fortemente dipendente dal problema in considerazione, ed è una procedura complicata se gli elementi di A cadono in un intervallo ristretto. Ma comunque la procedura di pre-filtraggio 2.52 adottata, considerando che A è una matrice sparsa ($nz(A) \ll n^2$, $nz(A) \approx O(n)$), risulta avere complessità computazionale lineare. Utilizzare quindi ripetutamente questa procedura per la costruzione di $\tilde{A}$ non comporta un impatto significativo in termini di costo computazionale sul calcolo del precondizionatore tramite FSAI. L'algoritmo 4 riassume la procedura generale per il calcolo di $\tilde{A}$, e ci permette di ottenere una pattern

Algorithm 4: FSAI input pattern

Input: A: matrice SPD, τ: parametro

Output: $\mathcal{S}_L$: pattern triangolare inferiore

1 Calcolare $\tilde{A}$ eliminando gli elementi tali che $a_{ij} \leq \tau\sqrt{a_{ii}a_{jj}}$

2 $B_0 = I$

3 **for** $p = 0 : q - 1$ **do**

4 $\quad B_{p+1} \leftarrow tril(B_p\tilde{A})$

5 $\mathcal{S}_L = SP\{tril(B_q)\}$

iniziale tenendo conto delle potenze anche superiori a $q = 3$ come invece considerate nel caso del metodo SPAI, ottenendo comunque una strutta del pattern di G sparsa.

2.2.2 FSAI in letteratura

Con l'acronimo FSAI si indica l'algoritmo presentato in [14], ma esso non è l'unico algoritmo che utilizza una fattorizzazione di A per approssimare l'inversa. I metodi che utilizzano alla base una procedura di tipo *sparse approximate inverse* con fattorizzazione sono diversi, e un altro metodo dei più conosciuti, oltre al già citato FSAI, è il metodo AINV, sviluppato in [3, 2]. Il metodo AINV (Approximate Inverse) è una tecnica numerica per costruire un'approssimazione dell'inversa di una matrice sparsa, simmetrica e definita positiva $A \in \mathbb{R}^{n\times n}$. L'obiettivo è ottenere una fattorizzazione approssimata della forma

$$A^{-1} \approx M \approx ZD^{-1}Z^T,$$

dove Z è una matrice triangolare superiore ed è l'approssimazione dell'inversa del fattore L^T nella decomposizione LDL^T di A, mentre $D \in \mathbb{R}^{n\times n}$ è una matrice diagonale. Il metodo AINV calcola direttamente Z e D attraverso un processo di coniugazione incompleta (A-*orthogonalization*) applicato ai vettori unitari di base. Questo metodo non richiede di imporre uno sparsity pattern iniziale, e la sparsità del precondizionatore M è dovuta a tecniche di "sparsificazione" del fattore Z, tramite filtraggio degli elementi. In generale il metodo AINV è anche applicabile a matrici sparse generiche, come mostrato in [2], anche se questo è stato provato anche per il metodo FSAI, vedi [19]. Il problema del metodo AINV risiede nella instabilità dell'algoritmo, e in una complessità maggiore rispetto al metodo FSAI nella parallelizzazione. Tale metodo è risultato robusto soltanto per alcuni tipi

di matrici speciali e si osserva inoltre che l'algoritmo è soggetto a breakdown, un problema che ha reso necessario l'adattamento del metodo, portando allo sviluppo di una nuova versione, denominata SAINV o *Stabilized*-AINV, vedi [4]. Inoltre Benzi, uno degli autori del metodo AINV, in [1] riporta che il metodo FSAI è un metodo efficace per matrici simmetriche definite positive generiche e in problemi provenienti dalla meccanica strutturale [4, 15], inoltre non soffre di problemi legati al breakdown come il metodo AINV. La mancanza di un approccio dinamico quindi per FSAI, è soppiantata da una euristica sul pattern $\mathcal{S}_L$ di G, come per esempio le strategie di selezione del pattern studiate da Chow in [6]. Analogamente al precondizionatore SPAI già analizzato, la sua variante fattorizzata FSAI è un algoritmo robusto e intrinsecamente parallelo, in grado di generare precondizionatori efficaci. Entrambi i metodi, SPAI e FSAI, costituiscono le fondamenta per gli approcci di tipo *sparse approximate inverse*.

Parte II

Test Numerici: Risultati e Analisi

Capitolo 3

Precondizionamento di matrici

Il principale obiettivo dei metodi SPAI e FSAI è di ottenere un precondizionatore M che riduca il numero di condizionamento della matrice sparsa $A \in \mathbb{R}^{n \times n}$ in input. Tale procedura si ottiene mediante un'approssimazione dell'inversa della matrice A, con $M \approx A^{-1}$. In questa sezione si presenta un'analisi dei risultati numerici ottenuti mediante l'applicazione dei due algoritmi per il precondizionamento di matrice. L'oggetto principale di queste simulazioni è ottenere dati riguardanti l'effetto del precondizionatore M attraverso diverse metriche, tra cui: il numero di condizionamento della matrice precondizionata (e.g. $\kappa_2(AM)$ per SPAI) e la sparsità del precondizionatore, considerando il numero di elementi non nulli $nz(M)$, così da poter analizzare come ciascun algoritmo bilanci la riduzione del numero di condizionamento e il mantenimento di una struttura sparsa nel precondizionatore. Tali proprietà sono essenziali per garantire sia l'efficacia del precondizionamento, sia per non compromettere eccessivamente il costo computazionale e di memoria. Gli esperimenti numerici sono stati eseguiti su un set di matrici sparse, matrici che provengono dalla collezione SuiteSparse [8], collezione di matrici derivanti da diversi ambiti ingegneristici. La caratteristica che accomuna queste matrici è la loro identificazione di matrici sparse, adatte quindi all'applicazione dei metodi FSAI e SPAI.

3.1 Precondizionamento di matrici sparse

Per i due metodi sono stati utilizzati diversi set di matrici, questo poiché il metodo SPAI può essere applicato a matrici generiche, mentre il metodo FSAI nasce come metodo adottato per matrici SPD.

3.1.1 Sintesi delle metriche

Il metodo SPAI come abbiamo esposto nel precedente capitolo, si basa su un problema di minimizzazione in norma di Frobenius

$$\min_{SP\{M\}\in\mathcal{S}} \|AM - I\|_F$$

e saranno quindi raccolti i valori di tale norma, e della norma iniziale $\|A - I\|_F$. Le altre quantità in esame dei test numerici effettuati riguardano il numero di condizionamento $\kappa_2(\cdot)$ e il rapporto fra il numero di elementi non zero $nz(M)$ di M e il numero di elementi non zero $nz(A)$ di A. Ricordando la definizione di valore singolare σ_i di una matrice $A \in \mathbb{R}^{n\times n}$ invertibile

$$\sigma_i = \sqrt{\lambda_i(A^T A)}, \qquad i = 1, ..., n \tag{3.1}$$

e considerando il loro ordinamento

$$\sigma_1 \geq ... \geq \sigma_{n-1} \geq \sigma_n > 0$$

il numero di condizionamento della matrice $\kappa_2(A)$ è definito come

$$\kappa_2(A) = \frac{\sigma_1}{\sigma_n}$$

che rappresenta il rapporto fra il valore singolare maggiore e minore rispettivamente. I dati raccolti quindi comprendono il numero di condizionamento della matrice $\kappa_2(A)$ e il numero di condizionamento $\kappa_2(AM)$. Infine, per quanto riguarda la sparsità del precondizionatore, al fine di considerare il costo di memoria associato a M, verrà introdotta un'altra metrica definita come

$$NZ_{ma} := \frac{nz(M)}{nz(A)} = \frac{\#elementi\ non\ zero\ di\ M}{\#elementi\ non\ zero\ di\ A} \tag{3.2}$$

metrica che consentirà di confrontare la sparsità del precondizionatore rispetto alla matrice originale A.

3.1.2 Simulazioni SPAI

Per ogni simulazione effettuata, il parametro che indica ad ogni iterazione il numero di nuovi indici s da aggiungere allo sparsity pattern $\mathcal{J}$ della colonna $k-$esima di M, ovvero m_k, è fissato ed è pari al valore $s = 3$. Tuttavia, al fine di evitare un eccessivo incremento dello sparsity pattern con indici non significativi, si è scelto di utilizzare un euristica sulle quantità ρ_j^2 in 2.23, poiché l'indice j viene selezionato

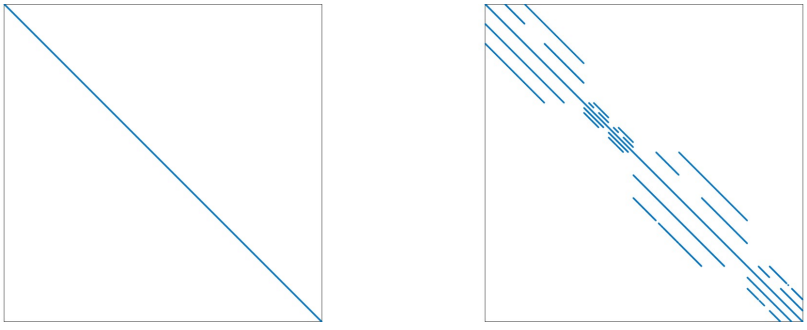
soltanto se

$$\rho_j^2 \leq \frac{1}{|\tilde{\mathcal{J}}|} \sum_{i=1}^{|\tilde{\mathcal{J}}|} \rho_i^2 \tag{3.3}$$

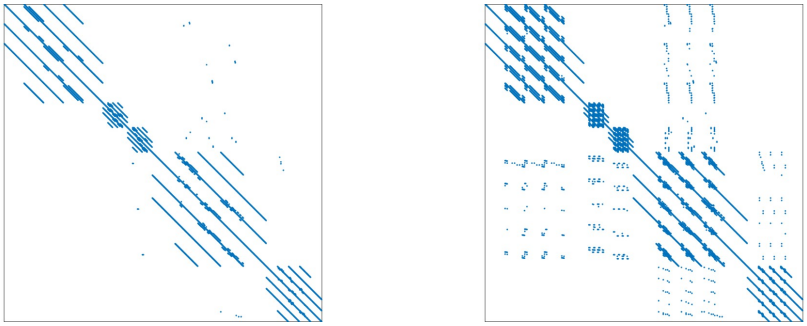
Questo ci permette di evitare di aggiungere ulteriori indici qualora non apportino un contributo significativo, rispetto agli altri indici, all'approssimazione. I restanti parametri riguardano i criteri d'arresto e lo sparsity pattern iniziale. I criteri d'arresto riguardano il numero di iterazioni $iter$, la tolleranza sul residuo ϵ, e il numero massimo di indici da aggiungere per colonna N_{max}. Per lo sparsity pattern in input, considerate equazioni 2.39, 2.40, si è scelto di utilizzare tre diversi pattern iniziali, considerando $\{I, I + abs(A), I + abs(A) + abs(A^T)\}$. Coerentemente con le simulazioni effettuate in [9], sono stati utilizzati valori di $\epsilon \in$ [0.5,0.2]. In prima analisi, si osservano i risultati ottenuti per delle matrici aventi pattern simmetrico, ovvero con $SP\{A\} = SP\{A^T\}$. La tabella 3.1 riporta i risultati del calcolo SPAI con parametri $(N_{max} = 35, iter = 20)$ per il pattern iniziale $\{I\}$, e $(N_{max} = 25, iter = 20)$ per il pattern iniziale $\{I + abs(A)\}$. Analizzando i dati si nota che in tutti i casi vi è effettivamente una riduzione della norma $\|AM - I||_F$, ottenendo nel secondo caso che tutte le norme sono $\|AM - I||_F < 10$. Riguardo al numero di condizionamento $\kappa_2(A)$, si è ottenuta una riduzione non inferiore ai 2 ordini di grandezza del numero di condizionamento in $\kappa_2(AM)$. Inoltre è interessante notare come in certi casi la riduzione del numero di condizionamento sia stata ottenuta con valori esigui di NZ_{max}. Per la matrice `rajat04` per esempio, la riduzione del numero di condizionamento da $\kappa_2(A) \approx 10^8$ a $\kappa_2(AM) \approx 10^4$ è stata ottenuta per $\epsilon = 0.5$ con un $NZ_{max} = 0.22$. Questo significa che la matrice M ha costo di memoria pari a circa 1/5 della memoria necessaria per memorizzare A, permettendo contemporaneamente una riduzione del numero di condizionamento di 4 ordini di grandezza.

Tabella 3.1:
Simulazioni SPAI con matrici a pattern simmetrico.

				$N_{max} = 35, iter = 20$			$N_{max} = 25, iter = 20$		
				I			$I + abs(A)$		
Matrix	ϵ	$\|A - I\|_F$	$\kappa_2(A)$	$\|AM - I\|_F$	$\kappa_2(AM)$	NZ_{ma}	$\|AM - I\|_F$	$\kappa_2(AM)$	NZ_{ma}
orsirr_1	0.5	1.847e+06	7.714e+04	1.185e+01	2.018e+02	0.61	9.431e+00	7.774e+01	1.20
-	0.3	-	-	7.478e+00	3.107e+01	1.49	7.963e+00	3.120e+01	1.86
orsirr_2	0.5	1.546e+06	6.327e+04	1.130e+01	2.118e+02	0.64	8.843e+00	7.695e+01	1.21
-	0.3	-	-	7.038e+00	3.106e+01	1.59	7.456e+00	3.119e+01	1.93
sherman1	0.5	6.802e+01	1.560e+04	1.048e+01	3.173e+02	0.82	9.593e+00	2.604e+02	1.09
-	0.3	-	-	7.122e+00	1.196e+02	1.75	7.017e+00	1.022e+02	1.87
rajat04	0.5	3.136e+04	1.640e+08	1.007e+01	9.223e+04	0.22	5.686e+00	2.454e+05	1.03
-	0.3	-	-	6.819e+00	5.167e+04	0.44	4.941e+00	2.373e+05	1.07



$M \quad iter = 1$ $M \quad iter = 2$



$M \quad iter = 3$ $M \quad iter = 10$

Figura 3.1: Sparsity pattern di M per iterazione, matrice `orsirr_1`

Altre simulazioni sono state effettuate per matrici generiche. Nella tabella successiva 3.2, sono riportati i risultati ottenuti dalle simulazioni con diversi pattern iniziali. La sigla "n.d." riportata indica che la matrice M restituita dall'algoritmo è singolare. Osservando i dati, e la differenza fra i tre diversi sparsity pattern in input, si nota come l'informazione riguardante lo sparsity pattern della matrice $SP\{abs(A)\}$, porti effettivamente ad una soluzione migliore per il precondizionamento della matrice. Nei casi limite, ovvero per le matrici `fs_541_2`,`fs_541_3` e `fpga_dcop_28`,`fpga_dcop_45`, l'informazione $SP\{abs(A)\}$ risulta rilevante nella costruzione del precondizionatore. Inizialmente tali matrici presentano un numero di condizionamento molto elevato. Ad esempio, la matrice `fs_541_2` ha un $\kappa_2(A) \approx 4.4 \times 10^{10}$, e la matrice `fpga_dcop_28` ha $\kappa_2(A) \approx 4.2 \times 10^{12}$. Tuttavia, dopo l'applicazione del precondizionamento SPAI con anche $SP\{abs(A)\}$, i valori del condizionamento per tutte queste matrici si avvicinano a $\kappa_2(AM) \approx 1$. Per quanto riguarda invece $SP\{abs(A^T)\}$, si nota dai dati riguardante la matrice `shl_0`, che tale informazione permette di ottenere un condizionamento e un NZ_{ma} inferiore rispetto agli altri due input, valido per entrambi i valori di $\epsilon \in \{0.3,0.2\}$. Dai dati emerge chiaramente la differenza nella riduzione del numero di condizionamento tra il pattern in input $\{I\}$ e i restanti pattern. Dalle simulazioni si

Tabella 3.2:
Simulazioni SPAI con matrici generiche, $iter = 20$.

			$N_{max} = 35$		$N_{max} = 25$		$N_{max} = 20$	
			I		$I + abs(A)$		$I + abs(A) + abs(A^T)$	
Matrix	ϵ	$\kappa_2(A)$	$\kappa_2(AM)$	NZ_{ma}	$\kappa_2(AM)$	NZ_{ma}	$\kappa_2(AM)$	NZ_{ma}
sherman2	0.3	9.643e+11	n.d.	n.d.	2.922e+06	1.40	6.637e+07	1.54
-	0.2	-	n.d.	n.d.	1.820e+06	1.50	9.363e+05	1.64
shl_0	0.3	8.573e+08	1.072e+06	6.61	n.d.	n.d.	1.251e+04	5.45
-	0.2	-	5.348e+04	10.25	2.438e+05	8.27	5.087e+03	7.32
pores_2	0.3	1.094e+08	8.513e+04	2.20	4.609e+04	2.47	5.059e+04	2.57
-	0.2	-	8.690e+04	2.42	4.489e+04	2.62	5.008e+04	2.63
fs_541_2	0.3	4.411e+10	2.067e+08	0.67	1.738e+00	1.11	1.556e+00	1.34
-	0.2	-	2.021e+08	0.75	1.409e+00	1.15	1.386e+00	1.35
fs_541_3	0.3	2.826e+11	2.083e+10	1.22	2.565e+00	1.31	2.569e+00	1.48
-	0.2	-	1.942e+10	1.37	1.974e+00	1.45	2.026e+00	1.60
fpga_dcop_28	0.3	4.223e+12	9.442e+11	1.44	5.075e+00	1.93	5.075e+00	2.07
-	0.2	-	9.356e+11	1.50	4.089e+00	2.04	4.089e+00	2.18
fpga_dcop_45	0.2	3.099e+12	6.635e+11	1.43	4.239e+00	2.00	4.239e+00	2.14
-	0.2	-	6.930e+11	1.37	5.250e+00	1.89	5.250e+00	2.03

ricava che l'informazione sul pattern della matrice A è risultata utile per ottenere una riduzione significativa del numero di condizionamento. Considerando inoltre la matrice `sherman2`, essa mostra inizialmente un numero di condizionamento estremamente elevato, con un valore di $\kappa_2(A) \approx 9.6 \times 10^{11}$. Applicando il metodo SPAI, si osserva una netta riduzione del numero di condizionamento nella matrice precondizionata. Per il caso con $\epsilon = 0.3$ e input $\{ I + abs(A) \}$,per la matrice precondizionata si ha che $\kappa_2(AM)$ si riduce di 5 ordini di grandezza, raggiungendo un valore di $\kappa_2(AM) \approx 2.9 \times 10^6$. Per cercare di abbassare ulteriormente il condizionamento di `sherman2`, poiché si è ritenuto utile esplorare l'efficacia del metodo SPAI, è stato effettuato un ulteriore esperimento. In tale prova è stata impiegata una variante del metodo SPAI orientata per righe, ed è stata applicata alle matrici `sherman2` e `pores_2`. I risultati ottenuti da questi esperimenti sono di particolare interesse e saranno analizzati in dettaglio dopo una breve introduzione al metodo SPAI orientato per righe.

Metodo orientato per righe

Considerando il metodo SPAI, abbiamo trattato il metodo come orientato per colonne. Ovvero si cerca di approssimare la colonna $k-$esima di A^{-1}. In alcuni casi particolari però, risulta utile approssimare l'inversa orientandosi per righe. Infatti poniamo per esempio un caso limite dove l'inversa reale di A^{-1} è della

forma

$$A^{-1} = \begin{pmatrix} \times & & & & \\ \times & \times & & & \\ \times & & \times & & \\ \times & & & \times & \\ \times & & & & \times \end{pmatrix}$$

orientarsi per colonne significherebbe avere un calcolo non bilanciato, poiché la prima colonna contiene molti più indici delle restanti. Orientarsi per righe invece significherebbe trovare i due elementi non nulli per riga, con un costo bilanciato lungo tutte le righe. L'alternativa quindi consiste nel costruire un'approssimazione

$$\|MA - I\|_F^2 = \sum_{j=1}^{n} \|m_j^T A - e_j^T\|_2^2 \tag{3.4}$$

dove le m_j^T sono le righe di M. Questo è equivalente a ricercare la matrice M^T tale da minimizzare la quantità

$$\|A^T M^T - I\|_F^2 \tag{3.5}$$

Quindi l'algoritmo originale per colonne può essere riadattato per righe sostituendo A e M con le loro trasposte, ottenendo così un *precondizionatore sinistro.*
Applicando SPAI adattato per righe alle matrici `sherman2` e `pores_2`, si nota in effetti come si ottenga un risultato diverso, ottenendo un numero di condizionamento $\kappa_2(MA)$ (con MA poiché precondizionatore sinistro) minore rispetto alla versione per colonne riportata in tabella 3.2 di $\kappa_2(AM)$

Tabella 3.3:
Simulazioni SPAI di `sherman2` e `pores_2` per righe

			$N_{max} = iter = 20$	
			$I + abs(A) + abs(A^T)$	
Matrix	ϵ	$\kappa_2(A)$	$\kappa_2(MA)$	NZ_{ma}
sherman2	0.3	9.643e+11	5.529e+02	1.46
-	0.2	-	2.302e+01	2.51
pores_2	0.3	1.094e+08	1.376e+03	2.05
-	0.2	-	7.927e+02	2.62

I dati ottenuti dalla simulazione riportati in tabella 3.3, mostrano un numero di condizionamento minore e un minor costo di memoria NZ_{ma} per entrambe le matrici. Principalmente la netta diminuzione del numero di condizionamento si ha per la matrice `sherman2`, dove si è ottenuto nel caso con $\epsilon = 0.2$ un condizionamento approssimativamente pari a $\kappa_2(MA) \approx 23$.

3.1.3 Simulazioni FSAI

La procedura FSAI è stata applicata a matrici simmetriche definite positive (SPD). Il pattern in input è stato generato mediante l'algoritmo 4, che introduce naturalmente il parametro q, che rappresenta il numero di iterazioni per la costruzione del pattern, e il parametro τ come soglia per il filtraggio degli elementi. In questo caso il numero di condizionamento è stato introdotto attraverso la fattorizzazione della matrice $M = G^T G$. Infatti data la condizione imposta in 2.49, possiamo identificare il numero di condizionamento $\kappa_2(GAG^T)$, esplicitando la natura fattorizzata della procedura FSAI. I risultati ottenuti delle simulazioni sono riassunti in tabella 3.4. Sono stati utilizzati due differenti valori per $\tau \in \{0.2,0.3\}$ e tre differenti valori di $q \in \{3,5,7\}$ per il calcolo del pattern iniziale. Osservando i risultati per un valore di $\tau = 0.20$ e $\tau = 0.30$, emerge chiaramente che un aumento di τ tende a ridurre il numero di elementi non nulli nel precondizionatore (NZ_{ma}). Questo effetto è visibile su tutte le matrici testate e in tutte le configurazioni di q. La riduzione di NZ_{ma} riflette un risparmio nella memoria necessaria per memorizzare il precondizionatore. Un chiaro esempio è dato dalle due prime righe in tabella 3.4, le quali riportano la simulazione ottenuta per la matrice `bcsstk14`, matrice derivante da problemi di ingegneria strutturale. Tale matrice presenta un numero di condizionamento $\kappa_2(A) \approx 10^{10}$, che risulta ridursi, in tutti e tre i casi con i valori q diversi, a un ordine di grandezza approssimativamente pari a $\kappa_2(GAG^T) \approx 10^2$.

Tabella 3.4:
Simulazioni FSAI per matrici SPD

			$q=3$		$q=5$		$q=7$	
Matrix	τ	$\kappa_2(A)$	$\kappa_2(GAG^T)$	NZ_{ma}	$\kappa_2(GAG^T)$	NZ_{ma}	$\kappa_2(GAG^T)$	NZ_{ma}
bcsstk14	0.20	1.192e+10	2.310e+02	0.74	1.615e+02	2.44	1.232e+02	5.22
-	0.30	-	4.403e+02	0.18	4.033e+02	0.31	3.833e+02	0.45
bcsstk19	0.20	1.340e+11	7.372e+05	1.31	2.366e+05	2.05	8.487e+04	2.78
-	0.30	-	5.735e+06	1.02	6.179e+06	1.50	6.371e+06	1.94
ex10hs	0.20	5.484e+11	3.341e+03	6.45	5.414e+02	12.21	2.350e+02	16.23
-	0.30	-	5.947e+03	4.62	1.495e+03	10.31	3.389e+02	14.71
msc01440	0.20	3.306e+06	1.440e+03	1.34	1.478e+03	2.58	1.482e+03	4.13
-	0.30	-	2.563e+03	0.91	2.564e+03	1.76	2.562e+03	2.70
mhdb416	0.20	3.994e+09	3.448e+00	1.00	3.448e+00	1.44	3.448e+00	1.84
-	0.30	-	3.988e+00	0.41	3.874e+00	0.54	3.867e+00	0.65
nasa1824	0.20	1.896e+06	1.145e+03	1.90	7.282e+02	4.75	4.482e+02	8.95
-	0.30	-	3.372e+03	0.48	2.990e+03	0.73	2.803e+03	0.90
nos5	0.20	1.100e+04	1.809e+02	0.25	1.762e+02	0.27	1.762e+02	0.27
-	0.30	-	1.863e+02	0.21	1.807e+02	0.23	1.807e+02	0.23

Inoltre per il rapporto NZ_{ma}, sia con i valori $(q = 3, \tau = 0.2)$, ma principalmente nel caso con $\tau = 0.3$, si sono ottenuti valori tutti $NZ_{ma} < 1$, con caso limite $NZ_{ma} = 0.18$ per la coppia di valori $(q = 3, \tau = 0.3)$. Ciò ci permette di avere un precondizionatore ottimale, che produce una riduzione del numero di condizionamento di 8 ordini di grandezza da $10^{10} \rightarrow 10^2$ con un costo di memoria pari a meno di $1/5$ del costo di memoria per A. In generale in tutti i casi si è riscontrato una riduzione del numero di condizionamento. Nel caso della matrice `bcsstk19` inoltre il valore inferiore di $\tau = 0.20$ si accompagna ad una diminuzione nel numero di condizionamento della matrice precondizionata $\kappa_2(GAG^T)$ anche per valori più bassi di q, mentre per $\tau = 0.30$ rimane quasi invariata all'aumentare del parametro q. Un caso da evidenziare è il caso della matrice `mhdb416`, che presenta un comportamento particolarmente interessante. Partendo da un numero di condizionamento iniziale estremamente elevato ($\kappa_2(A) \approx 3.9 \times 10^9$), l'applicazione del precondizionatore FSAI riduce drasticamente il numero di condizionamento della matrice precondizionata $\kappa_2(GAG^T)$ a valori prossimi a 1 per tutte le configurazioni di q. Ad esempio, per $\tau = 0.20$ e $q = 3$, $\kappa_2(GAG^T) \approx 3$, che rimane praticamente invariato all'aumentare di q. Questo suggerisce che, per questa matrice, il precondizionatore è già altamente efficace anche con valori bassi di q. Per $\tau = 0.30$, il comportamento è analogo, ma si osserva una ulteriore diminuzione di NZ_{ma}. Un ulteriore approfondimento è stato effettuato per la matrice `nos5`. Essa presenta un numero di condizionamento iniziale pari a $\kappa_2(A) \approx 1.1 \times 10^4$, che si riduce di 2 ordini di grandezza con l'applicazione del precondizionatore FSAI. Ad esempio, per $\tau = 0.20$ e $q = 3$, il numero di condizionamento della matrice precondizionata risulta essere $\kappa_2(GAG^T) = 1.809 \times 10^2$, con una riduzione a 1.762×10^2 per $q = 7$. Valori simili si riscontrano per $\tau = 0.30$, con $\kappa_2(GAG^T) = 1.863 \times 10^2$ quando $q = 3$. Un aspetto particolarmente significativo per questa matrice è rappresentato dal rapporto NZ_{ma}, che rimane costantemente basso in tutte le configurazioni analizzate. Ad esempio, per $\tau = 0.20$, NZ_{ma} varia appena da 0.25 $(q = 3)$ a 0.27 $(q = 7)$, mentre per $\tau = 0.30$ i valori si attestano rispettivamente a 0.21 e 0.23. Questi valori di NZ_{ma}, estremamente ridotti per entrambi i valori di τ rispetto ad altre matrici, suggeriscono che la soglia τ potrebbe essere stata impostata a un livello troppo alto, limitando la densità del precondizionatore e, di conseguenza, la sua efficacia nel migliorare il numero di condizionamento. Una possibile soluzione potrebbe consistere nel ridurre il valore di τ, al fine di consentire una maggiore densità della matrice precondizionata e valutare l'effetto sul numero di condizionamento $\kappa_2(GAG^T)$. Tale approccio potrebbe rivelarsi utile per aumentare l'efficacia del precondizionatore rispetto alla metrica $\kappa_2(\cdot)$. Per verificare questa ipotesi, sono stati quindi condotti ulteriori esperimenti con una soglia ridotta a $\tau = 0.10$, come riportato in Tabella 3.5. In questo scenario, si osserva una drastica riduzione del numero di condizionamento $\kappa_2(GAG^T)$, che passa a 3.435×10^1 per $q = 3$, con un rapporto NZ_{ma} pari a 2.01. Per $q = 5$, il comportamento rimane analogo,

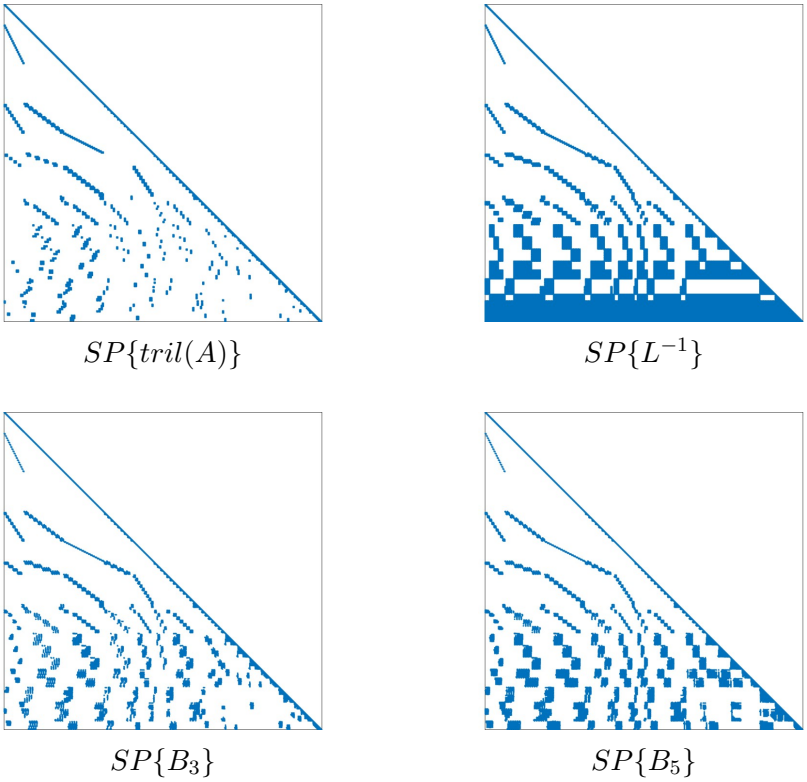


Figura 3.2: Sparsity pattern della parte triangolare inferiore di A, dell'inversa del fattore di Cholesky L, e di G per $q \in \{3{,}5\}$, $\tau = 0.2$, matrice `msc01440`

con un numero di condizionamento $\kappa_2(GAG^T) \approx 3.371 \times 10^1$ e un rapporto NZ_{ma} pressoché invariato ($NZ_{ma} = 2.02$).

Tabella 3.5: Simulazione FSAI per matrice `nos5`

			$k = 3$		$k = 5$	
Matrix	τ	$\kappa_2(A)$	$\kappa_2(GAG^T)$	NZ_{ma}	$\kappa_2(GAG^T)$	NZ_{ma}
nos5	0.10	1.100e+04	3.435e+01	2.01	3.371e+01	2.02

Questi risultati dimostrano chiaramente che una soglia di dropping τ più bassa permette di ottenere un precondizionatore più denso e, di conseguenza, più efficace a discapito di un aumento del costo di memoria. Nel caso della matrice `nos5`, le scelte di $\tau = 0.10$ o $\tau = 0.20$ presentano un compromesso tra efficacia del precondizionatore e costo in termini di memoria. Infatti si evidenzia come per questi due valori il numero di condizionamento e il costo di memoria variano in

modo inverso di un ordine di grandezza

$$\tau = 0.2 \rightarrow \ NZ_{max} \approx 0.2, \ \ \kappa_2(GAG^T) \approx 10^2$$
$$\tau = 0.1 \rightarrow \ NZ_{max} \approx 2, \quad \kappa_2(GAG^T) \approx 10^1$$

Questo suggerisce che una soglia di dropping troppo alta può limitare l'efficacia del precondizionatore e che valori più bassi di τ potrebbero essere preferiti nel caso in cui si voglia minimizzare il numero di condizionamento.

3.2 Analisi spettrale

In questa sezione verrà condotta un'analisi spettrale del precondizionatore M ottenuto attraverso le tecniche precedentemente descritte. L'obiettivo principale è valutare la qualità dell'approssimazione degli autovalori e degli autovettori di M rispetto alla matrice originale. Tale analisi rappresenta un passaggio cruciale, in quanto le proprietà spettrali di M influenzano direttamente l'efficacia del precondizionatore, nonché l'approssimazione di A^{-1} stessa. A tal fine, si confronteranno gli autovalori stimati del precondizionatore con quelli della matrice A, e si esaminerà in che misura gli autovettori associati siano preservati.

3.3 Autovalori, autovettori e diagonalizzazione

Prima di verificare i risultati ottenuti, richiamiamo brevemente le definizioni di autovalore e autovettore di una matrice.

Definizione 3.3.1. *Sia $A \in \mathbb{R}^{n\times n}$. Se $0 \neq v \in \mathbb{C}^n$ e $\lambda \in \mathbb{C}$ soddisfano*

$$Av = \lambda v$$

allora λ e v sono chiamati rispettivamente autovalore e autovettore di A □

Inoltre possiamo anche definire gli autovalori attraverso le radici del polinomio caratteristico di A. Infatti si ha che il sistema $Av = \lambda v$ si può riscrivere nella forma

$$(A - \lambda I)v = 0 \tag{3.6}$$

ed esso ammette soluzioni non nulle se e solo se

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

Si ha quindi che

Teorema 3.3.1. *λ è autovalore di A se e solo se*

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

□

Gli autovalori sono dunque le radici del polinomio caratteristico, mentre gli autovettori sono tutte le soluzioni non nulle del sistema lineare omogeneo 3.6. Si sottolinea anche che un autovettore corrispondente ad un autovalore λ risulta determinato a meno di una costante moltiplicativa $c \neq 0$. Inoltre si può dimostrare che per autovalori distinti $\lambda_1 \neq \lambda_2$, si ha che i rispettivi autovettori saranno linearmente indipendenti. Proprio in quest'ottica si arriva quindi infine a definire la coppia autovalore e autovettore (λ, v), e si introduce la definizione di *autospazio*

Definizione 3.3.2. *Sia λ autovalore di A, si definisce* ***autospazio*** *di λ l'insieme*

$$V_\lambda := \{v \in \mathbb{C}^n \,|\, Av = \lambda v\} = \{v \in \mathbb{C}^n \,|\, (A - \lambda I)v = 0\}$$

cioè formato dagli autovettori corrispondenti a λ e il vettore nullo. □

Inoltre si introducono due quantità fondamentali riferite agli autovalori, e sono la molteplicità algebrica e geometrica di un autovalore. La ***molteplicità algebrica*** di un autovalore, indicata con $m_a(\lambda)$ corrisponde alla molteplicità di λ come radice del polinomio caratteristico. Si definisce invece con ***molteplicità geometrica*** $m_g(\lambda)$ la dimensione dell'autospazio associato a λ. La relazione fondamentale che sussiste fra le due molteplicità è

$$1 \leq m_g(\lambda) \leq m_a(\lambda)$$

cioè che la molteplicità geometrica è sempre inferiore o uguale a quella algebrica. Il perché riguardo l'interesse di queste quantità riguarda la definizione di matrice diagonalizzabile. Infatti si definisce diagonalizzabile una matrice tale che

Definizione 3.3.3. *Una matrice $A \in \mathbb{R}^{n\times n}$ si definisce* ***diagonalizzabile*** *se esiste una matrice non singolare P e una matrice diagonale D tale che*

$$P^{-1}AP = D$$

□

definizione che si collega con la condizione di diagonalizzabilità per A mediante il seguente teorema

Teorema 3.3.2. *Sia $A \in \mathbb{R}^{n\times n}$. A è diagonalizzabile se e solo se la somma delle molteplicità geometriche di tutti gli autovalori di A è uguale a n.* □

L'idea quindi finale è quella di poter scrivere la matrice A attraverso una fattorizzazione, con $A = PDP^{-1}$. La procedura adottata consiste nel

1. Trovare gli autovalori di $A \in \mathbb{R}^{n\times n}$
2. Per ogni autovalore λ trovare una base dell'autospazio associato V_λ
3. A è diagonalizzabile se e solo se la somma delle dimensioni degli autospazi è n. E si ottiene un base di autovettori $v_1, ..., v_n$
4. Infine si considera

$$P = [v_1 \mid ...\mid v_n], \quad P^{-1}AP = \Lambda = \begin{bmatrix} \lambda_1 & 0 & ... & 0 \\ 0 & \lambda_2 & \ddots & 0 \\ \vdots & \ddots & \ddots & 0 \\ 0 & 0 & ... & \lambda_n \end{bmatrix}$$

Un esempio particolare è dato dalle matrici $A \in \mathbb{R}^{n\times n}$ simmetriche, dove si dimostra che tutti gli autovalori sono reali. L'idea è quindi di ottenere la matrice P avente come colonne gli autovettori normalizzati di A, consentendo quindi di avere

$$AP = P\Lambda \implies A = P\Lambda P^{-1}$$

Questo ci permette quindi poter fattorizzare A mediante le matrici P e Λ, con Λ matrice diagonale contente gli autovalori di A. Collegandosi quindi all'approssimazione dell'inversa di una matrice, possiamo intanto notare che

Teorema 3.3.3. *Sia (λ, v) coppia autovalore autovettore di A. Allora $(\frac{1}{\lambda}, v)$ è coppia autovalore autovettore di A^{-1}.*

Dimostrazione.

$$\begin{aligned} Av = \lambda v \implies A^{-1}Av = A^{-1}\lambda v &\implies v = A^{-1}\lambda v \\ &\implies \frac{1}{\lambda}v = A^{-1}v \end{aligned}$$ □

Si può quindi usare la diagonalizzazione di A per ottenere

$$AA^{-1} = \overbrace{P\Lambda P^{-1}}^{A} \underbrace{P\Lambda^{-1}P^{-1}}_{A^{-1}} = I \tag{3.7}$$

con

$$\Lambda^{-1} = \begin{bmatrix} \frac{1}{\lambda_1} & 0 & \dots & 0 \\ 0 & \frac{1}{\lambda_2} & \ddots & 0 \\ \vdots & \ddots & \ddots & 0 \\ 0 & 0 & \dots & \frac{1}{\lambda_n} \end{bmatrix}$$

il che evidenzia come la qualità dell'approssimazione di A^{-1}, e di conseguenza l'efficacia del precondizionamento, dipenda anche dalla capacità di approssimare lo spettro di A. Essendo inoltre gli autovalori una quantità essenziale anche nella definizione di $\kappa_2(\cdot)$, si effettuerà un'analisi dello spettro e degli autovettori del precondizionatore $M \approx A^{-1}$.

Nozioni preliminari e notazione

Le matrici che abbiamo considerato in questi test, sono le matrici `mhdb416` e `nos5`. Entrambe le matrici possono essere usate da entrambi i metodi, sia FSAI che SPAI, potendo quindi infine effettuare anche un confronto fra le due analisi spettrali dei rispettivi metodi. Per facilitare la comprensione delle quantità considerate, introduciamo una notazione per gli autovalori e gli autovettori di A e di A^{-1}. Consideriamo che la matrice test in considerazione sia simmetrica definita positiva (SPD), ovvero che abbia tutti gli autovalori reali $\lambda > 0$. Allora prossiamo introdurre un ordinamento degli autovalori di $A \in \mathbb{R}^{n \times n}$ definito come segue:

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n > 0 \tag{3.8}$$

e di A^{-1}

$$\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n > 0$$

che ovviamente corrispondono a

$$\frac{1}{\lambda_n} \geq \frac{1}{\lambda_{n-1}} \geq \cdots \geq \frac{1}{\lambda_1} > 0 \qquad \mu_i = \frac{1}{\lambda_{(1+n-i)}} \tag{3.9}$$

Per il generico autovettore v_i invece si ha che esso appartiene contemporaneamente alla coppia (λ_i, v_i), coppia autovalore autovettore di A, e a (μ_i, v_i) coppia autovalore autovettore di A^{-1}. Per semplificazione, si indicheranno direttamente gli autovettori e gli autovalori stimati attraverso M (i.e. i reciproci), mediante la notazione con

$$\hat{\lambda}_i \approx \lambda_i\,, \quad \text{e con} \quad \hat{v}_i \approx v_i$$

Riguardo l'approssimazione $\hat{v}_i$ va sottolineato innanzitutto che quanto più paralleli sono gli autovettori $\hat{v}_i$ e v_i, tanto più l'approssimazione è accurata. Per valutare effettivamente l'approssimazione dell'autovettore $\hat{v}_i$ possiamo utilizzare due quantità. La prima quantità che possiamo considerare corrisponde alla proiezione dell'autovettore $\hat{v}_i$ sulla direzione dell'autovettore v_i, ottenendo quindi $(v^T \hat{v}_i)\, v_i$. Partendo quindi da questa proiezione si ricava il vettore $\hat{v} - (v^T \hat{v}_i)\, v_i$, dato dalla differenza fra l'autovettore approssimato e la sua proiezione lungo l'autovettore di riferimento v_i. Più accurata sarà l'approssimazione più la norma $\|\hat{v} - (v^T \hat{v}_i)\, v_i\|_2$ tenderà a zero. Viceversa la norma tenderà al modulo di $\hat{v}_i$, cioè a 1. La prima

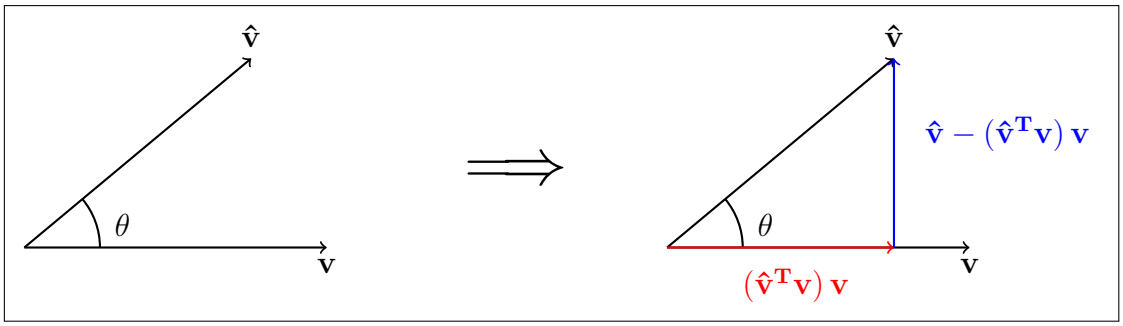


Figura 3.3: Rappresentazione dei vettori $\mathbf{v}$ e $\hat{\mathbf{v}}$, con la proiezione di $\hat{\mathbf{v}}$ su $\mathbf{v}$ in rosso, e il vettore differenza in blu.

quantità è schematizzata in figura 3.3, dove è possibile inturne la costruzione geometrica. Per evitare errori di cancellazione dovuti all'operazione di differenza, si può utilizzare una seconda quantità, considerata anch'essa in questa analisi, e che corrisponde al coseno dell'angolo formato tra i due vettori. Infatti considerando i vettori normalizzati, si avrà che

$$\cos(\theta) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|} = \mathbf{u} \cdot \mathbf{v} \quad \text{poiché} \quad \|\mathbf{u}\| = \|\mathbf{v}\| = 1 \tag{3.10}$$

Per verificare quindi l'approssimazione degli autovettori attraverso M, si considereranno le due quantità

$$\begin{aligned} err_1 &= \Delta v := \|\hat{v} - (\hat{v}^T v)\, v\|_2 \\ err_2 &:= |cos(\theta)| = |\hat{v} \cdot v| = |\hat{v}^T v| \end{aligned} \tag{3.11}$$

Queste due misure, in particolare la misura del coseno, sono state utilizzate per effettuare delle analisi qualitative dei due metodi. Le misure in analisi quindi daranno una qualità dell'approssimazione dell'autovettore, tendendo a due valori

diversi. Infatti si avrà che per la definizione del primo errore Δv, si avrà che

$$\textbf{\textit{"Lower is better"}} \quad \text{considerando} \quad \Delta v \to 0$$

mentre viceversa per la seconda metrica $|cos(\theta)|$ si avrà che

$$\textbf{\textit{"Higher is better"}} \quad \text{considerando} \quad |cos(\theta)| \to 1$$

Indicazioni che ci permetteranno, dati i vari plot effettuati in analisi, di ottenere una visualizzazione grafica della misura dell'approssimazione degli autovettori.

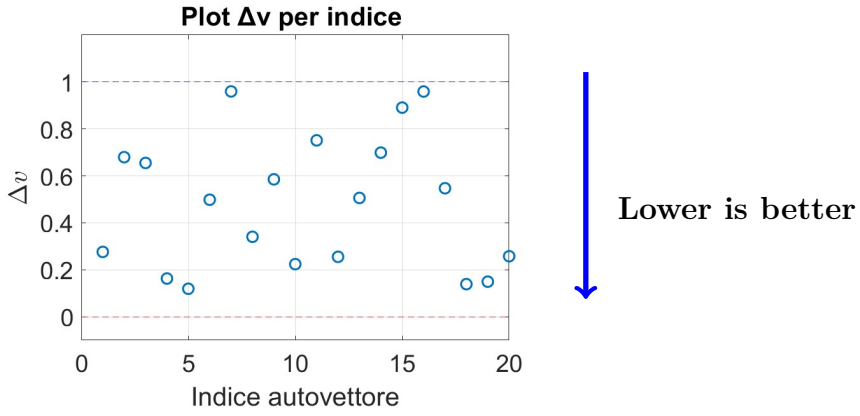


Figura 3.4: Plot di esempio per Δv

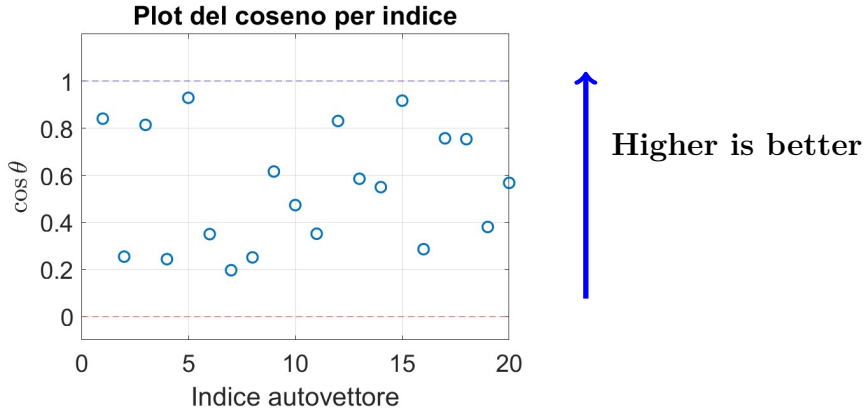


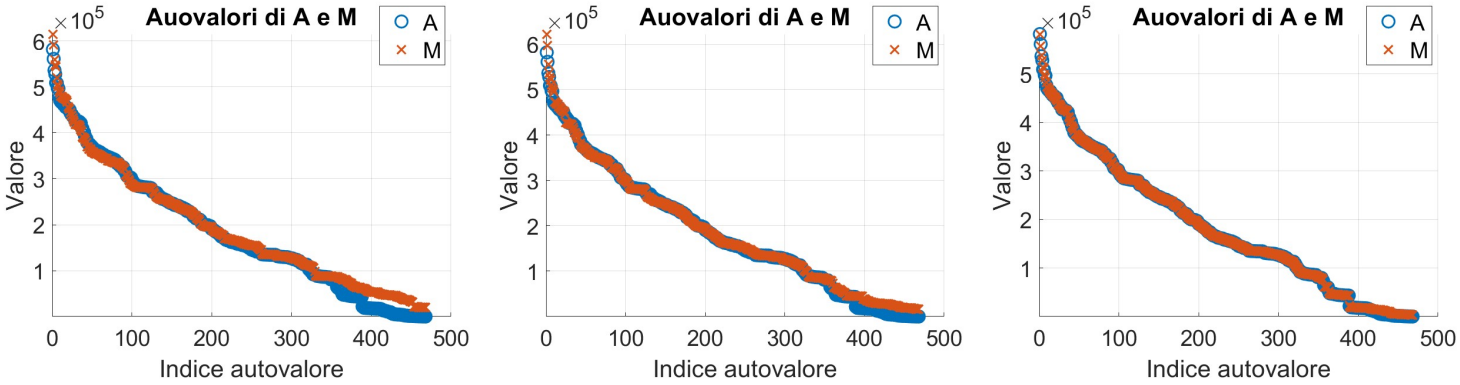
Figura 3.5: Plot di esempio per $\cos(\theta)$

3.3.1 Analisi spettrale per SPAI e FSAI

Le analisi che sono state condotte riguardano l'approssimazione di due matrici, `nos5` e `mhdb416`, entrambe utilizzate per la procedura SPAI e FSAI. Le matrici considerate presentano principali differenze nello spettro, soprattutto riguardo i valori estremali di esso. Infatti la matrice `nos5` è caratterizzata da un autovalore massimo tale che $\lambda_{max} \gg 1$, mentre la matrice `mhdb416` presenta un autovalore minimo con $0 < \lambda_{min} \ll 1$. L'azione del precondizionatore, il cui obiettivo è ridurre il numero di condizionamento, sarà quindi caratterizzato anche dall'approssimazione di questi due valori estremali. Come prima analisi vedremo la matrice `nos5`. Denoteremo per convenzione la matrice in esame tramite A.

Matrice `nos5`: analisi SPAI

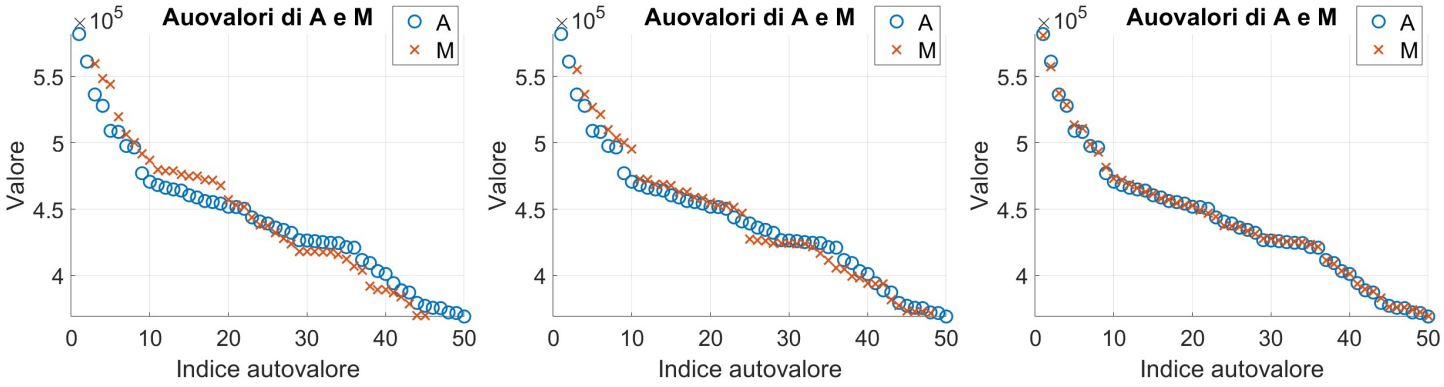
La prima analisi effettuata riguarda la matrice `nos5`. La matrice presenta autovalori massimo e minimo pari a $\lambda_{max} \approx 5.82 \times 10^5$, e $\lambda_{min} \approx 50.28$. La prima analisi effettuata per tale matrice riguarda il metodo SPAI. Volendo valutare la bontà della procedura SPAI nell'approssimare in modo dinamico l'inversa A^{-1}, si è considerato come pattern in input soltanto la matrice identità, ovvero soltanto la diagonale principale. Verranno mostrate nelle figure successive i risultati della simulazione SPAI con tre diversi parametri di N_{max}, ovvero il numero massimo di indici per colonna che è possibile aggiungere al pattern di partenza $SP\{I\}$. In figura 3.6 sono riportate le approssimazioni degli autovalori della matrice A. È evidente, innanzitutto, che all'aumentare del parametro N_{max}, la curva degli autovalori approssimati $\hat{\lambda}$ si avvicina progressivamente all'andamento della curva tracciata dal valore dagli autovalori esatti.



$N_{max} = 3$ $N_{max} = 5$ $N_{max} = 15$

Figura 3.6: SPAI: Approssimazione spettro matrice `nos5`

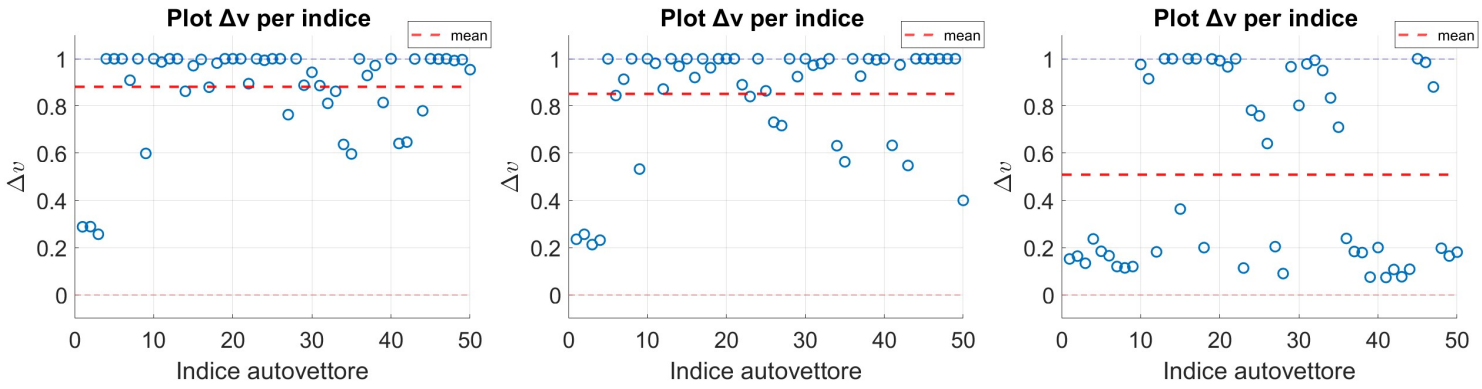
Questo comportamento rispecchia in questo caso l'effetto della procedura dinamica in SPAI, poiché con l'incremento del parametro, sempre più indici vengono aggiunti allo sparsity pattern, da cui deriva un'approssimazione sempre più precisa, con un errore tra gli autovalori approssimati e quelli esatti in diminuzione. Inoltre, osservando attentamente l'evoluzione della curva nei grafici, si nota in particolare che la curva tracciata dagli autovalori approssimati tende a seguire con maggiore accuratezza l'andamento della curva degli autovalori esatti, tanto per i valori massimi quanto per quelli minimi. Tale caratteristica suggerisce quindi che l'approssimazione degli autovalori migliora progressivamente con l'aumento di N_{max}, e che, per valori sufficientemente alti di questo parametro, l'approssimazione diventa molto vicina al valore esatto degli autovalori. Però i vari autovalori differiscono anche di diversi ordini di grandezza, e soprattutto bisogna valutare attentamente anche l'approssimazione degli autovalori estremali maggiori.



$N_{max} = 3$ $N_{max} = 5$ $N_{max} = 15$

Figura 3.7: SPAI: Approssimazione dei primi 50 autovalori maggiori di `nos5` (ordine autovalori decrescente)

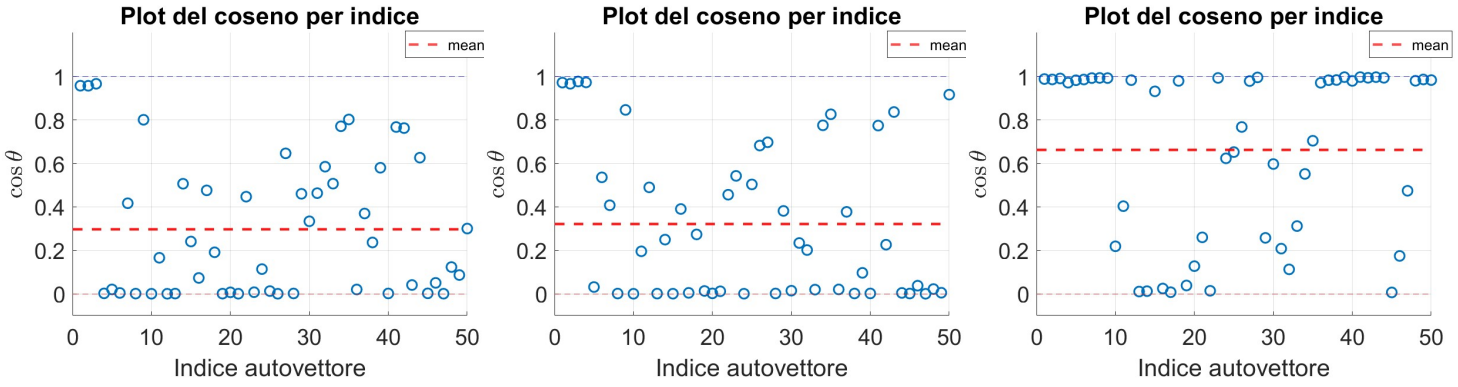
In figura 3.7 vengono mostrate le approssimazioni dei primi 50 autovalori più grandi di A. Quello che si riscontra è effettivamente un fenomeno di convergenza, in cui l'andamento dell'approssimazione tende al valore reale di λ, con il crescere di N_{max}. Connessa all'approssimazione di tali autovalori, si è anche effettuata l'analisi dei corrispondenti autovettori, in modo da riscontrare se vi è un parallelismo fra l'andamento delle approssimazioni delle due quantità. Nella figura successiva 3.8 infatti viene mostrata l'approssimazione degli autovettori v_i associati ai primi 50 autovalori maggiori di A. Ricordando che più $\Delta v \to 0$ più l'approssimazione è accurata, si può notare come anche in questo che per i valori del parametro $N_{max} \in \{5{,}15\}$ la stima dei v_i diventi più accurata in media (linea rossa), anche se in maniere lieve nel caso da $N_{max} = 3$ a $N_{max} = 5$, con un'approssimazione in media che tende ad aumentare. In generale non si riscontra una convergenza



$N_{max} = 3$ $N_{max} = 5$ $N_{max} = 15$

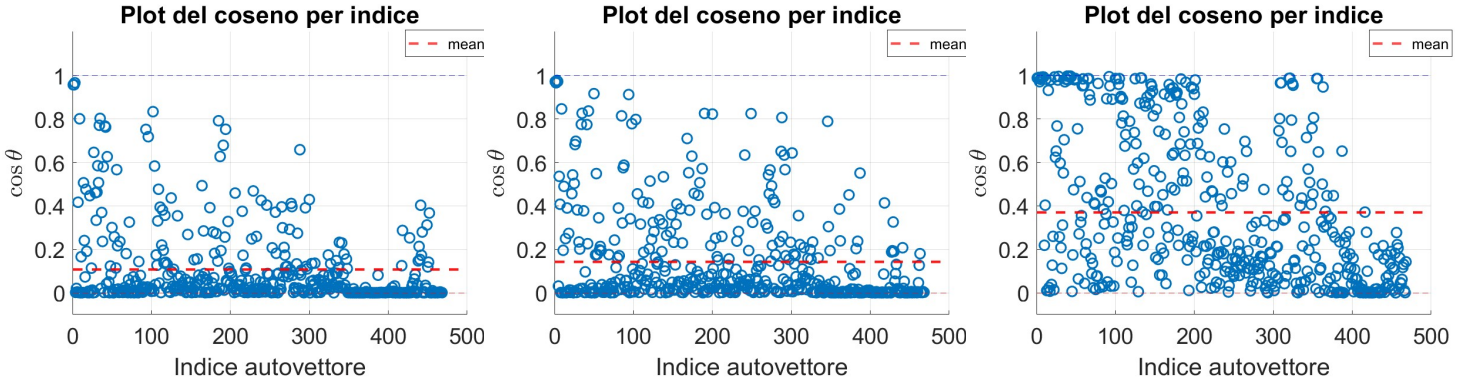
Figura 3.8: SPAI: Approssimazione degli autovettori corrispondenti ai 50 autovalori maggiori di `nos5` (ordine autovalori decrescente)

abbastanza rilevante e qualitativamente simile all'approssimazione degli autovalori. Per evitare che possano insorgere difetti di approssimazione che siano riconducibili ad errori numerici dati dal calcolo diretto della quantità Δv, si è effettuata anche un analisi sul $\cos(\theta)$. Considerando quindi che in questo caso, più $\cos(\theta) \to 1$, più l'approssimazione è migliore, si ottiene comunque un andamento analogo alla figura precedente 3.8, e i risultati sono riportati nella figura 3.9. Ovvero, si nota che vi è una crescita media esigua delle approssimazioni dal caso $N_{max} = 3$ a $N_{max} = 5$, e un'approssimazione media rilevante nell'ultimo caso con $N_{max} = 15$. Inoltre questo riscontro si ha anche nei dati relativi all'intero insieme degli autovettori, i quali mostrano un andamento simile rispetto a quanto osservato nei precedenti grafici. I grafici riportati in 3.10 rivelano chiaramente una tendenza, congruente ai casi in esame precedenti, verso l'approssimazione degli autovettori. Tuttavia, con una visuale sull'intero spettro, l'approssimazione riscontrata degli autovettori sembra favorire gli autovettori associati agli autovalori maggiori, poiché non si riscontra una variazione uniforme nell'approssimazione media, ma tale variazione è più presente nella regione corrispondente agli autovettori associati ai primi 50 autovalori maggiori. Al contrario, negli autovettori associati agli autovalori più piccoli, l'approssimazione appare meno evidente e non presenta una numerosa componente di elementi con variazione sopra la media.



$N_{max} = 3$ $N_{max} = 5$ $N_{max} = 15$

Figura 3.9: SPAI: Approssimazione di $cos(\theta)$, autovettori corrispondenti ai 50 autovalori maggiori di `nos5` (ordine autovalori decrescente)

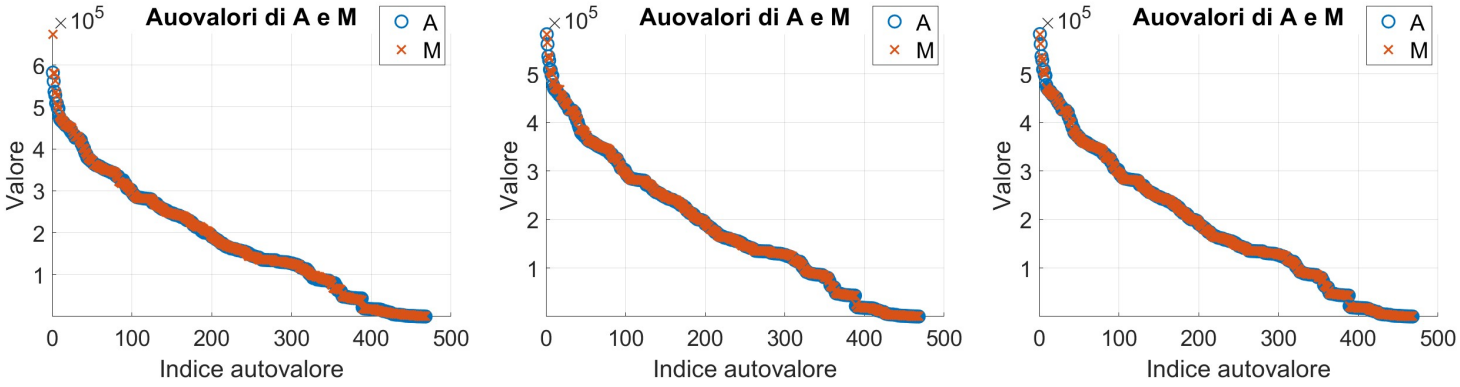


$N_{max} = 3$ $N_{max} = 5$ $N_{max} = 15$

Figura 3.10: SPAI: Approssimazione di $cos(\theta)$.
Aumenta la variazione media di approssimazione

Matrice `nos5`: analisi FSAI

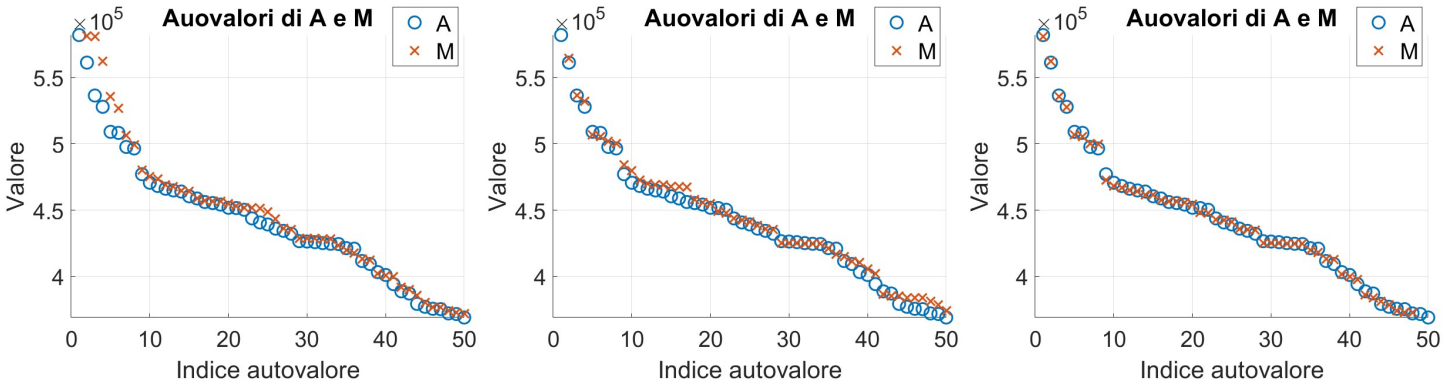
In questa sezione verranno mostrati i risultati riguardanti la procedura FSAI applicati alla matrice `nos5`. I parametri che variano nel confronto dei risultati riguardano il numero di iterazioni q per la generazione del pattern statico di G. Come per i test SPAI, in questo caso si nota l'approssimazione dell'intero spettro degli autovalori della matrice in figura 3.11. Rispetto a SPAI, in questo caso si può direttamente visualizzare già nel primo grafico, con $q = 3$, che l'approssimazione degli autovalori risulta omogenea in tutto lo spettro. Tale grafico, tuttavia, non consente di osservare in modo chiaro l'andamento dell'approssimazione degli autovalori maggiori. Per ovviare a questa limitazione, come già fatto per SPAI, si è realizzato un grafico che evidenzia l'andamento delle approssimazioni dei primi 50 autovalori maggiori di A.



$q = 3$ $q = 5$ $q = 7$

Figura 3.11: FSAI: Approssimazione spettro matrice `nos5`

Il grafico riportato il 3.12, mostra l'andamento dell'approssimazione dei primi 50 autovalori maggiori. Ad una prima analisi, si può notare come l'errore di approssimazione si riduca effettivamente all'aumentare del parametro q. Bisogna comunque notare che da un aumento del parametro q, si ricava, come visto nel precedente capitolo, un pattern iniziale sempre più denso. Il fenomeno quindi di approssimazione degli autovalori, e la riduzione dell'errore di tale approssimazione, si riconduce nel caso FSAI, all'aumento della densità del pattern di G. Bisogna comunque constatare che la procedura per ottenere il pattern di G, risulta dai grafici effettivamente coerente con l'approssimazione di A^{-1}, notando come all'aumentare

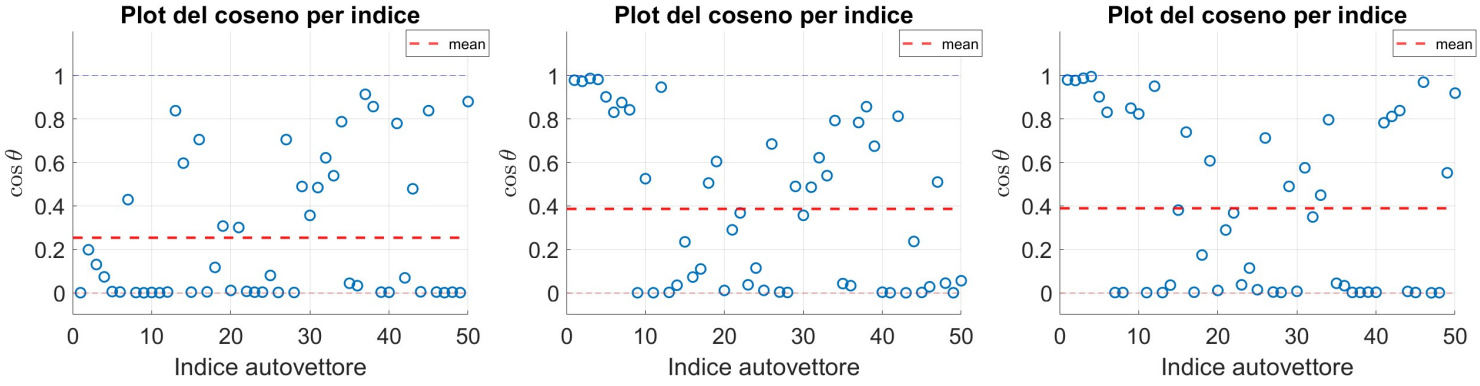


$q = 3$ $q = 5$ $q = 7$

Figura 3.12: FSAI: Approssimazione dei primi 50 autovalori maggiori (ordine autovalori decrescente)

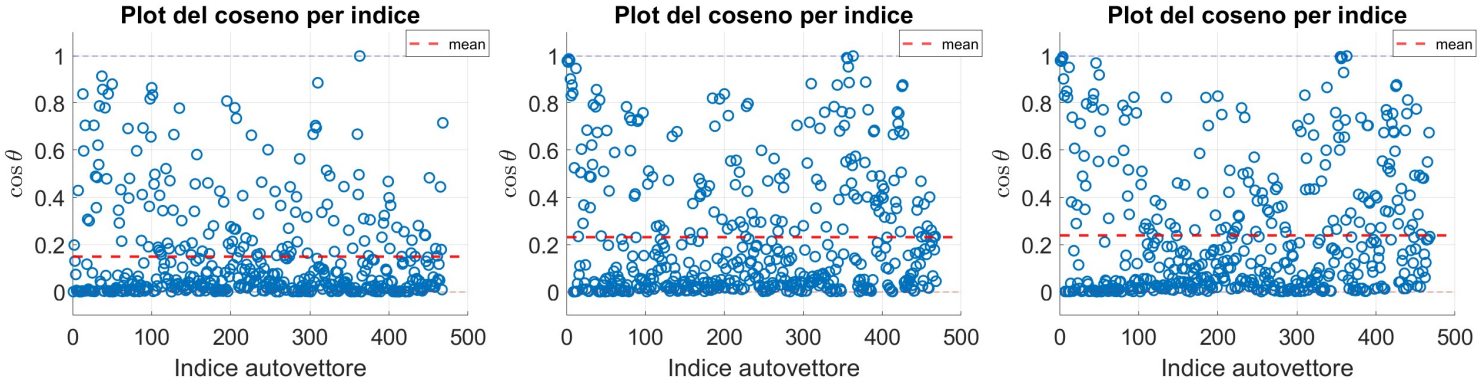
di q, si ha un effetto di convergenza della curva di approssimazione formata dai valori di $\hat{\lambda}$ verso la curva degli autovalori λ di A.

Per quanto riguarda invece gli autovettori di A, la loro approssimazione è riportata nelle figure successive 3.13 e 3.14, utilizzando la metrica del coseno. In particolare, in figura 3.13, dove sono riportati gli autovettori corrispondenti ai 50 autovalori maggiori, si riscontra un andamento crescente della loro approssimazione, così come già ottenuto nella controparte SPAI. Nel passaggio dal primo grafico con $q = 3$ al secondo grafico con $q = 5$, si riscontra un aumento netto dell'approssimazione degli autovettori, con un andamento che rispecchia l'aumento medio dell'approssimazione riportato in figura 3.14. Le variazioni delle approssimazioni risultano più uniformi rispetto al caso SPAI, e si riscontra principalmente una netta approssimazione dell'autovettore corrispondente all'autovalore estremale maggiore, come mostrato nel secondo e terzo grafico di 3.14. Riconsiderando invece i plot in figura 3.13, nel passaggio dal secondo grafico $q = 5$ al terzo $q = 7$, si è ottenuta una un'approssimazione media quasi invariata. In sintesi, dalla generazione del pattern statico per la procedura FSAI riguardo il caso della matrice `nos5`, si è ottenuto un riscontro positivo nell'approssimazione generale degli autovalori, con una approssimazione media degli autovettori associati agli autovalori maggiori della matrice superiore alla media totale.



$q = 3$ $q = 5$ $q = 7$

Figura 3.13: FSAI: Approssimazione di $cos(\theta)$, autovettori corrispondenti ai 50 autovalori maggiori di `nos5` (ordine autovalori decrescente)

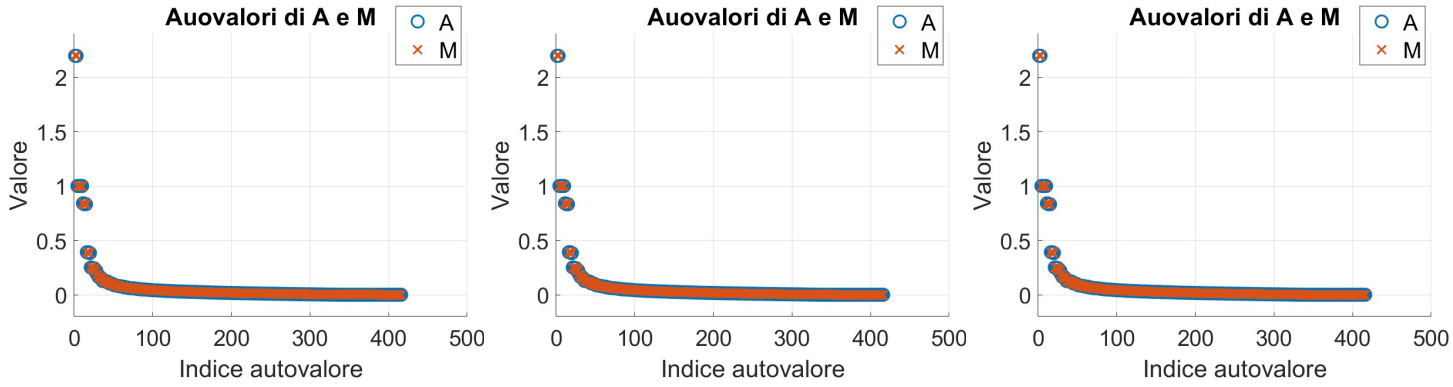


$q = 3$ $q = 5$ $q = 7$

Figura 3.14: FSAI: Approssimazione di $cos(\theta)$

Matrice `mhdb416`: analisi SPAI

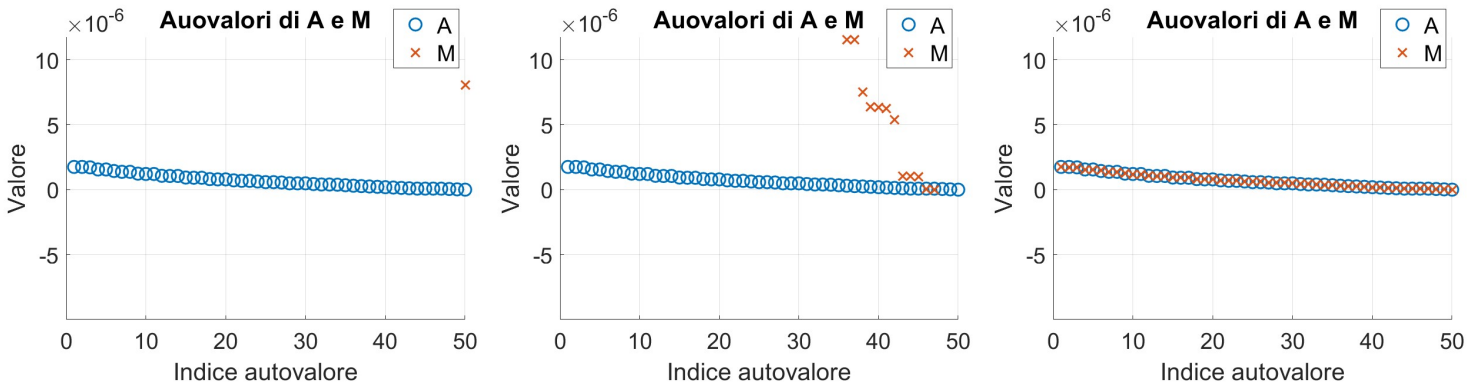
La seconda analisi effettuata riguarda la matrice `mhdb416`. La matrice in questo caso presenta, differentemente dalla matrice `nos5` precedente, degli autovalori estremali pari a $\lambda_{max} \approx 2.19$, e per il minimo $\lambda_{min} \approx 5.249 \times 10^{-10}$. In questo caso quindi, in un ottica di approssimazione dell'inversa, è interessante notare come viene approssimato l'autovalore minore, poiché l'autovalore maggiore è già quasi prossimo all'unità. Analogamente all'analisi precedente, si considera inizialmente l'approssimazione dell'intero spettro, per poi analizzare lo spettro dei 50 autovalori in questo caso minori di A. Nella figura 3.15 è riportata l'approssimazione dell'intero spettro degli autovalori, al variare del parametro di riferimento per il pattern N_{max}. In questo caso, a differenza del caso precedente con `nos5`, si riscontra già inizialmente una curva che rispecchia l'andamento dell'approssimazione. Va però notato che gli ordini di grandezza in gioco sono differenti rispetto a `nos5`. L'interesse principale è osservare come si comporta per le approssimazioni degli autovalori inferiori, e quindi è utile considerare le quantità ottenute per tali autovalori.



$N_{max} = 3$ $N_{max} = 5$ $N_{max} = 15$

Figura 3.15: SPAI: Approssimazione spettro di `mhdb416`

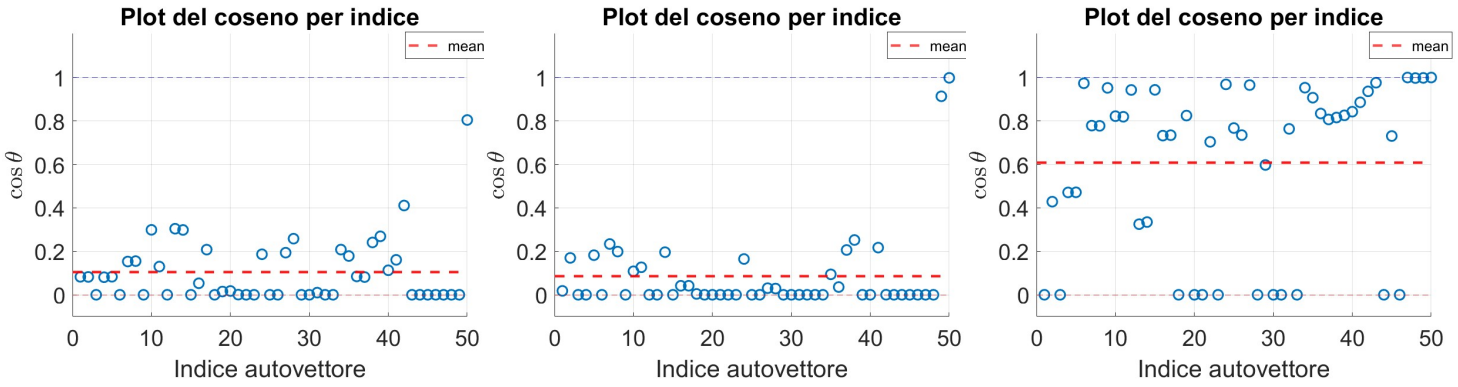
Le approssimazioni ottenute per gli autovalori inferiori sono riportare nella figura successiva 3.16. Dai grafici in successione si nota l'andamento convergente dell'approssimazione verso gli autovalori di A. Nel grafico a sinistra, con $N_{max} = 3$, non c'è traccia, se non per l'autovalore inferiore, di una approssimazione netta per gli autovalori inferiori. L'aumento del numero di indici possibili da inserire in ciascun colonna, passando quindi al parametro con valore $N_{max} = 5$, favorisce



$N_{max} = 3$ $N_{max} = 5$ $N_{max} = 15$

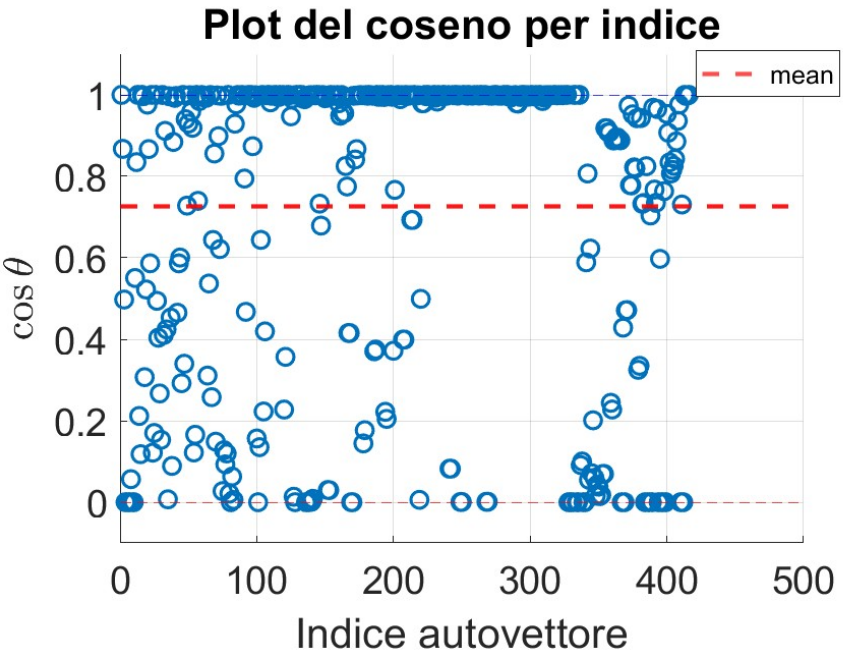
Figura 3.16: SPAI: Approssimazione dei 50 autovalori minori di `mhdb416` (ordine autovalori decrescente)

l'approssimazione che risulterà pienamente completa per $N_{max} = 15$. Anche in questo caso è stata effettuata un'analisi riguardante gli autovettori associati agli autovalori estremali, in questo caso inferiori. Visualizzando l'andamento del coseno per gli autovettori associati, si nota principalmente che con l'approssimazione completa degli autovalori, si ha una variazione anche dell'approssimazione degli autovettori. La variazione degli autovettori però, così come riscontrato nel caso `nos5` con SPAI, risulta presentarsi come una variazione positiva in media, e non per il singolo elemento. Infatti, fatta eccezione per l'autovettore corrispondente all'autovalore minore, in cui si nota una tendenza convergente netta, non si riscontra una convergenza monotona nei singoli autovettori considerando il grafico con $N_{max} = 3$ a $N_{max} = 5$. La relazione fra approssimazione degli autovalori e degli autovettori per $N_{max} = 15$, quindi quando il parametro permette di aggiungere una quantità sufficiente di elementi per colonna, risulta comunque evidente dai risultati ottenuti per questa simulazione. Infatti nel grafico 3.18 si riscontra un numero ampio numero di autovettori convergenti all'autovettore esatto corrispondente, con una media di approssimazione > 0.7, evidenziata dal segmento rosso in figura. Dai grafici in figura 3.15, 3.17 e 3.18, riguardo al parametro $N_{max} = 15$, per la matrice `mhdb416`, si ha che la bontà del metodo SPAI nell'approssimazione dell'inversa, ha diretta conseguenza nell'approssimazione sia degli autovalori di A, sia degli autovettori associati.



$N_{max} = 3$ $N_{max} = 5$ $N_{max} = 15$

Figura 3.17: SPAI: Approssimazione autovettori associati ai 50 autovalori minori di `mhdb416` (ordine autovalori decrescente)



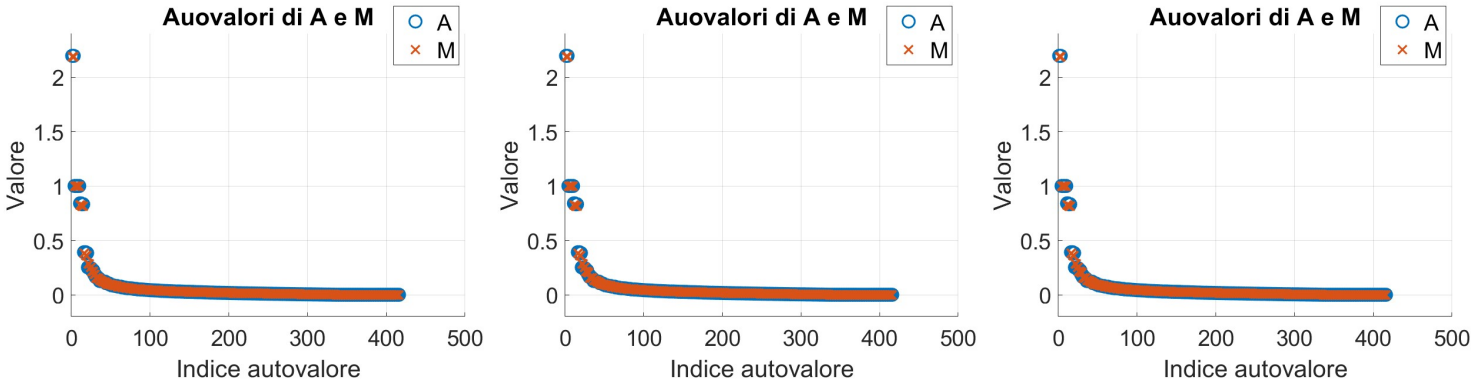
$N_{max} = 15$

Figura 3.18: SPAI: Approssimazione autovettori di `mhdb416` (ordine autovalori decrescente)

Matrice `mhdb416`: analisi FSAI

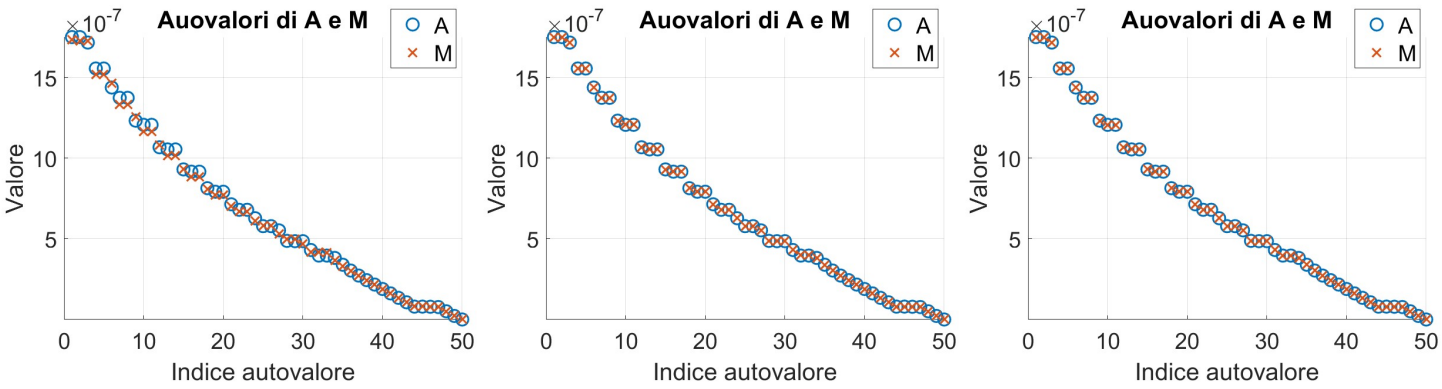
Seguendo un'impostazione analoga all'analisi precedente, si inizia esaminando l'approssimazione dell'intero spettro, per poi focalizzarsi sui 50 autovalori minori della matrice A. Nella figura 3.20 è illustrata l'accuratezza dell'approssimazione spettrale complessiva, in relazione alla variazione del parametro di riferimento per il pattern q. Anche per FSAI, diversamente dal caso `nos5`, in questa configurazione si osserva sin dalle prime iterazioni un andamento della curva che riflette il comportamento dell'approssimazione, come mostrato in figura 3.20. Ma tuttavia l'obiettivo principale di questa analisi è comprendere il comportamento dell'approssimazione in relazione agli autovalori inferiori, rendendo quindi essenziale un'attenta valutazione delle quantità ottenute per questi specifici autovalori. Dati i diversi ordini di grandezza degli autovalori di `mhdb416`, si ripropone come fatto precedentemente in SPAI, di visualizzare solamente l'andamento degli autovalori minori. Nel grafico riguardante i 50 autovalori inferiori, in figura 3.21, si nota chiaramente un comportamento convergente degli autovalori inferiori all'autovalore esatto di riferimento, al crescere del parametro q. La convergenza risulta essere già adeguata con $q = 2$, sinonimo del fatto che l'identificazione del pattern a priori mediante la procedura FSAI risulta ottimale. L'evoluzione dell'approssimazione dello spettro

Figura 3.19:
Approssimazione dello spettro di `mhdb416` mediante procedura FSAI



$q = 2$ $q = 4$ $q = 6$

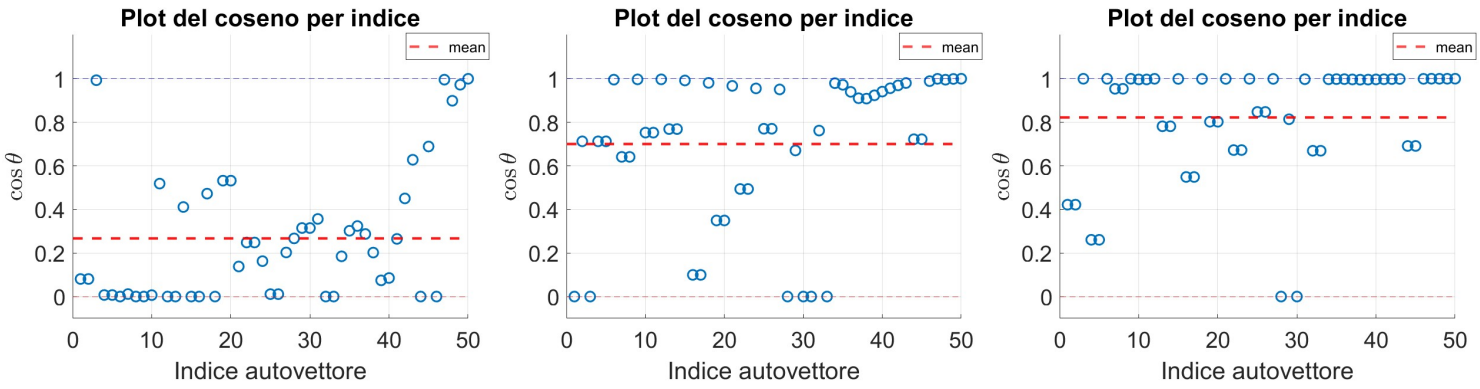
Figura 3.20: FSAI: approssimazione spettro di A



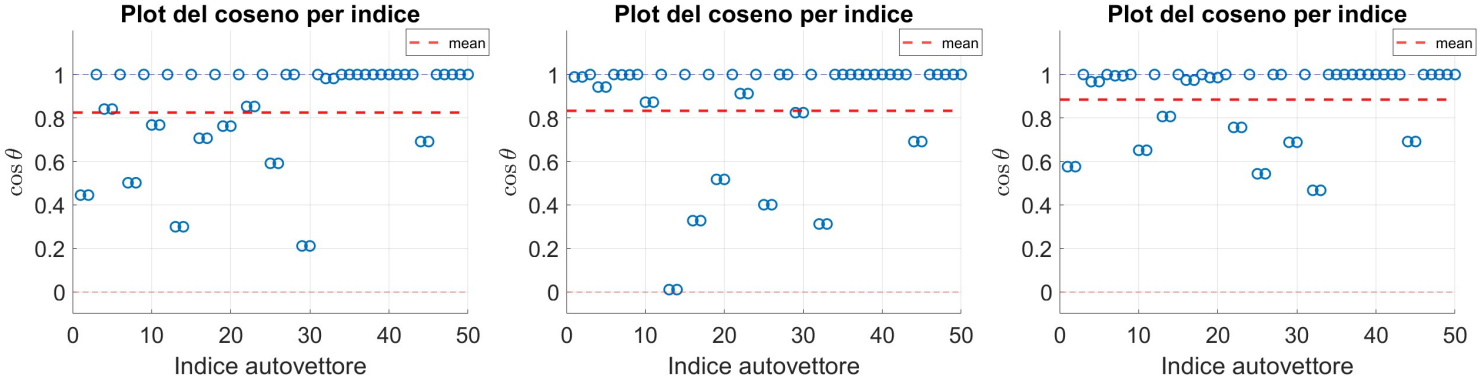
$q = 2$ $q = 4$ $q = 6$

Figura 3.21: FSAI: approssimazione dei 50 autovalori minori (ordine autovalori decrescente)

in questo caso risulta efficiente già a basse iterazioni. Una simile evoluzione si è anche riscontrata nell'approssimazione degli autovettori.
Nella figura successiva 3.22 sono riportate le approssimazioni degli autovettori di A associati ai 50 autovalori minori. Ogni grafico è associato a un differente valore del parametro q, utilizzato nella generazione del pattern per la procedura FSAI. In una prima analisi, si osserva che la media dell'approssimazione cresce all'aumentare del parametro q. Relativamente all'approssimazione degli autovalori, si nota quindi che l'approssimazione autovalore-autovettore segue un andamento generalmente crescente. La differenza tra le medie nei vari grafici è particolarmente evidente nella transizione da $q = 1$ a $q = 2$, dove si registra un significativo incremento della media di approssimazione, che passa da un valore di circa 0.3 a circa 0.7. Con l'incremento di q, l'approssimazione tende a assestarsi, pur mantenendo un andamento crescente. Inoltre, sebbene in questo caso si osservi una iniziale approssimazione notevole dell'autovettore associato all'autovalore inferiore, si conferma, come già evidenziato in precedenza, una tendenza generale alla crescita dell'approssimazione media anche nei restanti autovettori associati agli autovalori inferiori.



$q = 1$ $q = 2$ $q = 3$



$q = 4$ $q = 5$ $q = 6$

Figura 3.22: FSAI: Approssimazione autovettori associati ai 50 autovalori minori (ordine autovalori decrescente)

Conclusioni

L'oggetto della presente tesi è stata la discussione delle tecniche di precondizionamento tramite sparse approximate inverse, con particolare focus sugli algoritmi SPAI (Sparse Approximate Inverse), ideato per matrici generiche, e sulla sua variante FSAI (Factorized Sparse Approximate Inverse), ideato per matrici simmetriche definite positive. Questi metodi sono stati introdotti grazie ai contributi pionieristici di Grote e Huckle, in cui l'idea principale è trovare un'approssimazione $M \approx A^{-1}$ attraverso un problema di minimizzazione. Sebbene esista un ampio panorama di precondizionatori in letteratura, gli algoritmi SPAI e FSAI si distinguono per la loro natura intrinsecamente parallela, rendendoli quindi procedure altamente scalabili.
Un aspetto cruciale per il successo di questi algoritmi è lo sparsity pattern iniziale in input, che ha un impatto significativo sull'efficienza dell'approssimazione dell'inversa. I risultati numerici ottenuti hanno dimostrato che, mediante l'adozione di euristiche mirate, è possibile ottenere approssimazioni più precise dell'inversa A^{-1}, con evidenti miglioramenti nelle prestazioni del precondizionatore.
I test effettuati su diverse matrici hanno confermato che SPAI e FSAI soddisfano i criteri per un buon precondizionatore, ottenendo una diminuzione significativa del numero di condizionamento, preservando al contempo una struttura sparsa per M. Inoltre, l'analisi dello spettro della matrice A e la valutazione degli autovettori associati all'approssimazione tramite M, hanno evidenziato un comportamento coerente con l'obiettivo di ottenere una buona approssimazione dell'inversa, e di ridurre il numero di condizionamento.
In conclusione, i risultati ottenuti suggeriscono che i precondizionatori SPAI e FSAI, grazie alla loro natura parallelizzabile e alla capacità di adattarsi alle strutture delle matrici di interesse, rappresentano un approccio promettente per ottimizzare il precondizionamento di sistemi lineari. Le direzioni future di ricerca potrebbero includere l'ulteriore ottimizzazione delle euristiche relative allo sparsity pattern e l'estensione di questi algoritmi a contesti più complessi, come i sistemi su larga scala e gli ambienti distribuiti.

Ringraziamenti

I traguardi importanti non si conquistano da soli, ed è per questo che dietro il conseguimento di questa laurea ci sono anche altre persone che mi hanno sostenuto.

Innanzitutto, desidero esprimere la mia gratitudine al Prof. Stefano Berrone, il mio relatore, per i Suoi consigli ed il Suo supporto durante la stesura di questo elaborato.

I miei genitori, che mi hanno sostenuto emotivamente ed economicamente in questo mio percorso di studi. Loro prima di me stesso, hanno capito quanto per frequentare questo percorso universitario la serenità sia molto importante e per questo gli sono veramente grato. Nonostante fossi abbattuto e snervato, in quei periodi durante il percorso di studi dove si presentavano grandi difficoltà, loro mi hanno sempre aiutato a superarli avendo piena fiducia nelle mie capacità, incoraggiandomi ad essere lucido e a non farmi sopraffare dalla fretta e dai risultati che a volte, anche sforzandoci, non arrivano nei tempi desiderati. La loro felicità per il percorso che stavo intraprendendo e la loro gioia per i miei risultati, anche per il più semplice esame superato, mi ha aiutato a continuare con sempre più convinzione e motivazione il mio percorso.

I miei fratelli Elisa, Andrea e Gianni, che mi hanno aiutato a scegliere questo corso di laurea, aiutandomi a capire quale fosse la scelta migliore per la strada che desideravo intraprendere, tutto ciò nonostante le difficoltà che sapevano avrei incontrato nel mio percorso. Mi hanno insegnato come andare avanti quando si rimane bloccati nel percorso e come superare tali situazioni che ci ostacolano, e che impediscono molte volte di proseguire con serenità gli studi. Mi hanno introdotto per primi nel mondo universitario, dandomi già la conoscenza e l'esperienza per orientarmi e organizzarmi nel sistema universitario. Mi hanno insegnato molto e continuano anche oggi a insegnarmi come crescere e come affrontare le questioni più importanti della vita sociale e personale. Sono sempre pronti ad ascoltarmi, specialmente nelle questioni più personali, dove spiegarsi è difficile ed essere capito lo è ancora di più, ma pur sapendo come profondamente i miei fratelli mi conoscono, resto sempre stupito di come riescono a capirmi e consigliarmi la giusta soluzione.

La mia fidanzata Cristina, che mi ha sempre supportato durante tutto il mio percorso universitario. Mi ha insegnato ad essere diligente, a programmarmi lo studio, e all'uso dei "programmini" giornalieri per organizzare le giornate, fra lezioni, studio e riposo. Che mi è sempre stata accanto durante tutte le sessioni, anche quando era impegnata con gli esami universitari. Grazie a lei ho potuto migliorare il mio approccio universitario, a battere i miei record e a prendere coscienza che il giusto impegno viene ripagato. Ha saputo risollevarmi quando ero giù nelle mie sessioni che definivo inconcludenti, e ha gioito con me quando riuscivo a portare a termine gli esami. Ma non solo, sapendo la mia tendenza a lamentarmi, soprattutto durante le sessioni d'esame, hai sempre avuto la pazienza di ascoltarmi e di capirmi, e di questo ti sono davvero grato!

I miei amici di una vita, che sono la mia seconda famiglia. Vi ringrazio per essermi stati sempre vicino in questo percorso universitario, nonostante la lunga distanza che ci separa e la sistemazione fuori sede che permette solo brevi periodi in cui possiamo rincontrarci per stare insieme. Sono felice di aver condiviso con voi gioie e dolori del mio percorso ma soprattutto per il vostro entusiasmo nel sapere che sono riuscito a completarlo. Ci divertiamo e scherziamo insieme, ma anche ci si confronta e ci si aiuta nelle scelte personali e generali. Un punto d'appoggio di cui sai di poterti fidare, sapendo di poter parlare sia di ciò che diverte e piace, sia di quello che ti turba, soprattutto senza la necessità a volte di dover spiegare a lungo, poiché proprio perché ti conoscono a fondo, sanno cosa stai pensando.

Voglio ringraziare tutti gli amici che ho conosciuto qui a Torino, gli amici del collegio Onaosi e dell'università. Durante il mio percorso universitario ho avuto modo di conoscere delle splendide persone che si sono rilevate dei grandi amici, con cui condividere sia l'emozione dei propri esami superati, dei successi del proprio percorso, sia i fallimenti e le delusioni per gli obiettivi non raggiunti e le difficoltà che a volte non vengono superate nonostante la grande fatica impiegata. Vi ringrazio non solo per i momenti di divertimento e le serate di spensieratezza, ma anche per avermi aiutato quando nel mio percorso di studi mi sono sentito bloccato, facendomi capire consapevolmente che bisogna sempre essere pronti a migliorare e a correggere i propri errori.

Ci tengo a ringraziare tutte le persone che mi sono state vicino, la mia famiglia, la mia fidanzata, i miei parenti, gli amici e vi ringrazio fortemente per avermi sostenuto in questo mio percorso di laurea!

Bibliografia

[1] Michele Benzi. Preconditioning techniques for large linear systems: A survey. *Journal of Computational Physics*, 182(2):418–477, November 2002. ISSN 0021-9991. doi: 10.1006/jcph.2002.7176. URL http://dx.doi.org/10.1006/jcph.2002.7176.

[2] Michele Benzi and Miroslav Tuma. A sparse approximate inverse preconditioner for nonsymmetric linear systems. *SIAM Journal on Scientific Computing*, 19(3):968–994, May 1998. ISSN 1095-7197. doi: 10.1137/s1064827595294691. URL http://dx.doi.org/10.1137/S1064827595294691.

[3] Michele Benzi, Carl D. Meyer, and Miroslav Tůma. A sparse approximate inverse preconditioner for the conjugate gradient method. *SIAM Journal on Scientific Computing*, 17(5):1135–1149, September 1996. ISSN 1095-7197. doi: 10.1137/s1064827594271421. URL http://dx.doi.org/10.1137/S1064827594271421.

[4] Michele Benzi, Jane K. Cullum, and Miroslav Tuma. Robust approximate inverse preconditioning for the conjugate gradient method. *SIAM Journal on Scientific Computing*, 22(4):1318–1332, January 2000. ISSN 1095-7197. doi: 10.1137/s1064827599356900. URL http://dx.doi.org/10.1137/S1064827599356900.

[5] Massimo Bernaschi, Mauro Bisson, Carlo Fantozzi, and Carlo Janna. A factored sparse approximate inverse preconditioned conjugate gradient solver on graphics processing units. *SIAM Journal on Scientific Computing*, 38(1): C53–C72, January 2016. ISSN 1095-7197. doi: 10.1137/15m1027826. URL http://dx.doi.org/10.1137/15M1027826.

[6] Edmond Chow. A priori sparsity patterns for parallel sparse approximate inverse preconditioners. *SIAM Journal on Scientific Computing*, 21(5): 1804–1822, January 2000. ISSN 1095-7197. doi: 10.1137/s106482759833913x. URL http://dx.doi.org/10.1137/S106482759833913X.

[7] Edmond Chow and Yousef Saad. Experimental study of ilu preconditioners for indefinite matrices. *Journal of Computational and Applied Mathematics*, 86(2):387–414, 1997. ISSN 0377-0427. doi: https://doi.org/10.1016/S0377-0427(97)00171-4. URL https://www.sciencedirect.com/science/article/pii/S0377042797001714.

[8] Timothy A. Davis and Yifan Hu. The university of florida sparse matrix collection. *ACM Transactions on Mathematical Software*, 38(1):1–25, November 2011. ISSN 1557-7295. doi: 10.1145/2049662.2049663. URL http://dx.doi.org/10.1145/2049662.2049663.

[9] Nicholas I. M. Gould and Jennifer A. Scott. Sparse approximate-inverse preconditioners using norm-minimization techniques. *SIAM Journal on Scientific Computing*, 19(2):605–625, 1998. doi: 10.1137/S1064827595288425. URL https://doi.org/10.1137/S1064827595288425.

[10] Marcus J. Grote and Thomas Huckle. Parallel preconditioning with sparse approximate inverses. *SIAM Journal on Scientific Computing*, 18(3):838–853, May 1997. ISSN 1095-7197. doi: 10.1137/s1064827594276552. URL http://dx.doi.org/10.1137/S1064827594276552.

[11] Thomas Huckle. Approximate sparsity patterns for the inverse of a matrix and preconditioning. *Applied Numerical Mathematics*, 30(2):291–303, 1999. ISSN 0168-9274. doi: https://doi.org/10.1016/S0168-9274(98)00117-2. URL https://www.sciencedirect.com/science/article/pii/S0168927498001172.

[12] Higham N. J. *Accuracy and Stability of Numerical Algorithms, 2nd edn.* SIAM Publications., 2002.

[13] Carlo Janna, Massimiliano Ferronato, Flavio Sartoretto, and Giuseppe Gambolati. Fsaipack: A software package for high-performance factored sparse approximate inverse preconditioning. *ACM Trans. Math. Softw.*, 41(2), February 2015. ISSN 0098-3500. doi: 10.1145/2629475. URL https://doi.org/10.1145/2629475.

[14] L. Yu. Kolotilina and A. Yu. Yeremin. Factorized sparse approximate inverse preconditionings i. theory. *SIAM Journal on Matrix Analysis and Applications*, 14(1):45–58, January 1993. ISSN 1095-7162. doi: 10.1137/0614004. URL http://dx.doi.org/10.1137/0614004.

[15] L. YU. KOLOTILINA and A. YU. YEREMIN. Factorized sparse approximate inverse preconditioning ii: Solution of 3d fe systems on massively parallel computers. *International Journal of High Speed Computing*, 07(02):191–215,

June 1995. ISSN 0129-0533. doi: 10.1142/s0129053395000117. URL http://dx.doi.org/10.1142/S0129053395000117.

[16] MathWorks. Matlab documentation center. URL http://www.mathworks.it/it/help/matlab/.

[17] Yousef Saad. *Iterative methods for sparse linear systems.* SIAM, 2003. ISBN 978-0-89871-534-7.

[18] K. Wang, O. Lawlor, and L. V. Kale. The nonsingularity of sparse approximate inverse preconditioning and its performance based on processor virtualization. Technical report, Department of Computer Science, University of Illinois at Urbana-Champaign, 2005. Available at https://www.researchgate.net/publication/228574264_The_Nonsingularity_of_Sparse_Approximate_Inverse_Preconditioning_and_Its_Performance_Based_on_Processor_Virtualization.

[19] A. Yu. Yeremin and A. A. Nikishin. Factorized-sparse-approximate-inverse preconditionings of linear systems with unsymmetric matrices. *Journal of Mathematical Sciences*, 121(4):2448–2457, June 2004. ISSN 1072-3374. doi: 10.1023/b:joth.0000026282.08256.45. URL http://dx.doi.org/10.1023/B:JOTH.0000026282.08256.45.