Università' degli studi di Cagliari
Dipartimento di Matematica e Informatica
Corso di Laurea Magistrale in Matematica

Metodi iterativi per la risoluzione di sistemi lineari

Studente: Marco Ratto

Algoritmi Numerici e Applicazioni
Prof. Giuseppe Rodriguez

A.A. 2021/2022

Indice

1 Introduzione **2**

2 Metodi iterativi **3**
2.1 Metodi del prim'ordine 3
2.2 Metodi di Jacobi e Gauss-Seidel 4
2.3 Metodo del gradiente 5
2.4 Metodo del gradiente coniugato 6
2.5 Precondizionamento 7

3 Risultati **8**
3.1 Gradiente e gradiente coniugato 8
3.2 Gradiente coniugato precondizionato 9
3.3 Ulteriori test 10

4 Conclusioni **13**

1 Introduzione

Quando vogliamo risolvere sistemi lineari del tipo

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

dove A è una matrice $n \times n$ sparsa, con n molto grande, i metodi diretti non risultano essere efficaci. Questo succede perché questi ultimi, per esempio la fattorizzazione LU o la fattorizzazione QR agiscono modificando la matrice di partenza A, con il risultato che i fattori ottenuti perdono la proprietà di sparsità, generando problemi sia nel tempo di calcolo, sia nella memorizzazione. Per evitare questi problemi, vengono introdotti i cosiddetti **metodi iterativi**.

In questa tesina parleremo in breve dei metodi iterativi e vedremo nello specifico alcuni di questi. La trattazione teorica seguirà lo schema utilizzato in [1]. Successivamente vedremo i risultati di una breve sperimentazione effettuata sui metodi visti, utilizzando il software Matlab.

2 Metodi iterativi

I metodi iterativi per la risoluzione di sistemi lineari consistono nella creazione, a partire da un vettore $\mathbf{x}^{(0)}$, di una successione $\mathbf{x}^{(k)}$, $k = 0, 1, ...$ che converga alla soluzione esatta del sistema.

Questi metodi presentano diversi vantaggi rispetto ai metodi diretti:

1. **Tempo di calcolo:** come abbiamo detto, i metodi iterativi consentono di mantenere la proprietà di sparsità della matrice A e questo permette di svolgere prodotti matrice-vettore in maniera molto più rapida;
2. **Scelta del vettore di partenza:** la scelta di $\mathbf{x}^{(0)}$ ci permette di sfruttare eventuali informazioni a priori sul sistema che si vuole risolvere per arrivare più rapidamente alla covergenza;
3. **Criteri di arresto:** trattandosi di metodi che convergono alla soluzione esatta, posso decidere quando arrestare il mio processo a seconda della precisione che voglio ottenere.

Diremo che un metodo iterativo **converge globalmente** se per ogni vettore iniziale $\mathbf{x}^{(0)} \in \mathbb{R}^n$ si ha che

$$\lim_{k\to\infty} \|\mathbf{x}^{(k)} - \mathbf{x}\| = 0\,,$$

o, equivalentemente, se $\lim_{k\to\infty} \|\mathbf{e}^{(k)}\| = 0$, dove $\mathbf{e}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}$ è l'**errore** al passo k.

2.1 Metodi del prim'ordine

Diamo adesso alcune definizioni utili per classificare i metodi iterativi.

Definizione: Un metodo iterativo si dice **del prim'ordine** se il calcolo di un termine della successione dipende solamente dal termine precedente, cioè:

$$\mathbf{x}^{(k+1)} = \psi\left(\mathbf{x}^{(k)}\right).$$

Definizione: Un metodo iterativo del prim'ordine si dice **lineare** se la relazione per determinare i termini della successione lo è:

$$\mathbf{x}^{(k+1)} = B_k\,\mathbf{x}^{(k)} + \mathbf{f}_k\,.$$

Definizione: Un metodo iterativo si dice **stazionario** se il passo per determinare un nuovo elemento della successione non dipende dall'indice di iterazione.

Quindi un metodo iterativo **lineare**, **stazionario del prim'ordine** assume la forma

$$\mathbf{x}^{(k+1)} = B\,\mathbf{x}^{(k)} + \mathbf{f}\,.$$

Si può dimostrare il seguente teorema che fornisce una condizione necessaria e sufficiente per la convergenza:

Teorema: Un metodo iterativo lineare è convergente se e solo se il raggio spettrale $\rho(B)$ della matrice di iterazione B è minore di 1.

2.2 Metodi di Jacobi e Gauss-Seidel

Per costruire alcuni metodi iterativi lineari si procede scomponendo la matrice A utilizzando il cosiddetto *splitting additivo*, cioè si scrive:

$$A = P - N,$$

con P matrice invertibile. Solitamente si sceglie P *facilmente* invertibile, per esempio diagonale o triangolare.
A questo punto poniamo

$$B = P^{-1}N, \qquad f = P^{-1}\mathbf{b}\,,$$

e risulta definito il metodo iterativo lineare:

$$\mathbf{x}^{(k+1)} = P^{-1}N\,\mathbf{x}^{(k)} + P^{-1}\mathbf{b}\,.$$

Come particolare *splitting additivo* possiamo utilizzare

$$A = D - E - F\,,$$

dove D è la matrice diagonale

$$D_{ij} = \begin{cases} a_{ii} & \text{se } i = j \\ 0 & \text{se } i \neq j\,, \end{cases}$$

mentre E ed F sono le matrici triangolari

$$E_{ij} = \begin{cases} -a_{ij} & \text{se } i > j \\ 0 & \text{se } i \leq j \end{cases}, \qquad F_{ij} = \begin{cases} -a_{ij} & \text{se } i < j \\ 0 & \text{se } i \geq j \end{cases}.$$

Effettuando opportune permutazioni su A ci possiamo assicurare che D sia non singolare, in questo modo possiamo porre

$$P = D, \qquad N = E + F,$$

da cui si ricava

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

Questo metodo è chiamato **metodo di Jacobi**.

La matrice di iterazione è

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

Dal teorema enunciato in precedenza possiamo dire che il metodo di Jacobi converge se e solo se $\rho(I - D^{-1}A) < 1$.

Un'altra possibile scelta è quella di porre

$$P = D - E, \qquad N = F,$$

da cui si ottiene

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

dalla quale si può ottenere $\mathbf{x}^{(k+1)}$ risolvendo un sistema triangolare.

Questo metodo è chiamato **metodo di Gauss-Seidel**.

2.3 Metodo del gradiente

Se A è simmetrica definita positiva possiamo introdurre un nuovo metodo, chiamato **metodo del gradiente**.

Per iniziare consideriamo la forma quadratica $\phi : \mathbb{R}^n \to \mathbb{R}$ definita come

$$\phi(\mathbf{y}) = \frac{1}{2}\mathbf{y}^T A\mathbf{y} - \mathbf{y}^T\mathbf{b},$$

il cui gradiente è

$$\nabla\phi(\mathbf{y}) = A\mathbf{y} - \mathbf{b}.$$

Essendo A definita positiva, il punto in cui il gradiente è nullo risulta essere un punto di minimo, quindi si ha che la risoluzione del sistema $A\mathbf{x} = \mathbf{b}$ è equivalente al problema di minimizzazione di $\phi(\mathbf{y})$.

Proviamo a calcolare questo minimo con un metodo non stazionario:

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

quindi cerchiamo di determinare α_k in modo che $\phi\left(\mathbf{x}^{(k+1)}\right)$ sia minima per la direzione $\mathbf{d}^{(k)}$.

Sviluppando l'espressione di $\phi\left(\mathbf{x}^{(k+1)}\right)$ e imponendo che la sua derivata rispetto ad α sia uguale a 0, si ottiene

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

con $\mathbf{r}^{(k)} = \mathbf{b} - A\mathbf{x}^{(k)}$ vettore **residuo**.

Ci resta da scegliere la direzione $\mathbf{d^{(k)}}$: essendo il gradiente la direzione di massima crescita, noi scegliamo l'opposto del gradiente, cioè la direzione di massima discesa (*steeepest descent*):

$$\mathbf{d^{(k)}} = -\nabla\phi\left(\mathbf{x}^{(k)}\right).$$

2.4 Metodo del gradiente coniugato

Vediamo alcuni concetti teorici che saranno fondamentali per lo sviluppo del metodo del gradiente coniugato.

Definizione: $\mathbf{x}^{(k)}$ è **ottimale** rispetto alla direzione $\mathbf{p}$ se

$$\phi\left(\mathbf{x}^{(k)}\right) \leq \phi\left(\mathbf{x}^{(k)} + \lambda\mathbf{p}\right), \qquad \forall\, \lambda \in \mathbb{R}.$$

Teorema: $\mathbf{x}^{(k)}$ è **ottimale** rispetto alla direzione $\mathbf{p}$ se e solo se $\mathbf{p}^T\mathbf{r}^{(k)} = 0$.

Nel metodo del gradiente $\mathbf{x}^{(k+1)}$ è, per costruzione, ottimale rispetto a $\mathbf{r}^{(k)}$, quindi il teorema appena enunciato ci dice che $\mathbf{r}^{(k+1)}$ è ortogonale a $\mathbf{r}^{(k)}$.
Il problema però è che l'ottimalità rispetto a $\mathbf{r}^{(k)}$ viene persa da $\mathbf{x}^{(k+2)}$.
Vogliamo quindi far sì che l'ottimalità di $\mathbf{x}^{(k)}$ rispetto ad una certa direzione venga ereditata da tutti gli elementi successivi della successione.

Per fare questo, sia $\mathbf{x}^{(k)}$ ottimale rispetto a $\mathbf{p}^{(k)}$ $\left(\left(\mathbf{p}^{(k)}\right)^T \mathbf{r}^{(k)} = 0\right)$, poniamo

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{p}^{(k+1)}.$$

Perché anche $\mathbf{x}^{(k+1)}$ sia ottimale rispetto a $\mathbf{p}^{(k)}$, $\mathbf{p}^{(k)}$ e $\mathbf{p}^{(k+1)}$ devono essere A**-coniugate**, cioè:

$$\left(\mathbf{p}^{(k)}\right)^T A\,\mathbf{p}^{(k+1)} = 0.$$

Consideriamo adesso direzioni del tipo:

$$\mathbf{p}^{(k+1)} = \mathbf{r}^{(k+1)} - \beta_k \mathbf{p}^{(k)},$$

la condizione $\left(\mathbf{p}^{(k)}\right)^T A\,\mathbf{p}^{(k+1)} = 0$ ci consente di ricavare:

$$\beta_k = \frac{\left(\mathbf{p}^{(k)}\right)^T A\,\mathbf{r}^{(k+1)}}{\left(\mathbf{p}^{(k)}\right)^T A\,\mathbf{p}^{(k)}}.$$

In questo modo abbiamo determinato i passi del **metodo del gradiente coniugato**.

Questo metodo risulta raggiungere la soluzione esatta in massimo n iterazioni, quindi potrebbe essere usato come metodo diretto, ma quando la dimensione del problema è molto elevata la successione viene troncata molto prima di aver concluso n iterazioni in quanto l'errore tende a 0 molto rapidamente.

2.5 Precondizionamento

I metodi appena visti possono essere velocizzati notevolmente con l'utilizzo di un **precondizionatore**, ovvero di una matrice P che approssimi A ma che allo stesso tempo sia invertibile con un basso costo computazionale (cosa che non vale per A).

Nei metodi iterativi lineari stazionari, si ha

$$\begin{aligned}\mathbf{x}^{(k+1)} = B\,\mathbf{x}^{(k)} + \mathbf{f} = (I - P^{-1}A)\mathbf{x}^{(k)} + P^{-1}\mathbf{b} = \\ = \mathbf{x}^{(k)} + P^{-1}\left(\mathbf{b} - A\mathbf{x}^{(k)}\right) = \mathbf{x}^{(k)} + P^{-1}\mathbf{r}^{(k)}.\end{aligned}$$

Nel metodo del gradiente, che non è stazionario, diventa

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

dove $\mathbf{z}^{(k)}$ è la soluzione del sistema $P\mathbf{z}^{(k)} = \mathbf{r}^{(k)}$. In questo passaggio entra in gioco la particolare struttura di P che rende semplice il calcolo di $\mathbf{z}^{(k)}$. Un esempio molto efficace di precondizionatore nel caso di matrici di matrici

sparse è la **fattorizzazione** LU **incompleta**.
Sappiamo che nella fattorizzazione LU si verifica il problema che i fattori triangolari perdono la proprietà di sparsità che aveva la matrice di partenza, quindi si fissa una soglia σ e quando gli elementi sono, in valore assoluto, minori di questa soglia vengono posti uguali a 0.
In questo modo si determina il precondizionatore

$$P = LU \approx A$$

con L e U matrici triangolari **sparse**.

Se A è simmetrica definita positiva posso calcolare analogamente la **fattorizzazione di Cholesky incompleta**, dove il precondizionatore sarà

$$P = R^T R \approx A$$

con R matrice triangolare superiore **sparsa**.

3 Risultati

In questa sezione andiamo a vedere alcuni risultati, ottenuti con il software Matlab, che mettono a confronto i metodi del gradiente e del gradiente coniugato. Inoltre vedremo come un opportuno precondizionamento possa ridurre notevolmente il tempo di calcolo e il numero di iterazioni richieste per raggiungere un grado di precisione fissato.
I metodi utilizzati sono stati implementati da me, anche se su Matlab è presente la funzione *pcg* che utilizza il metodo del gradiente coniugato con osenza precondizionamento.

3.1 Gradiente e gradiente coniugato

Cominciamo confrontando il metodo del gradiente con quello del gradiente coniugato senza precondizionamento (CG).
Definiamo una sistema con matrice dei coefficienti sparsa (densità 0.01), simmetrica definita positiva (condizionamento 100) e vediamo al variare della dimensione del problema come si comportano i due metodi. Per farlo utilizziamo la funzione di Matlab *sprandsym*.

Abbiamo fissato come criterio di arresto la norma del vettore residuo minore di 10^{-12} o il un numero massimo di iterazioni, quest'ultimo è stato scelto molto elevato per far sì che si ottenesse sempre la convergenza con gli

esempi analizzati.

Notiamo che qualsiasi sia la dimensione il metodo del gradiente coniugato richiede molte meno iterazioni rispetto al metodo del gradiente (Tabella 1). Questo viene evidenziato anche dal tempo di calcolo che è molto diverso a seconda del metodo utilizzato (Figura 1).

In conclusione possiamo dire il metodo del gradiente coniugato è sempre preferibile rispetto a quello del gradiente.

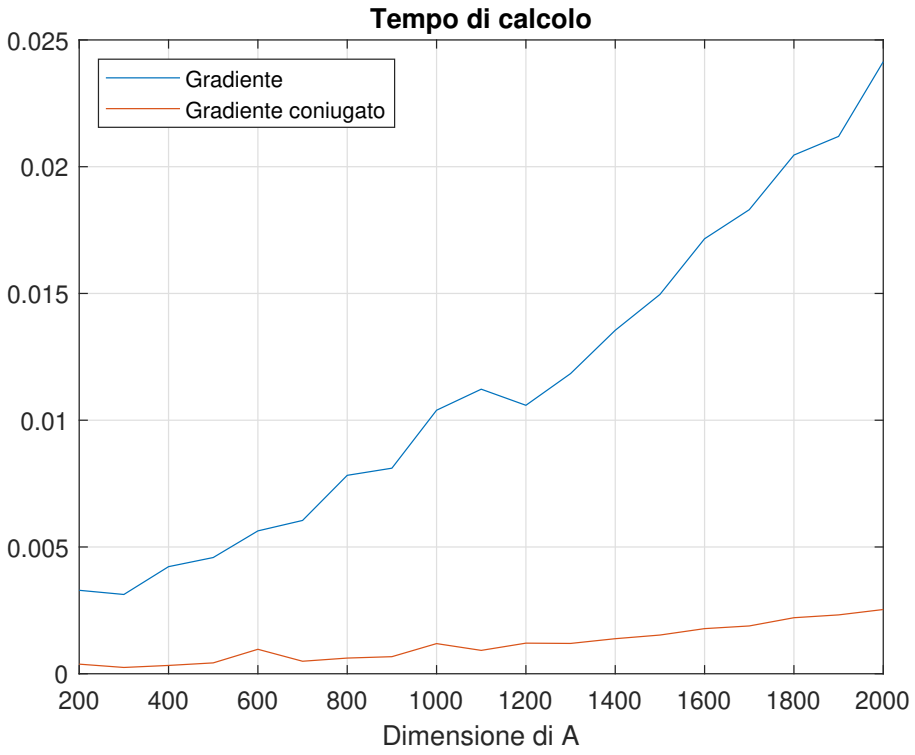

Figura 1: Confronto tra gradiente e gradiente coniugato

3.2 Gradiente coniugato precondizionato

Vediamo adesso il metodo del gradiente coniugato può essere migliorato utilizzando un opportuno precondizionamento (PCG).
Dato che questo metodo funziona per matrici simmetriche definite positive, posso effettuare la fattorizzazione di Cholesky incompleta (funzione *ichol* su Matlab) che usa come precondizionatore $P = R^T R \approx A$, con R sparsa.

Tabella 1: Vediamo il numero di iterazioni a variare di n

Dimensione di A	**Grad**	**CG**	**PCG**
200	1107	116	5
300	1103	121	6
400	1109	124	6
500	1087	125	6
600	1103	126	7
700	1079	126	7
800	1085	127	8
900	1095	128	8
1000	1087	128	7
1100	1081	128	8
1200	1085	128	8
1300	1087	128	8
1400	1087	128	8
1500	1091	129	9
1600	1091	129	9
1700	1087	129	9
1800	1081	129	9
1900	1081	129	9
2000	1081	129	9

L'utilizzo del precondizionatore produce un abbattimento ulteriore del numero di iterazioni, anche se il tempo di calcolo rimane simile. Questo accade perché oltre a svolgere i passi dell'iterazione, impieghiamo del tempo anche per costruire il precondizionatore (Tabella 1 e Figura 2).

3.3 Ulteriori test

Proviamo a testare i nostri metodi con un altro tipo di matrici.
Queste matrici vengono generate dalla funzione $delsq(G)$ di Matlab, con $G = numgrid('S', nn)$, e sono il risultato della discretizzazione del problema dell'equazione di Laplace. La matrice che otteniamo è una matrice sparsa definita positiva con elementi non nulli solo su 5 diagonali (Figura 3). L'utilizzo di questo tipo di matrici ci permette di effettuare la sperimentazione su matrici di dimensione molto più grande rispetto a quanto non fosse possibile generando le matrici con la funzione $sprandsym$.
Ancora di più con matrici così grandi (siamo arrivati ad avere A di dimensione $10^6 \times 10^6$), l'utilizzo di un precondizionatore risulta essere fondamentale

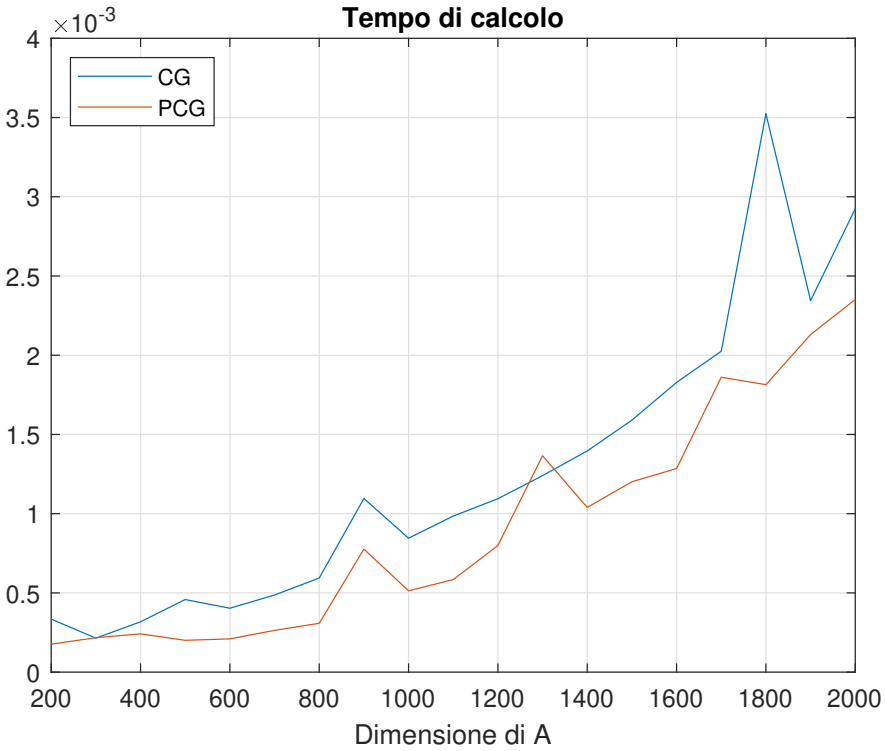

Figura 2: Confronto tra gradiente coniugato con e senza precondizionamento

per arrivare alla convergenza senza avere bisogno di un numero enorme di iterazioni (Figura 4 e Tabella 2).

Tabella 2: Vediamo il numero di iterazioni a variare di n

Dimensione di A	**CG**	**PCG**
9604	223	25
39204	446	47
88804	666	69
158404	885	89
248004	1104	107
357604	1320	121
487204	1536	139
636804	1751	156
806404	1964	174
996004	2176	192

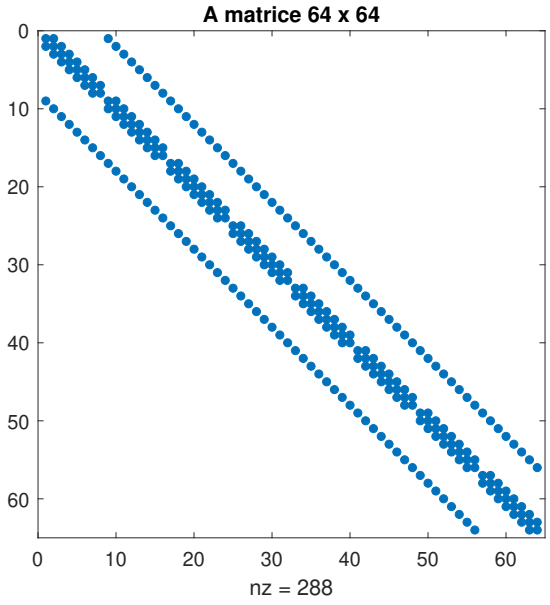

Figura 3: Visualizzazione con la funzione spy di Matlab

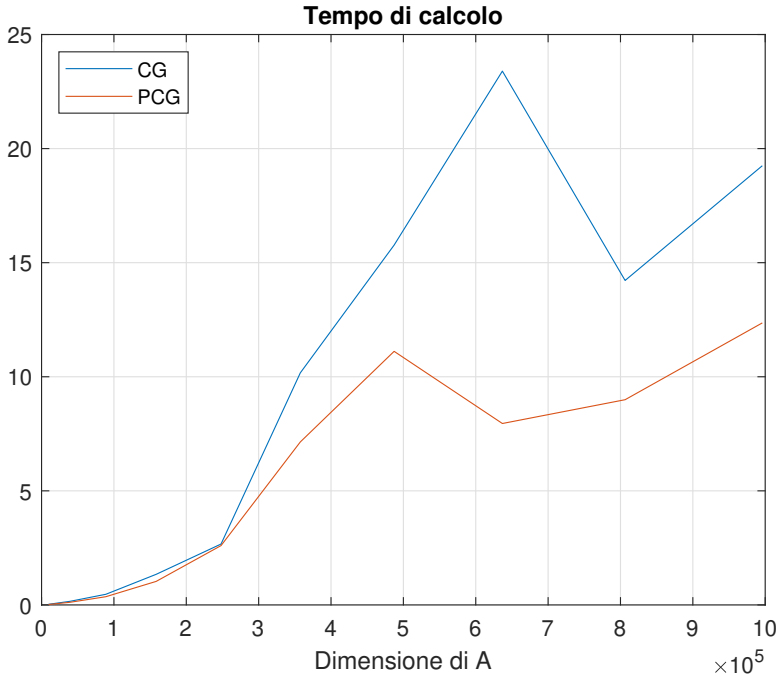

Figura 4: Confronto tra gradiente coniugato con e senza precondizionamento

4 Conclusioni

Abbiamo quindi visto quali sono i vantaggi dell'utilizzo di metodi iterativi per la risoluzione di sistemi lineari con matrice dei coefficienti sparsa e di dimensione elevata. In particolare il metodo del gradiente risulta significativo da un punto di vista didattico perché utilizza un'idea semplice e interessante, ma nella pratica può essere notevolmente migliorato con il metodo del gradiente coniugato.
Inoltre l'utilizzo di un precondizionatore porta ad un ulteriore aumento della velocità di convergenza del metodo.
L'importanza dell'utilizzo di opportuni metodi iterativi cresce all'aumentare della dimensione del problema, infatti nell'ultimo esempio che abbiamo visto la dimensione era $n \simeq 10^6$, quindi se avessimo provato a risolverlo con il metodo di Gauss la complessità computazionale sarebbe stata dell'ordine di $\frac{1}{3}10^{18}$, che con qualsiasi processore richiederebbe un tempo di calcolo spaventoso (oltre a comportare una grande propagazione degli errori dovuta all'enorme numero di operazioni).

Riferimenti bibliografici

[1] G. Rodriguez. *Algoritmi Numerici*. Pitagora Editrice Bologna, 2008.