Università degli studi di Cagliari
Corso di Laurea in Ingegneria per l'Ambiente ed il Territorio
A.A. 2003/2004

Studio dell'andamento degli errori prodotti dai metodi per la risoluzione di sistemi lineari di Gauss, Fattorizzazione LU pivotata, Fattorizzazione QR, Metodo del Gradiente Coniugato al variare delle dimensioni della matrice dei coefficienti

Lavoro di:

Manuela Di Mauro

Luisa Manigas

Stefano Milia

Il presente lavoro ha lo scopo di porre a confronto alcuni dei metodi per la risoluzione di un sistema lineare appresi durante il corso di Calcolo Numerico. Sono stati utilizzati come termini di paragone gli errori prodotti da ogni metodo, valutati tramite norma due della differenza tra risultato calcolato e soluzione vera, ed il tempo che ogni metodo impiega nel calcolo, utilizzando matrici random quadrate di dimensioni crescenti, rese definite positive per le ragioni che saranno spiegate in seguito, e matrici di Hilbert, notoriamente mal condizionate.

```
% Gauss senza Pivoting

function errGauss=Gauss(A,b,e);
n=length(A);
for k=1:n-1
    for i=k+1:n
        m=A(i,k)/A(k,k);
        for j=k+1:n
            A(i,j)=A(i,j)-m*A(k,j);
        end
        b(i)=b(i)-m*b(k);
        A(i,k)=0;
    end

end
z=sistris(A,b);
errGauss=norm(z-e);
```

Il primo algoritmo in esame è l'algoritmo di triangolarizzazione di Gauss, che è un algoritmo che trasforma un sistema in un altro triangolare. Per fare ciò si basa sulle operazioni elementari, che sono quelle che non variano il risultato, ossia:

$$\boldsymbol{EQ}1 \rightarrow \alpha \cdot \boldsymbol{EQ}2$$
$$\boldsymbol{EQ}1 \rightarrow \boldsymbol{EQ}1 + \boldsymbol{EQ}2$$
$$\boldsymbol{EQ}1 \leftrightarrow \boldsymbol{EQ}2$$

Infatti, moltiplicando per un numero l'equazione non varia il risultato, purché si moltiplichi anche il termine noto; si può sostituire al posto di un'equazione la somma di questa e un'altra; le equazioni (ed i relativi termini noti) possono essere scambiate fra loro.

Procedendo in questo modo l'algoritmo rende la matrice di partenza una upper triangular e la risoluzione del sistema costa:

$$2 \cdot \frac{n^3}{3}$$

dove n è la dimensione della matrice.

Il problema dalla triangolarizzazione di Gauss è che ha una potenziale instabilità: all'i-esimo passo divide gli elementi della colonna i-esima per il termine che occupa la posizione (i,i), ossia per il termine in diagonale. pertanto se tale termina fosse nullo o molto piccolo, avrei un'instabilità o una forte propagazione degli errori. Per cui il metodo va bene, ad esempio, se la matrice è:

- Definita positiva
- Diagonalmante dominante $|a_{ii}| > \sum_{j=1}^{n} |a_{ij}|$

Per essere certi che l'algoritmo lavori bene e non propaghi errori elevati, si deve evitare che l'elemento in diagonale sia nullo o troppo piccolo. Per questo motivo si utilizza l'algoritmo di Gauss con Pivoting, nel quale si utilizza una matrice di permutazione che scambia la riga avente un elemento minore di una soglia determinata in diagonale con un'altra in cui questo non accade. Per fare ciò moltiplica la matrice A con una matrice di scambio che è simmetrica ed ortogonale. Per effettuare tutti gli scambi necessari sarà necessario moltiplicare A per più matrici di scambio, ovvero moltiplicare A per il prodotto di tali matrici, ossia una matrice detta, appunto, matrice di permutazione che è ancora ortogonale ma non è simmetrica. Inoltre, poiché l'algoritmo di Gauss, visto dal punto di vista matriciale, spezza la matrice A in due matrici triangolari, una inferiore ed una superiore, senza nessun costo, si può vedere l'operazione di pivoting come la suddivisione della matrice prodotto della matrice A per quella di permutazione in due matrici triangolari, una inferiore ed una superiore.

$$P \cdot A = L \cdot U$$

In Matlab questo algoritmo è già implementato. Il vantaggio di vedere il sistema in questo modo è che esso può essere risolto come:

$$\boldsymbol{P \cdot A \cdot = L \cdot U \rightarrow A \cdot x = b \rightarrow L \cdot U \cdot x = P \cdot b}$$

$$\boldsymbol{L \cdot g = P \cdot b}$$

$$\boldsymbol{U \cdot x = g}$$

Pertanto si devono risolvere due problemi triangolari di complessità $\boldsymbol{n^2}$ per cui la complessità totale sarà $2 \cdot \frac{\boldsymbol{n^3}}{3} + \boldsymbol{n^2}$ come prima.

Per quanto riguarda il condizionamento, però:

$$\boldsymbol{K_\infty(U)} \leq 4^{\boldsymbol{n}-1} \cdot \boldsymbol{K_\infty(A)}$$

L'altra fattorizzazione presa in considerazione è la fattorizzazione QR in cui la matrice A è scomposta nelle matrici Q, ortogonale, ed R, triangolare superiore. Questa fattorizzazione ha un vantaggio molto forte per quanto riguarda il condizionamento. Infatti, essendo Q una matrice ortogonale e pertanto unitaria, ed essendo:

$$\|\boldsymbol{A}\| \leq \|\boldsymbol{Q} \cdot \boldsymbol{R}\| \leq \|\boldsymbol{Q}\| \cdot \|\boldsymbol{R}\| \rightarrow \|\boldsymbol{A}\| \leq \|\boldsymbol{R}\|$$

$$\|\boldsymbol{R}\| \leq \left\|\boldsymbol{Q}^{\boldsymbol{T}}\right\| \cdot \|\boldsymbol{A}\| \leq \|\boldsymbol{A}\| \Rightarrow \|\boldsymbol{A}\| = \|\boldsymbol{R}\| \rightarrow \left\|\boldsymbol{A}^{-1}\right\| = \left\|\boldsymbol{R}^{-1}\right\|$$

$$\boldsymbol{K}_2(\boldsymbol{A}) = \|\boldsymbol{A}\| \cdot \left\|\boldsymbol{A}^{-1}\right\| = \|\boldsymbol{R}\| \cdot \left\|\boldsymbol{R}^{-1}\right\| = \boldsymbol{K}_2(\boldsymbol{R})$$

Ossia questa fattorizzazione mi restituisce lo stesso condizionamento. Quindi si ha un grande vantaggio nel condizionamento che però si paga con il doppio delle operazioni rispetto a Gauss. Questo fatto, come si vede in seguito negli esempi, è molto pesante nella propagazione degli errori.

L'ultimo algoritmo utilizzato è quello del Metodo del Gradiente Coniugato, implementato con Matlab come CGS. Questo metodo è stato realizzato per matrici definite positive. L'idea di questo metodo è quella di superare l'andamento del Gradiente che cambia direzione non appena il gradiente cambia di segno, ma va avanti finché non trova un punto ottimale per il cambiamento di direzione che accelera enormemente la

```
%GradCon

function errGC=GradCon(A,b,e)
z=cgs(A,b);
errGC=norm(z-e);
```

convergenza. Questo viene eseguito dall'algoritmo scegliendo un vettore ottimale. Un vettore x si dice ottimale rispetto a p se e solo se $r^T \cdot p = 0$ ossia se r è perpendicolare rispetto a p perché quest'ultima è la direzione di tangenza e r è la direzione di discesa.

Si procede in questo modo:

- si calcola il residuo e si pone pari a questo la prima direzione.
- le iterate successive si calcolano come $x^{(k+1)} = x^{(k)} + \alpha_k \cdot p^{(k)}$ dove $\alpha_k = \frac{p^{(k)T} \cdot r^k}{p^{(k)T} \cdot A \cdot p^{(k)}}$ e

$$r^{(k+1)} = r^{(k)} - \alpha_k \cdot A \cdot p^{(k)}$$

- Le direzioni successive sono $p^{(k+1)} = r^{(k+1)} - \beta_k \cdot p^k$ dove $\beta_k = \frac{p^{(k)T} \cdot A \cdot r^{k+1}}{p^{(k)T} \cdot A \cdot p^{(k)}}$

Questo metodo converge in N iterazioni, ossia in realtà è un metodo diretto, però la convergenza è molto rapida dopo le prime iterazioni, pertanto, fissata una soglia, si considera accettabile la soluzione.

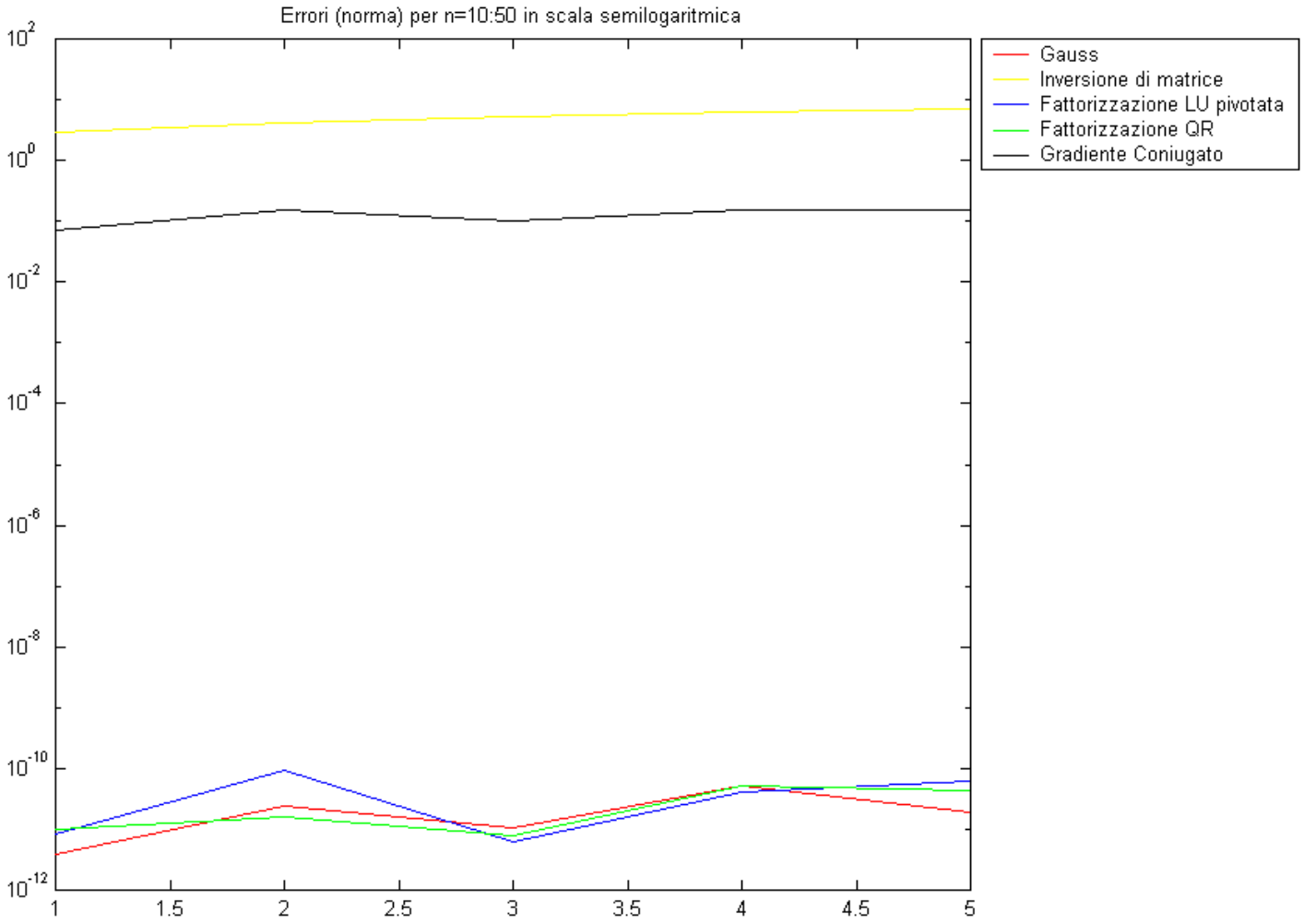
La prima applicazione realizzata utilizza una matrice Random modificata in modo da renderla definita positiva moltiplicando la trasposta della matrice creata per la matrice stessa:

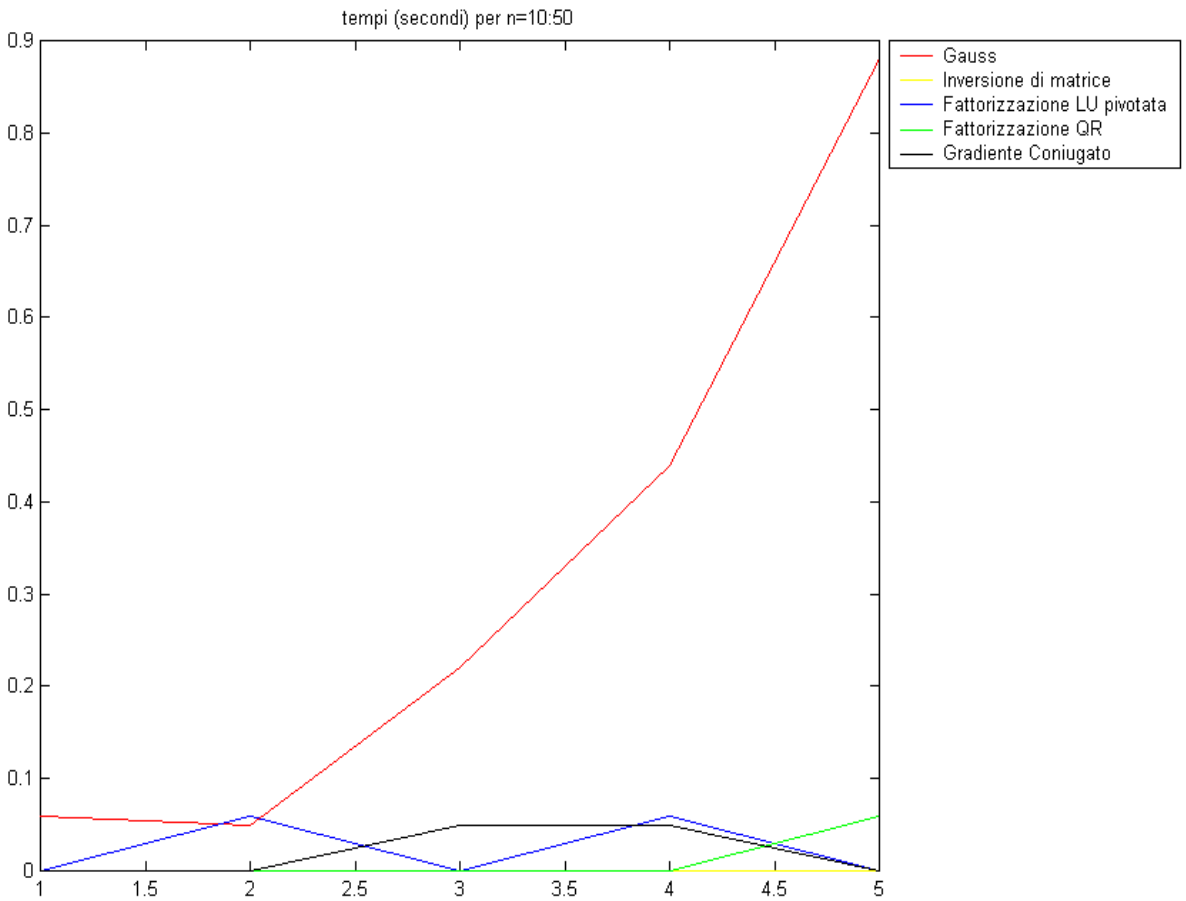
$$\boldsymbol{M = M^T \cdot M}$$

Sono state generate matrici di questo tipo con le dimensioni che vanno da 10 a 50 a passo 10. Nelle figure sottostanti sono riportati l'andamento degli errori, compresa anche l'applicazione della funzione che risolve il sistema tramite l'inversione di per consentire il confronto fra gli algoritmi utilizzati oltre all'inversione pura che, si ricorda, sono:

- L'algoritmo di Gauss senza Pivoting
- La fattorizzazione LU pivotata
- La fattorizzazione QR
- Il metodo del Gradiente Coniugato

Il grafico è plottato in scala semilogaritmica per consentire il confronto anche per errori che differiscono di molti ordini di grandezza.
E' da notare come l'errore sia massimo per l'inversione di matrice, data la complessità dell'operazione di inversione.


Errori (norma) per n=10:50 in scala semilogaritmica
10^{2}
10^{0}
10^{-2}
10^{-4}
10^{-6}
10^{-8}
10^{-10}
10^{-12}
1
1.5
2
2.5
3
3.5
4
4.5
5
Gauss
Inversione di matrice
Fattorizzazione LU pivotata
Fattorizzazione QR
Gradiente Coniugato

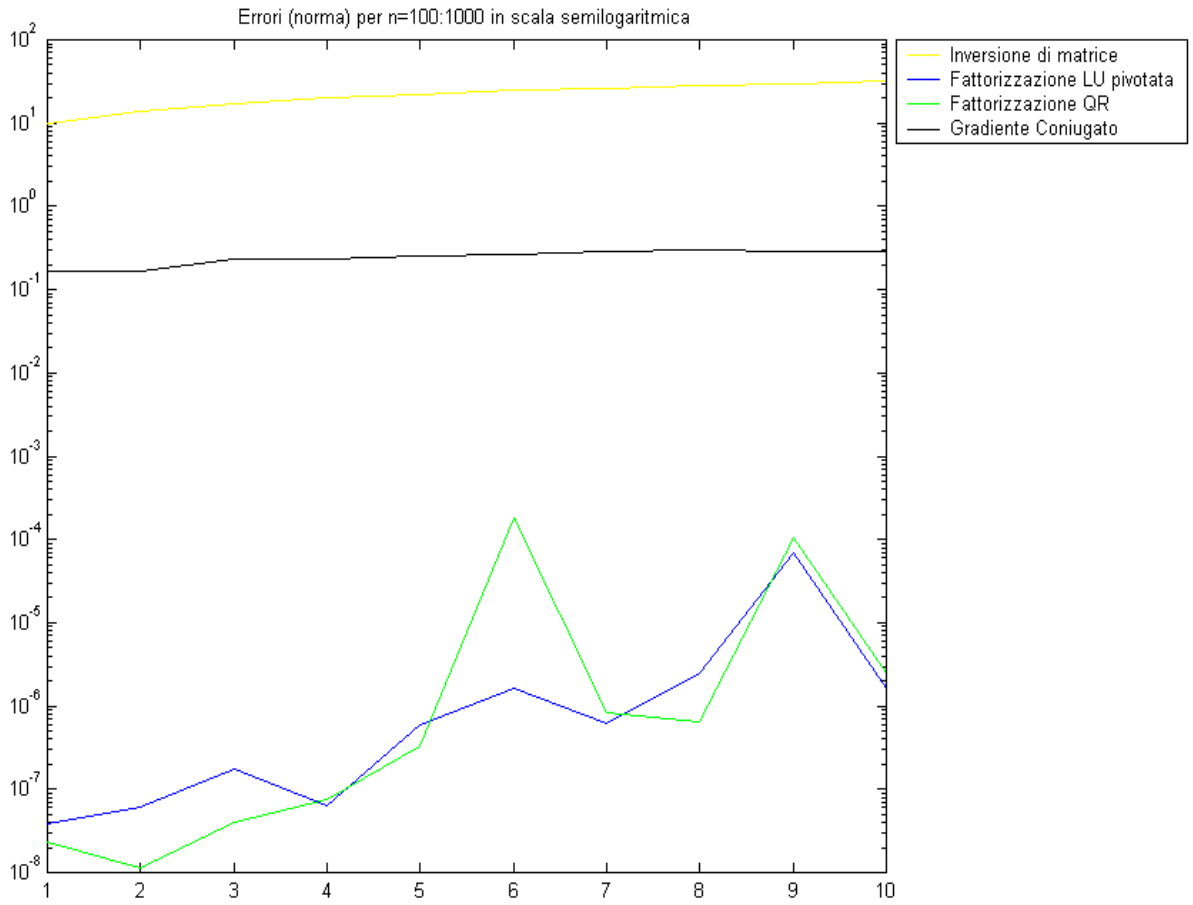


Nel grafico successivo è stato riportato l'andamento dei tempi di esecuzione del calcolo.

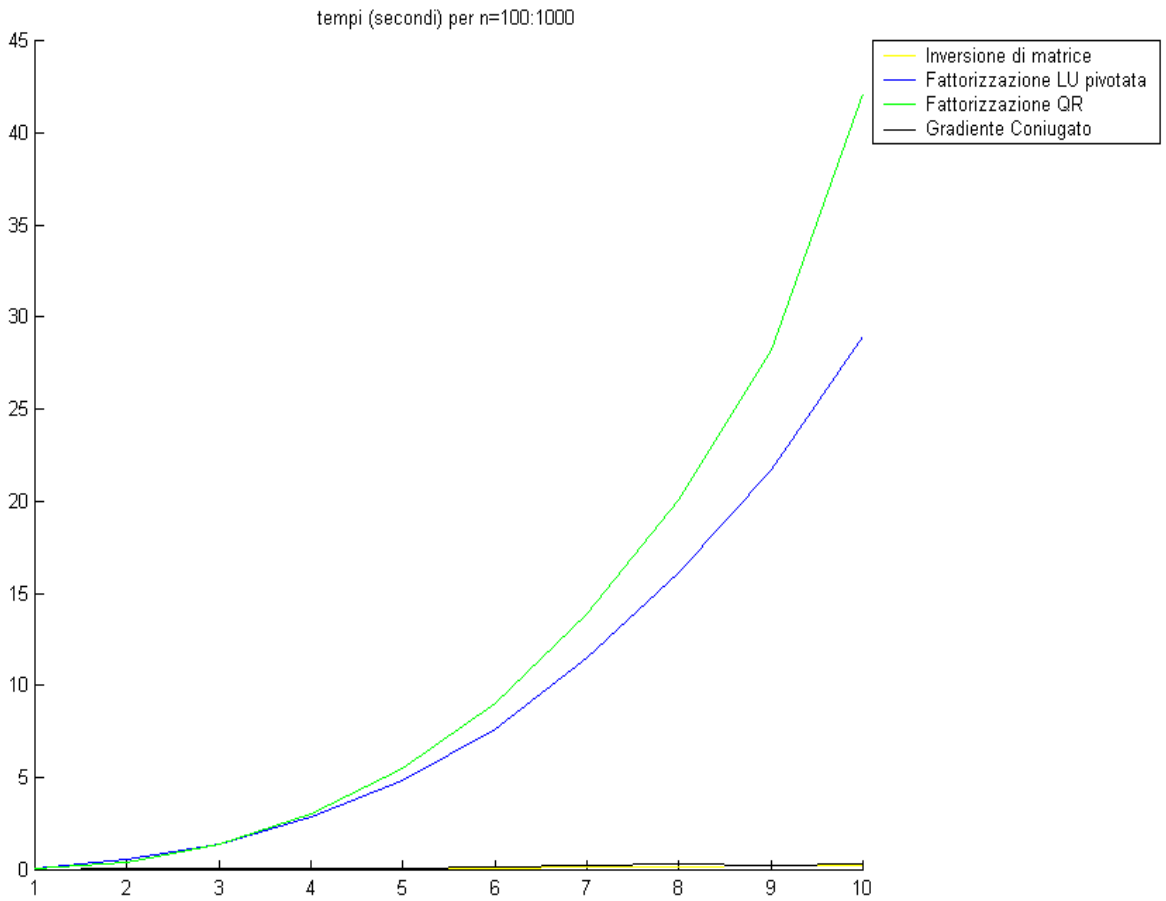
Si può notare come il metodo di Gauss, pur producendo un errore in norma molto piccolo, abbia dei tempi di calcolo estremamente lunghi, che crescono molto rapidamente all'aumentare delle dimensioni, seppure molto modeste, dei sistemi.

Questo aumento dei tempi ha implicato l'eliminazione del metodo di Gauss nel secondo esempio, in cui sono state fatte variare le dimensioni della

matrice da 100 a 1000 con passo 100. Il tempo di calcolo con Gauss si è rivelato essere dell'ordine di un'ora e mezza, il che ha portato alla logica conclusione di escludere quest'algoritmo dal confronto fra gli algoritmi nel caso in esame di per dimensioni superiori a 50 perché assolutamente non utilizzabile nella risoluzione di problemi reali.



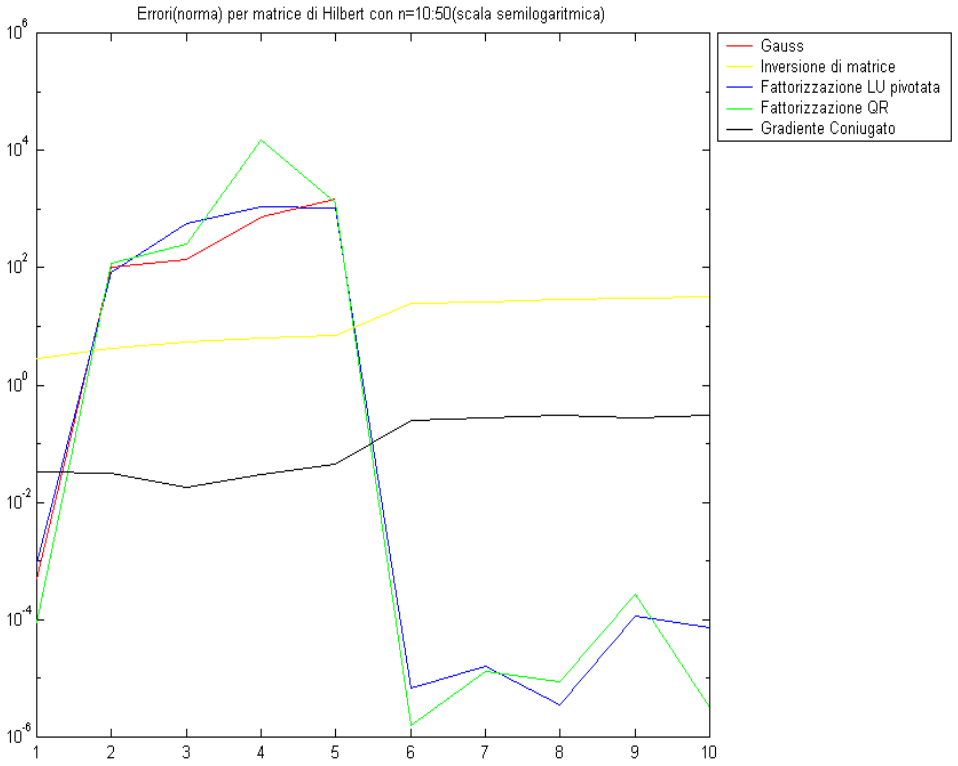
Si può notare come in entrambi i casi l'errore prodotto dal metodo del Gradiente Coniugato, essendo questo un metodo iterativo, è maggiore rispetto a quella prodotto dagli altri algoritmi (fattorizzazioni QR ed LU pivotata).L'errore più elevato è compensato decisamente dalla velocità di

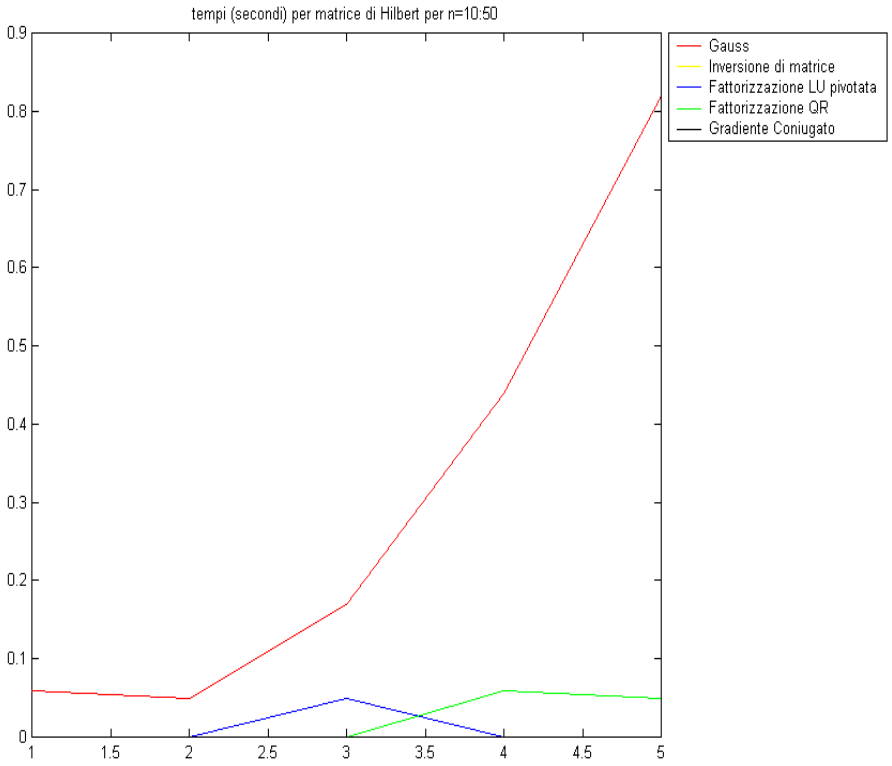


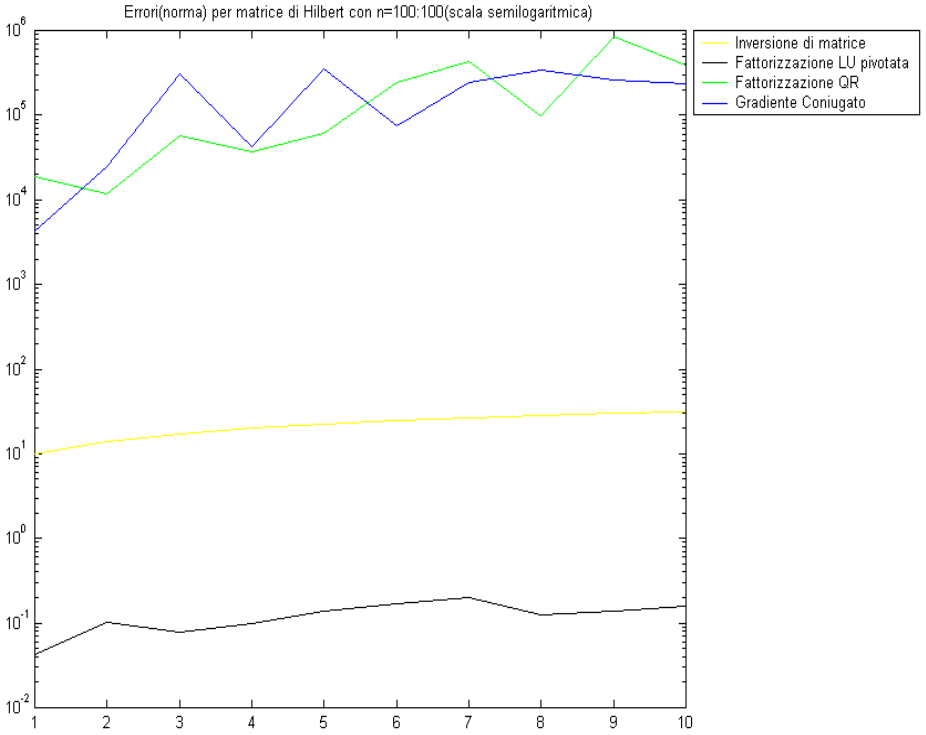
calcolo che, invece, non cresce in modo rilevante all'aumentare delle dimensioni. Questo non è assolutamente vero per le fattorizzazioni QR ed LU pivotata i cui errori, come è già stato notato prima, sono molto piccoli, ma i cui tempi di calcolo hanno una crescita quasi esponenziale.

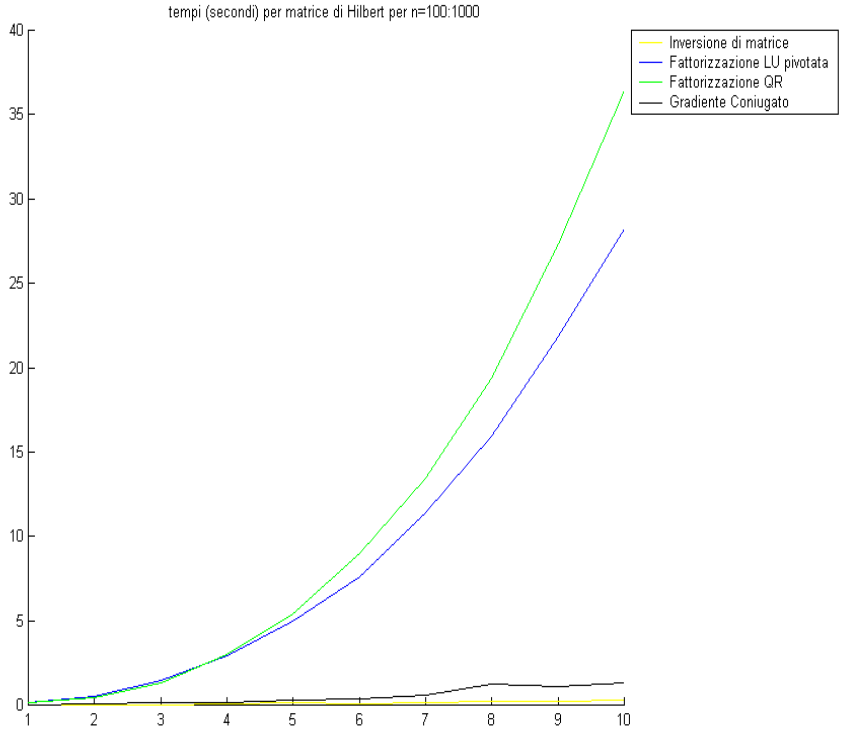
I maggiori tempi di esecuzione della fattorizzazione QR sono dovuti, come si è detto prima, al fatto che l'algoritmo richiede un numero elevato di operazioni. Per quanto riguarda quella LU, le motivazione è nell'algoritmo stesso che è simile a quello di Gauss.

Nell'esempio successivo è stata ripetuta la medesima procedura, utilizzando, però una matrice di Hilbert, notoriamente molto mal condizionata sia per dimensioni che vanno da 10 a 50 con passo 10 che per dimensioni che vanno da 100 a 1000 con passo 100, escludendo, anche in questo caso, l'algoritmo di Gauss per le dimensioni maggiori di 50, in quanto il tempo di esecuzione è enormemente più lungo, per cui valgono le osservazioni fatte precedentemente.









E' anche in questo caso da notare come l'ordina di grandezza dei tempi di esecuzione varia, nel passare da 50 a 1000, da 0,1 a 30-40 nel caso delle fattorizzazioni QR ed LU, mentre rimane praticamente stazionario per il metodo del Gradiente coniugato. Pertanto, dovendo trattare un sistema con una matrice definita positiva è conveniente utilizzare i primi due per dimensioni minori di 100, in quanto minimizzano la norma dell'errore rispetto agli altri algoritmi considerati (eccetto Gauss, da considerarsi, però inaccettabile per dimensioni maggiori di dieci), con tempi

di calcolo ancora accettabili. Trovare problemi con dimensioni minori di 10 è assolutamente irrealistico. Nelle situazioni più comuni che si devono affrontare nella realtà, infatti, le dimensioni della matrice dei coefficienti sono superiori al centinaio (solitamente dell'ordine delle migliaia o delle decine di migliaia). In questi casi è certamente preferibile sacrificare in precisione del risultato per avere, però, un netto miglioramento dei tempi di calcolo, che si riducono ad una scala accettabile utilizzando un algoritmo come il Gradiente Coniugato.